

BASIC COMPUTER CODING: RESPONSIVE DESIGN

2nd Edition



www.bibliotex.com

BASIC COMPUTER CODING: RESPONSIVE DESIGN

2ND EDITION



www.bibliotex.com

email: info@bibliotex.com

e-book Edition 2022

ISBN: 978-1-98467-607-8 (e-book)

This book contains information obtained from highly regarded resources. Reprinted material sources are indicated. Copyright for individual articles remains with the authors as indicated and published under Creative Commons License. A Wide variety of references are listed. Reasonable efforts have been made to publish reliable data and views articulated in the chapters are those of the individual contributors, and not necessarily those of the editors or publishers. Editors or publishers are not responsible for the accuracy of the information in the published chapters or consequences of their use. The publisher assumes no responsibility for any damage or grievance to the persons or property arising out of the use of any materials, instructions, methods or thoughts in the book. The editors and the publisher have attempted to trace the copyright holders of all material reproduced in this publication and apologize to copyright holders if permission has not been obtained. If any copyright holder has not been acknowledged, please write to us so we may rectify.

Notice: Registered trademark of products or corporate names are used only for explanation and identification without intent of infringement.

© 2022 3G E-learning LLC

In Collaboration with 3G E-Learning LLC. Originally Published in printed book format by 3G E-Learning LLC with ISBN 978-1-98465-900-2

EDITORIAL BOARD



Aleksandar Mratinković was born on May 5, 1988 in Arandjelovac, Serbia. He has graduated on Economic high school (2007), The College of Tourism in Belgrade (2013), and also has a master degree of Psychology (Faculty of Philosophy, University of Novi Sad). He has been engaged in different fields of psychology (Developmental Psychology, Clinical Psychology, Educational Psychology and Industrial Psychology) and has published several scientific works.



Dan Piestun (PhD) is currently a startup entrepreneur in Israel working on the interface of Agriculture and Biomedical Sciences and was formerly president-CEO of the National Institute of Agricultural Research (INIA) in Uruguay. Dan is a widely published scientist who has received many honours during his career including being a two-time recipient of the Amit Golda Meir Prize from the Hebrew University of Jerusalem, his areas of expertise includes stem cell molecular biology, plant and animal genetics and bioinformatics. Dan's passion for applied science and technological solutions did not stop him from pursuing a deep connection to the farmer, his family and nature. Among some of his interest and practices counts enjoying working as a beekeeper and onboard fishing.



Hazem Shawky Fouda has a PhD. in Agriculture Sciences, obtained his PhD. From the Faculty of Agriculture, Alexandria University in 2008, He is working in Cotton Arbitration & Testing General Organization (CATGO).



Felecia Killings is the Founder and CEO of LiyahAmore Publishing, a publishing company committed to providing technical and educational services and products to Christian Authors. She operates as the Senior Editor and Writer, the Senior Writing Coach, the Content Marketing Specialist, Editor-in-Chief to the company's quarterly magazine, the Executive and Host of an international virtual network, and the Executive Director of the company's online school for Authors. She is a former high-school English instructor and professional development professor. She possesses a Master of Arts degree in Education and a Bachelor's degree in English and African American studies.



Dr. Sandra El Hajj, Ph.D. in Health Sciences from Nova Southeastern University, Florida, USA is a health professional specialized in Preventive and Global Health. With her 12 years of education obtained from one of the most prominent universities in Beirut, in addition to two leading universities in the State of Florida (USA), Dr. Sandra made sure to incorporate interdisciplinary and multicultural approaches in her work. Her long years of studies helped her create her own miniature world of knowledge linking together the healthcare field with Medical Research, Statistics, Food Technology, Environmental & Occupational Health, Preventive Health and most noteworthy her precious last degree of Global Health. Till today, she is the first and only doctor specialized in Global Health in the Middle East area.



Fozia Parveen has a Dphil in Sustainable Water Engineering from the University of Oxford. Prior to this she has received MS in Environmental Sciences from National University of Science and Technology (NUST), Islamabad Pakistan and BS in Environmental Sciences from Fatima Jinnah Women University (FJWU), Rawalpindi.



Igor Kronic 2003-2007 in the School of Economics. After graduating in 2007, he went on to study at The College of Tourism, at the University of Belgrade where he got his bachelor degree in 2010. He was active as a third-year student representative in the student parliament. Then he went on the Faculty of science, at the University of Novi Sad where he successfully defended his master's thesis in 2013. The crown of his study was the work titled "Opportunities for development of cultural tourism in Cacak". Later on, he became part of a multinational company where he got promoted to a deputy director of logistic. Nowadays he is a consultant and writer of academic subjects in the field of tourism.



Dr. Jovan Pehceviski obtained his PhD in Computer Science from RMIT University in Melbourne, Australia in 2007. His research interests include big data, business intelligence and predictive analytics, data and information science, information retrieval, XML, web services and service-oriented architectures, and relational and NoSQL database systems. He has published over 30 journal and conference papers and he also serves as a journal and conference reviewer. He is currently working as a Dean and Associate Professor at European University in Skopje, Macedonia.



Dr. Tanjina Nur finished her PhD in Civil and Environmental Engineering in 2014 from University of Technology Sydney (UTS). Now she is working as Post-Doctoral Researcher in the Centre for Technology in Water and Wastewater (CTWW) and published about eight International journal papers with 80 citations. Her research interest is wastewater treatment technology using adsorption process.



Stephen obtained his PhD from the University of North Carolina at Charlotte in 2013 where his graduate research focused on cancer immunology and the tumor microenvironment. He received postdoctoral training in regenerative and translational medicine, specifically gastrointestinal tissue engineering, at the Wake Forest Institute of Regenerative Medicine. Currently, Stephen is an instructor for anatomy and physiology and biology at Forsyth Technical Community College.



Michelle holds a Masters of Business Administration from the University of Phoenix, with a concentration in Human Resources Management. She is a professional author and has had numerous articles published in the Henry County Times and has written and revised several employee handbooks for various YMCA organizations throughout the United States.

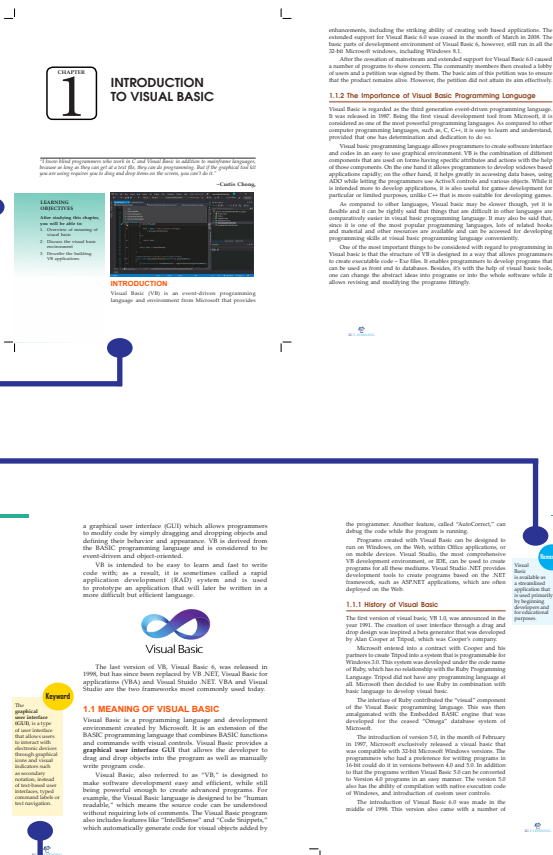
This book has been divided into many chapters. Chapter gives the motivation for this book and the use of templates. The text is presented in the simplest language. Each paragraph has been arranged under a suitable heading for easy retention of concept. Keywords are the words that academics use to reveal the internal structure of an author's reasoning. Review questions at the end of each chapter ask students to review or explain the concepts. References provides the reader an additional source through which he/she can obtain more information regarding the topic.

See what you are going to cover and what you should already know at the start of each chapter

An introduction is a beginning of section which states the purpose and goals of the topics which are discussed in the chapter. It also starts the topics in brief.

This revitalizes a must read information of the topic.

This section contains some important definitions that are discussed in the chapter. A keyword is an index entry that identifies a specific record or document. It also gives the extra information to the reader and an easy way to remember the word definition.



DID YOU KNOW?

This section equip readers the interesting facts and figures of the topic.

EXAMPLE

The book cabinets' examples to illustrate specific ideas in each chapter.

In each case, the name of the variable and its data type are provided as part of the declaration.

Visual Basic reserves the amount of memory required to hold the variable as soon as the declaration statement is executed. After a variable is declared, it is not possible to change its data type, although it is quite easy to convert the value of a variable and assign the converted value to another variable.

2.2.2 Comparing Implicit and Explicit Variable Performance

The default data type for Visual Basic variables is the variant. This means that, unless you specify otherwise, every variable in your application will be a variant. The data type is not very efficient. Its data storage requirements are greater than the equivalent simple data type. The computer spends more time keeping track of the data type contained in a variant than for other data types.

Variable names can't be duplicated with the same scope. This means, that you can't have two variables of the same name within a procedure. You can, however, have two variables with the same name in two different procedures.

An explicit declaration statically types the variable it declares. In a language that requires explicit declaration, you will get a compilation error for any reference to a variable that has not been explicitly declared.

By contrast, in a language that supports implicit declaration, simply using a variable to make implies the declaration. If your code assigns a string to the variable, then it is declared to be a string.

Convenient, yes? Not so much. Any time you misreport a variable name you get a new one and the program moves on, with no error conditional behavior or a strongly compiled value.

Given the rise of very smart editors like Visual Studio Code, implicit declaration need not be the menace it was, at least for languages that support the notion of optional

- There are some, fairly minor disadvantages compared with C. One better declaration of arrays is possible to initialize an array of structures in C at declaration time. This is impossible in VB.

1.2 VISUAL BASIC ENVIRONMENT

Did You Know?

Visual Basic 1.0 for Windows was released in September 1993. The language was designed to be quite compatible with Visual Basic 3.0 for Windows, as it is currently the only version of Microsoft's BASIC compiler, Microsoft QuickBASIC, supported and Microsoft Professional Development System 7.1. The language applied its own syntax rules to create the appearance of a C++.



Figure 1: The Visual Basic Start-up Dialog Box.

- In figure 2, the Visual Basic Environment consists of the
 - The Work Area window which you can design your application's interface.
 - The Project window displays the files that are created in your application.
 - The Properties window which displays the properties of various controls and objects that are created in your applications.
- It also includes a toolbar that consists of all the controls essential for developing a VB Application. Controls are tools

ROLE MODEL

A biography of someone who has/had acquired remarkable success in their respective field as Role Models are important because they give us the ability to imagine our future selves.

CASE STUDY

This reveals what students need to create and provide an opportunity for the development of key skills such as communication, group working and problem solving.

ROLE MODEL

ALAN COOPER: FATHER OF VISUAL BASIC



Born in San Francisco in 1952 and raised in Marin County, California, Alan Cooper has always taken the path less traveled. A rebellious teenager, he dropped out of high school, but eventually made his way to the College of Marin to pursue his interest in architecture. After an exploratory course in programming, it became clear that his future was in architecture—software architecture. After getting his associate degree and a CIBAC programming job, he saw an advertisement for one of the first personal computers and conceived an idea for a new business venture.

In 1976, Cooper founded Structured Systems Group (SSG), a company firm in the Valley authors Paul Freiberger and Michael Swaine said created "the first serious business software for microcomputers." In four years, Cooper wrote and shipped a dozen application programs. SSG became the archetype for many software startups in the early days of the personal computer revolution.

During the 1980s, after leaving SSG, Cooper invented, wrote, and sold three major software packages to prominent publishers. One of those was the visual programming front-end code named "Ruby." In what became Visual BASIC, Bill Gates purchased it from Cooper in 1988, noting that it would have significant impact across Microsoft's entire product line. Visual BASIC was deemed both a commercial and critical success, earning Cooper the moniker "Father of Visual BASIC." Visual BASIC has influenced integrated development languages ever since.

In 1990 Cooper became fascinated with the challenge of making software products that were easy to use and understand. He and his wife, Susan, founded Cooper Interaction Design (now "Cooper") to assist in what Cooper calls "interaction design." In the design field, Cooper's software development background was unique and, over the next few years, he invented many of the tools and techniques now standard in the user experience industry, including personas and scenarios.

CASE STUDY

FUJITSU FACILITATES SMOOTH MIGRATION TO VB.NET AT AN POST

Fujitsu has an excellent technical team, which works closely with our staff. We have had a good working relationship for many years and Fujitsu has an in-depth knowledge of our mission-critical application gained from several years' development and support work.

Challenge

A Post, one of Ireland's largest companies, is a major commercial organization providing a wide range of postal, communication, retail and financial services. With 9,600 employees throughout its national network of retail, processing and delivery points, the business also provides services to government departments, the National Treasury Management Agency and its own National Lottery Company. A decade ago, A Post implemented a new nationwide time and attendance system to calculate and record staff salary and wages functions. The Staff Remuneration and Administration Management System (STREAMS) is a bespoke, mission-critical application developed by Fujitsu as a reliable, scalable client server system using Microsoft technologies. The STREAMS front-end system gathers information and feeds the data to the company's HR, payroll and financial departments. It primarily creates more efficient processes for A Post to capture data for the weekly payroll run whilst simultaneously minimizing the number of payroll queries by employees. Following deployment, STREAMS improved cost center reporting, significantly lowered the time to record pay details and enhanced the processing of casual staff pay. During this period, Fujitsu provided quality support and maintenance services and application enhancements to increase functionality, ensuring the long-term reliability of STREAMS. For instance, as employee numbers steadily increased to exceed original expectations, Fujitsu boosted system performance by upgrading the infrastructure and optimizing the software. STREAMS originally employed Visual Basic (VB), a third-generation event-driven programming language and integrated development environment (IDE) from Microsoft. IDE provides programmers with comprehensive facilities for software development and comprises a source code editor, a compiler and/or an interpreter, build automation tools and a debugger. However, Microsoft no longer supports VB version 6.0, the edition employed by A Post. Sri Byrne, IT Manager Remuneration Services, A Post, explains: "To ensure that our business-critical application is future-proof, we needed to move to a platform that Microsoft will support for the foreseeable future." A Post therefore decided to migrate STREAMS to the VRNET platform, an object-orientated programming language. This strategy would protect its investment for the next 10 years by creating a secure, scalable

KNOWLEDGE CHECK

This is given to the students for progress check at the end of each chapter.

REVIEW QUESTIONS

This section is to analyze the knowledge and ability of the reader.

REFERENCES

References refer those books which discuss the topics given in the chapters in almost same manner.

KNOWLEDGE CHECK

1. The Visual Basic Code Editor will automatically detect certain types of errors as you are entering code.

- a. True
- b. False

2. Keywords are also referred to as reserved words.

- a. True
- b. False

3. The `Str` function returns a string of the number, which is a number, but is not a string.

- a. True
- b. False

4. Visual Basic responds to events using which of the following?

- a. a code procedure
- b. an event procedure
- c. a form procedure
- d. a property

5. When the user clicks a button, _____ is triggered.

- a. an event
- b. a method
- c. a setting
- d. a property

6. What property of controls tells the order they receive the focus when the tab key is pressed during run time?

- a. Focus order
- b. Focus number
- c. Tab index
- d. Control order

7. Storing Handles make it very easy to realize virtually any control when developing applications with Visual Basic. When working in the Form Designer, how are these stored handles displayed?

- a. A rectangle with 4 arrows, one in each corner, around your control.
- b. A 3-D outline around your control.
- c. A rectangle with small squares around your control.

REVIEW QUESTIONS

1. What is Visual Basic? Why are importance of visual basic programming language?
2. What is Visual Basic environment?
3. Describe the structure of a visual basic application.
4. How to creating your first application?
5. Discuss the saving projects in VB.

Check Your Result

1. (a)	2. (a)	3. (a)	4. (b)	5. (a)
6. (c)	7. (c)	8. (a)	9. (a)	10. (b)

REFERENCES

- Cox, Philip T. Visual Programming Languages. In Encyclopedia of Computer Science and Engineering, B.V. Wah (Ed.) John Wiley & Sons Inc, Hoboken, (June 2008).
- Kindberg, Mikael. How Children Understand Concurrent Computation: Experiments from LOFT and HEP. Preprints, In 2001 IEEE Symposium on Human-Centric Computing Languages and Environments, Seoul, July, September 2001.
- Byrne, Barbara, Mary Lou Saffa and Margaret Burnett, The Impact of Software Engineering Research on Modern Programming Languages. In ACM Transactions on Software Engineering and Methodology, October, 2005, Pages 431 to 477.
- Storrie, Harold, VMQL: A Generic Visual Model Query Language. In IEEE Symposium on Visual Languages/Human-Centric Computing, Corvallis, Oregon, September 2008.
- Zhang, Kang. Visual Languages and Applications. In Research Manuscript, Springer, 2007.

TABLE OF CONTENTS

Preface

xiii

Chapter 1 Responsive Web Design

1

Introduction

1

1.1 Basic concept of Responsive Web Design

2

1.1.1 Importance of Responsive Web Designing

3

1.1.2 Challenges, and Other Approaches

4

1.1.3 Responsive vs. Adaptive vs. Mobile

5

1.2 The Features of Responsive Web Design

6

1.2.1 Adjusting Screen Resolution

6

1.2.2 Media Queries

13

1.2.3 Javascript

17

1.2.4 Showing or Hiding Content

18

1.2.5 Touchscreens vs. Cursors

21

1.2.6 A Showcase of Responsive Web Design

22

1.2.7 Responsive Design Techniques, Tools and Design Strategies

33

1.2.8 Design Process in the Responsive Age

35

Summary

41

Knowledge Check

42

Review Questions

43

References

44

Chapter 2 Rapid Prototyping

45

Introduction

45

2.1 Concept of Rapid prototyping

46

2.1.1 Importance of Rapid prototyping

48

2.1.2 Computer Aided Design

50

2.1.3 Rapid Prototyping: Process and Applications

52

2.1.4 History of Rapid Prototyping	53
2.1.5 Manufacturing Methods of Rapid Prototyping	54
2.1.6 The Advantages of Rapid Prototyping	56
2.1.7 Types of Prototype Technology	57
2.2 Types of Rapid Prototyping	59
2.2.1 Choose the Best Rapid Prototype Software	60
2.2.2 Stereo-lithography	61
2.2.3 Effective Prototyping for Quality Software	63
2.2.4 Prototype Tooling	64
2.2.5 Prototype Castings	65
2.2.6 Rapid Prototype Model	66
Summary	68
Knowledge Check	69
Review Questions	71
References	72

Chapter 3 Sketching **73**

Introduction	73
3.1 Concept of Sketchpad	74
3.1.1 Computer Sketching	77
3.1.2 How to Sketch By Computer	79
3.1.3 What Is a 3D Drawing Pad?	82
3.1.4 Fashion Sketches	84
3.2 The Role of Sketching in the Design Process	86
3.2.1 Uses for Sketching in Design	87
3.2.2 Related Work	92
3.2.3 Sketching and Sketch Design	94
3.2.4 Computer-based Sketching	95
Summary	98
Knowledge Check	99
Review Questions	101
References	102

Chapter 4 HTML/CSS/JavaScript **103**

Introduction	103
4.1 Hypertext Markup Language	104
4.1.1 Basic HTML Document	106
4.1.2 HTML Page Structure	114

4.1.3 HTML <!DOCTYPE> Declaration	115
4.1.4 HTML Version	117
4.1.5 HyperText Markup Language (1991)	118
4.1.6 HTML 2.0	120
4.1.7 HTML 3.2	121
4.1.8 HTML 4.01	124
4.1.9 XHTML	131
4.1.10 HTML5	139
4.2 Cascading Style Sheets (CSS)	140
4.2.1 Meaning of CSS	140
4.2.2 History of CSS	141
4.2.3 Three Ways to Insert CSS	143
4.2.4 CSS Selectors	146
4.2.5 Descendant Combinator	149
4.2.6 Attribute Selector	152
4.2.7 Pseudo-class	153
4.3 Inheritance	153
4.3.1 How Inheritance Works	154
4.3.2 Forcing Inheritance	157
4.3.3 The Cascade	158
4.3.4 Specificity	160
4.3.5 Source Order	162
4.4 Concept of JavaScript	163
4.4.1 Meaning of JavaScript	163
4.4.2 The History of JavaScript	164
4.4.3 Client-side JavaScript	168
4.4.4 Hello World: Writing Your First JavaScript Program	170
4.4.5 Creating a “Hello World” JavaScript Function	171
Summary	176
Knowledge Check	177
Review Questions	179
References	180

Chapter 5 Mobile Content Strategy 181

Introduction	181
5.1 Importance of Mobile Content Marketing Strategy	182
5.1.1 Key to a Successful Mobile Content Strategy	183
5.1.2 Who Should Use a Mobile Content Marketing Strategy?	184

5.1.3 Mobile Content Marketing	185
5.2 Mobile-First Design	188
5.2.1 Benefits of Mobile First Design	189
5.2.2 Mobile first is Content First	190
5.2.3 Difference between Mobile-first and Responsive Design	192
5.2.4 How mobile-first Design is Different	195
5.2.5 Mobile First Design Methodology	198
Summary	200
Knowledge Check	201
Review Questions	203
References	204

Chapter 6 Style Tiles and Web Design Techniques **205**

Introduction	205
6.1 Concept of Web Technologies	206
6.1.1 What are Style Tiles?	207
6.1.2 Use Style Tiles	209
6.1.3 Web Design Techniques	212
6.1.4 Effective Web Designing Techniques and their Importance for Content Gurus	218
6.1.5 Web Technologies: Importance and Carrier	223
6.1.6 Careers in Web Technologies	225
6.1.7 Job Role of Web Developer	226
6.1.8 How the Website Works?	229
6.2 Client and Server Scripting Languages	230
6.2.1 Domains and Hosting	232
6.2.2 Websites Creation	234
6.2.3 Types of Websites	238
6.2.4 Web Standards	240
Summary	243
Knowledge Check	244
Review Questions	246
References	247

Chapter 7 Bootstrap **249**

Introduction	249
7.1 What is Bootstrap?	250
7.1.1 Bootstrap File Structure	251

7.1.2 Basic HTML Template	251
7.1.3 Global Styles	252
7.1.4 Default Grid System	252
7.1.5 Basic Grid HTML	253
7.1.6 Offsetting Columns	253
7.1.7 Nesting Columns	254
7.1.8 Fluid Grid System	254
7.1.9 Container Layouts	255
7.1.10 Responsive Design	255
7.2 Responsive Design for Bootstrap	256
7.2.1 Helper classes	257
7.3 Bootstrap CSS	258
7.3.1 Typography	258
7.3.2 Headings	258
7.3.3 Lead Body Copy	259
7.3.4 Emphasis	259
7.3.5 Bold	259
7.3.6 Italics	259
7.3.7 Emphasis Classes	259
7.3.8 Abbreviations	260
7.3.9 Addresses	261
7.4 Blockquotes	261
7.4.1 Lists	262
7.4.2 Code	265
7.4.3 Tables	265
7.4.4 Optional Table Classes	266
7.5 Forms	268
7.5.1 Optional Form Layouts	269
7.5.2 Supported Form Controls	271
7.5.3 Extended Form Controls	273
7.5.4 Prepended and Appended Inputs	273
7.5.5 Form Control Sizing	275
7.5.6 Form Actions	278
7.5.7 Form Control States	279
7.6 Buttons	281
7.6.1 Button Sizes	281
7.6.2 Disabled Button Styling	282
7.6.3 Images	283
7.6.4 Icons	283

Summary	285
Knowledge Check	286
Review Questions	287
References	288

Index	289
--------------	------------

PREFACE

In the field of Web design and development, we're quickly getting to the point of being unable to keep up with the endless new resolutions and devices. For many websites, creating a website version for each resolution and new device would be impossible, or at least impractical. Responsive Web design is the approach that suggests that design and development should respond to the user's behavior and environment based on screen size, platform and orientation. This book introduces students to responsive web design using HTML (Hypertext Markup Language) and CSS (Cascading Style Sheets). Throughout the book students are introduced to planning and designing effective web pages; implementing web pages by writing HTML, CSS code, and JavaScript; enhancing web pages with the use of page layout techniques, text formatting, graphics, images, and multimedia; and producing a functional, multi-page website.

Organization of the Book

This edition is organized into seven chapters. In this book you'll learn the fundamentals of responsive web design using the viewport tag and CSS media queries. Additionally, you will learn how to apply concepts from interaction design and human computer interaction in order to design and build an interactive, professional looking website.

Chapter 1 discusses the concept and features of responsive web design that makes use of flexible layouts, flexible images and cascading style sheet media queries.

Chapter 2 informs about the Rapid Prototyping (RP) as a group of techniques used to quickly fabricate a scale model of a part or assembly using three-dimensional computer aided design (CAD) data.

In **Chapter 3** you will know the concept of sketchpad and the role of sketching in the design process.

Chapter 4 focuses on the Hypertext Markup Language (HTML), the standard markup language for creating web pages and web applications, including with Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web.

Chapter 5 focuses on why is it important to understand the mobile content strategy means more than having content that fits the typical screen on a mobile device. While a responsive site can be part of a mobile content marketing strategy, it's definitely not the entire kit and caboodle.

In **Chapter 6** you will know the importance of style tiles, which are the wonderful visual concept created by Samantha Warren that help form a common visual language between the designers and the stakeholders. This can then be used as a catalyst for conversations between the client and the designers.

Chapter 7 focuses on Bootstrap. Bootstrap enables you to create responsive websites without you needing to do the “responsive” bit. Bootstrap takes care of that.

CHAPTER 1

RESPONSIVE WEB DESIGN

“A successful website does three things: It attracts the right kinds of visitors. Guides them to the main services or product you offer. Collect Contact details for future ongoing relation.”

–Mohamed Saad

LEARNING OBJECTIVES

After studying this chapter,
you will be able to:

1. Discuss the concept of responsive web design
2. Explain the features of responsive web design



INTRODUCTION

Responsive web design is a design that adjusts and adapts to any viewport (user's visible area of the web page) rather than only a specific viewport, irrespective of the visualization device.

The idea of web sites able to adapt to different viewports was irrelevant for quite a lot of time mainly because the only way to surf the Internet was through a desktop computer and the size of its screens it is pretty much standard.

With the advent of smartphones and its rapid overcrowding, things changed rapidly, as you know the size of the screen may vary a lot from device to device in this sector and then a problem arose: web sites did not look good on small screens. In today's world Internet access through mobile devices is increasing each year and according to some sources, more users acceded the when from devices like smartphones and tablets than they did from desktops or laptops.

Developers had to think in new ways to display web content and make it look good in any sort of devices and viewports. The first ideas came up in 2008 when World Wide Web Consortium (W3C) published its Mobile Web Best Practices and it is been evolving since then until what we have today.

Responsive design is a huge thing in the UX design game nowadays. Everyone in the business knows that having a truly responsive design means good usability. It's non-negotiable.of having to be rewritten, which saves considerable development time.

1.1 BASIC CONCEPT OF RESPONSIVE WEB DESIGN

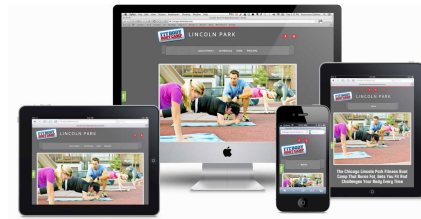
Responsive web design (RWD) is an approach to web design that makes web pages render well on a variety of devices and window or screen sizes. Recent work also considers the viewer proximity as part of the viewing context as an extension for RWD. Content, design and performance are necessary across all devices to ensure usability and satisfaction.

Responsive web design has become more important as the amount of mobile traffic now accounts for more than half of total internet traffic. Therefore, Google announced Mobilegeddon in 2015, and started to boost the ratings of sites that are mobile friendly if the search was made from a mobile device. Responsive web design is an example of user interface plasticity. The Internet took off quicker than anyone would have predicted, growing like crazy. Now, for the past few years, mobile growth has exploded onto the scene. The growth of mobile Internet usage is also far out pacing that of general Internet usage growth.

These days it is hard to find someone who does not own a mobile device, or multiple, connected to the Internet. In the UK there are more mobile phones than people, and should trends continue mobile Internet usage will surpass that of desktop Internet usage within the year. With the growth in mobile Internet usage comes the question of how to build websites suitable for all users. The industry response to this question has become responsive web design, also known as RWD.

1.1.1 Importance of Responsive Web Designing

Responsive Web design is the approach that suggests that design and development should respond to the user's behavior and environment based on screen size, platform and orientation. The practice consists of a mix of flexible grids and layouts, images and an intelligent use of CSS media queries. As the user switches from their laptop to iPad, the website should automatically switch to accommodate for resolution, image size and scripting abilities. In other words, the website should have the technology to automatically respond to the user's preferences. This would eliminate the need for a different design and development phase for each new gadget on the market. Responsive Web Design makes your web page look good on all devices (desktops, tablets, and phones). Responsive Web Design is about using CSS and HTML to resize, hide, shrink, enlarge, or move the content to make it look good on any screen:



Responsive web design (RWD) is an approach to web design aimed at crafting sites to provide an optimal viewing and interaction experience—easy reading and navigation with a minimum of resizing, panning, and scrolling—across a wide range of devices (from desktop computer monitors to mobile phones).

A site designed with RWD adapts the layout to the viewing environment by using fluid, proportion-based grids, flexible images, and CSS3 media queries, an extension of the media rule, in the following ways:

- The fluid grid concept calls for page element sizing to be in relative units like percentages, rather than absolute units like pixels or points.
- Flexible images are also sized in relative units, so as to prevent them from displaying outside their containing element.
- Media queries allow the page to use different CSS style rules based on characteristics of the device the site is being displayed on, most commonly the width of the browser.

Responsive web design is becoming more important as the amount of mobile traffic now accounts for more than half of total internet traffic. This trend is so prevalent that Google has begun to boost the ratings of sites that are mobile friendly if the search was made from a mobile device. This has the net effect of penalizing sites that are not mobile friendly.

Keyword

Progressive enhancement is a strategy for web design that emphasizes core webpage content first.

Remember

Feature detection also might not be completely reliable; some may report that a feature is available, when it is either missing or so poorly implemented that it is effectively nonfunctional.

Mobile first, Unobtrusive JavaScript, and Progressive Enhancement

“Mobile first”, unobtrusive JavaScript, and **progressive enhancement** are related concepts that predate RWD. Browsers of basic mobile phones do not understand JavaScript or media queries, so a recommended practice is to create a basic web site and enhance it for smart phones and PCs, rather than rely on graceful degradation to make a complex, image-heavy site work on mobile phones.

Progressive Enhancement Based on Browser, Device, or Feature Detection

Due to the high volume of usage of internet on mobile devices they can no longer be ignored. In 2014, for the first time more users accessed the internet from their mobile devices than desktop. Where a web site must support basic mobile devices that lack JavaScript, browser (“user agent”) detection (also called “browser sniffing”) and mobile device detection are two ways of deducing if certain HTML and CSS features are supported (as a basis for progressive enhancement)—however, these methods are not completely reliable unless used in conjunction with a device capabilities database.

For more capable mobile phones and PCs, JavaScript frameworks like Modernizr, jQuery, and jQuery Mobile that can directly test browser support for HTML/CSS features (or identify the device or user agent) are popular. Polyfills can be used to add support for features—e.g. to support media queries (required for RWD), and enhance HTML5 support, on Internet Explorer.

1.1.2 Challenges, and Other Approaches

Luke Wroblewski has summarized some of the RWD and mobile design challenges, and created a catalog of multi-device layout patterns. He suggests that, compared with a simple RWD approach, device experience or RESS (responsive web design with server-side components) approaches can provide a user experience that is better optimized for mobile devices. Server-side “dynamic CSS” implementation of stylesheet languages like Sass or Incentivated’s MML can be part of such an approach

by accessing a server based API which handles the device (typically mobile handset) differences in conjunction with a device capabilities database in order to improve usability. RESS is more expensive to develop, requiring more than just client-side logic, and so tends to be reserved for organizations with larger budgets. Google recommends responsive design for smartphone websites over other approaches.

Although many publishers are starting to implement responsive designs, one ongoing challenge for RWD is that some banner advertisements and videos are not fluid. However, search advertising and (banner) display advertising support specific device platform targeting and different advertisement size formats for desktop, smartphone, and basic mobile devices. Different landing page URLs can be used for different platforms, or Ajax can be used to display different advertisement variants on a page. CSS tables permit hybrid fixed+fluid layouts.

There are now many ways of validating and testing RWD designs, ranging from mobile site validators and mobile emulators to simultaneous testing tools like Adobe Edge Inspect. The Chrome, Firefox and Safari browsers and the Chrome console offer responsive design viewport resizing tools, as do third parties.

Use cases of RWD will now expand further with increased mobile usage, organic .search engine visits in the US coming from mobile devices has hit 51% and are increasing

1.1.3 Responsive vs. Adaptive vs. Mobile

The term *responsive* may not be new, and others might be even more acquainted with the terms *adaptive* or *mobile*. Which may leave you wondering what exactly the difference between all of them is.

Responsive and adaptive web design are closely related, and often transposed as one in the same. Responsive generally means to react quickly and positively to any change, while adaptive means to be easily modified for a new purpose or situation, such as change. With responsive design websites continually and fluidly change based on different factors, such as viewport width, while adaptive websites are built to a group of preset factors. A combination of the two is ideal, providing the perfect formula for functional websites. Which term is used specifically does not make a huge difference.

Mobile, on the other hand, generally means to build a separate website commonly on a new domain solely for mobile users. While this does occasionally have its place, it normally is not a great idea. Mobile websites can be extremely light but they do come with the dependencies of a new code base and browser sniffing, all of which can become an obstacle for both developers and users.

Currently the most popular technique lies within responsive web design, favoring design that dynamically adapts to different browser and device viewports, changing layout and content along the way. This solution has the benefits of being all three, responsive, adaptive, and mobile.

1.2 THE FEATURES OF RESPONSIVE WEB DESIGN

Remember

It should not require countless custom-made solutions for each new category of users.

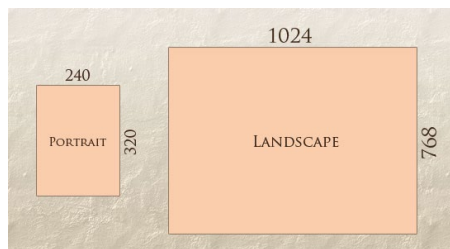
Transplant this discipline onto Web design, and we have a similar yet whole new idea. Why should we create a custom Web design for each group of users; after all, architects do not design a building for each group size and type that passes through it? Like responsive architecture, Web design should automatically adjust.

Obviously, we cannot use motion sensors and robotics to accomplish this the way a building would. Responsive Web design requires a more abstract way of thinking. However, some ideas are already being practiced: fluid layouts, media queries and scripts that can reformat Web pages and mark-up effortlessly (or *automatically*).

But responsive Web design is not only about adjustable screen resolutions and automatically resizable images, but rather about a whole new way of thinking about design. Let's talk about all of these features, plus additional ideas in the making.

1.2.1 Adjusting Screen Resolution

With more devices come varying screen resolutions, definitions and orientations. New devices with new screen sizes are being developed every day, and each of these devices may be able to handle variations in size, functionality and even color. Some are in landscape, others in portrait, still others even completely square. As we know from the rising popularity of the iPhone, iPad and advanced smartphones, many new devices are able to switch from portrait to landscape at the user's whim. How is one to design for these situations?



In addition to designing for both landscape and portrait (and enabling those orientations to possibly switch in an instant upon page load), we must consider the hundreds of different screen sizes. Yes, it is possible to group them into major categories, design for each of them, and make each design as flexible as necessary. But that can be overwhelming, and who knows what the usage figures will be in five years? Besides, many users do not maximize their browsers, which itself leaves far too much room for variety among screen sizes.

Since then even more devices have come out. It's obvious that we cannot keep creating custom solutions for each one. So, how do we deal with the situation?

Part of the Solution: Flexible Everything

A few years ago, when flexible layouts were almost a "luxury" for websites, the only things that were flexible in a design were the layout columns (structural elements) and the text. Images could easily break layouts, and even flexible structural elements broke a layout's form when pushed enough. Flexible designs were not really that flexible; they could give or take a few hundred pixels, but they often could not adjust from a large computer screen to a netbook.

Now we can make things more flexible. Images can be automatically adjusted, and we have workarounds so that layouts never break (although they may become squished and illegible in the process). While it's not a complete fix, the solution gives us far more options. It's perfect for devices that switch from portrait orientation to landscape in an instant or for when users switch from a large computer screen to an iPad.

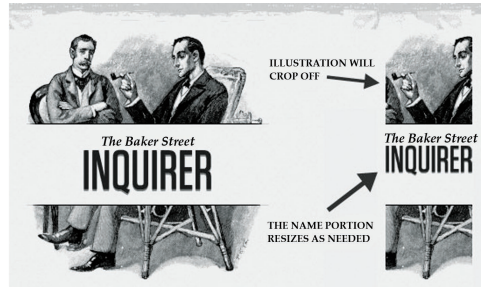
The created a sample Web design that features this better flexible layout:

The entire design is a lovely mix of fluid grids, fluid images and smart mark-up where needed. Creating fluid grids is fairly common practice, and there are a number of techniques for creating fluid images:

- Hiding and Revealing Portions of Images
- Creating Sliding Composite Images
- Foreground Images That Scale With the Layout

"Flexible Web Design: Creating Liquid and Elastic Layouts with CSS" by Zoe Mickley Gillenwater, and download the sample "Creating Flexible Images." In addition, Zoe provides the following extensive list of tutorials, resources, inspiration and best practices on creating flexible grids and layouts: "Essential Resources for Creating Liquid and Elastic Layouts."

While from a technical perspective this is all easily possible, it's not just about plugging these features in and being done. Look at the logo in this design, for example:



If resized too small, the image would appear to be of low quality, but keeping the name of the website visible and not cropping it off was important. So, the image is divided into two: one (of the illustration) set as a background, to be cropped and to maintain its size, and the other (of the name) resized proportionally.

```
<h1 id="logo"><a href="#"></a></h1>
```

Above, the h1 element holds the illustration as a background, and the image is aligned according to the container's background (the heading).

This is just one example of the kind of thinking that makes responsive Web design truly effective. But even with smart fixes like this, a layout can become too narrow or short to look right. In the logo example above (although it works), the ideal situation would be to not crop half of the illustration or to keep the logo from being so small that it becomes illegible and "floats" up.

Flexible Images

One major problem that needs to be solved with responsive **Web design** is working with images. There are a number of techniques to resize images proportionately, and many are easily done. The most popular option, noted on fluid images but first experimented with by Richard Rutter, is to use CSS's `max-width` for an easy fix.

```
img { max-width: 100%; }
```

As long as no other width-based image styles override this rule, every image will load in its original size, unless the viewing area becomes narrower than the image's original width. The **maximum width** of the image is set to 100% of the screen

Keyword

Web design

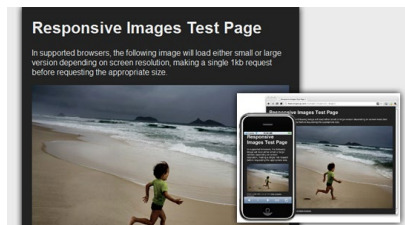
is a process of conceptualizing, planning, and building a collection of electronic files that determine the layout, colors, text styles, structure, graphics, images, and use of interactive features that deliver pages to your site visitors.

or browser width, so when that 100% becomes narrower, so does the image. Essentially, as Jason Grigsby noted, “The idea behind fluid images is that you deliver images at the maximum size they will be used at. You do not declare the height and width in your code, but instead let the browser resize the images as needed while using CSS to guide their relative size”. It’s a great and simple technique to resize images beautifully. Note that max-width is **not supported in IE**, but a good use of width: 100% would solve the problem neatly in an IE-specific style sheet. One more issue is that when an image is resized too small in some older browsers in Windows, the rendering is not as clear as it ought to be. There is a JavaScript to fix this issue, though.

While the above is a great quick fix and good start to responsive images, image resolution and download times should be the primary considerations. While resizing an image for mobile devices can be very simple, if the original image size is meant for large devices, it could significantly slow download times and take up space unnecessarily

Filament Group’s Responsive Images

This technique, presented by the Filament Group, takes this issue into consideration and not only resizes images proportionately, but shrinks image resolution on smaller devices, so very large images do not waste space unnecessarily on small screens



This technique requires a few files, all of which are available on Github. First, a JavaScript file (*rwd-images.js*), the *.htaccess* file and an image file (*rwd.gif*). Then, we can use just a bit of HTML to reference both the larger and smaller resolution images: first, the small image, with an *.r* prefix to clarify that it should be responsive, and then a reference to the bigger image using *data-fullsrc*

```

```

The *data-fullsrc* is a custom HTML5 attribute, defined in the files linked to above. For any screen that is wider than 480 pixels, the larger-resolution image (*largeRes.jpg*) will load; smaller screens would not need to load the bigger image, and so the smaller image (*smallRes.jpg*) will load.

The JavaScript file inserts a base element that allows the page to separate responsive images from others and redirects them as necessary. When the page loads, all files are rewritten to their original forms, and only the large or small images are loaded

as necessary. With other techniques, all higher-resolution images would have had to be downloaded, even if the larger versions would never be used. Particularly for websites with a lot of images, this technique can be a great saver of bandwidth and loading time.

This technique is fully supported in modern browsers, such as IE8+, Safari, Chrome and Opera, as well as mobile devices that use these same browsers (iPad, iPhone, etc.). Older browsers and Firefox degrade nicely and still resize as one would expect of a responsive image, except that both resolutions are downloaded together, so the end benefit of saving space with this technique is void.

Stop IPHONE Simulator Image Resizing

One nice thing about the iPhone and iPod Touch is that Web designs automatically rescale to fit the tiny screen. A full-sized design, unless specified otherwise, would just shrink proportionally for the tiny browser, with no need for scrolling or a mobile version. Then, the user could easily zoom in and out as necessary.

There was, however, one issue this simulator created. When responsive Web design took off, many noticed that images were still changing proportionally with the page even if they were specifically made for (or could otherwise fit) the tiny screen. This in turn scaled down text and other elements.



Because this works only with Apple's simulator, we can use an Apple-specific meta tag to fix the problem, placing it *below* the website's <head> section. Thanks to Think Vitamin's article on image resizing, we have the meta tag below:

```
<meta name="viewport" content="width=device-width; initial-scale=1.0">
```

Setting the initial-scale to 1 overrides the default to resize images proportionally, while leaving them as is if their width is the same as the device's width (in either portrait or landscape mode). Apple's documentation has a lot more information on the viewport meta tag.

Custom Layout Structure

For extreme size changes, we may want to change the layout altogether, either through a separate style sheet or, more efficiently, through a CSS media query. This does not have to be troublesome; most of the styles can remain the same, while specific style sheets can inherit these styles and move elements around with floats, widths, heights and so on.

We could have one main style sheet (which would also be the default) that would define all of the main structural elements, such as #wrapper, #content, #sidebar, #nav, along with colors, backgrounds and typography. Default flexible widths and floats could also be defined.



If a style sheet made the layout too narrow, short, wide or tall, we could then detect that and switch to a new style sheet. This new child style sheet would adopt everything from the .default style sheet and then just redefine the layout's structure

Here is the ***style.css*** (default) content:

/ Default styles that will carry to the child style sheet */*

```
html,body{
    background...
    font...
    color...
}
```

```
h1,h2,h3{
p, blockquote, pre, code, ol, ul{
```

/ Structural elements */*

```
#wrapper{
    width: 80%;
    margin: 0 auto;
```

```
background: #fff;
```



```
padding: 20px;  
}
```

```
#content{  
  width: 54%;  
  float: left;  
  margin-right: 3%;  
}
```

```
#sidebar-left{  
  width: 20%;  
  float: left;  
  margin-right: 3%;  
}
```

```
#sidebar-right{  
  width: 20%;  
  float: left;  
}
```

Here is the *mobile.css* (**child**) content:

```
#wrapper{  
  width: 90%;  
}
```

```
#content{  
  width: 100%;  
}
```

```
#sidebar-left{  
  width: 100%;  
  clear: both;
```



```
/* Additional styling for our new layout */
border-top: 1px solid #ccc;
margin-top: 20px;
}

#sidebar-right{
  width: 100%;
  clear: both;

  /* Additional styling for our new layout */
  border-top: 1px solid #ccc;
  margin-top: 20px;
}
```

1.2.2 Media Queries

CSS3 supports all of the same media types as CSS 2.1, such as screen, print and handheld, but has added dozens of new media features, including max-width, device-width, orientation and color. New devices made after the release of CSS3 (such as the iPad and Android devices) will definitely support media features. So, calling a media query using CSS3 features to target these devices would work just fine, and it will be ignored if accessed by an older computer browser that does not support CSS3.

We see an example of a media query in action:

```
<link rel="stylesheet" type="text/css"
      media="screen and (max-device-width: 480px)"
      href="shetland.css" />
```

This media query is fairly self-explanatory: if the browser displays this page on a screen (rather than print, etc.), and if the width of the screen (not necessarily the viewport) is 480 pixels or less, then load *shetland.css*.

New CSS3 features also include orientation (portrait vs. landscape), device-width, min-device-width and more. Look at “The Orientation Media Query” for more information on setting and restricting widths based on these media query features.

One can create multiple style sheets, as well as basic layout alterations defined to fit ranges of widths — even for landscape vs. portrait orientations.

Multiple media queries can also be dropped right into a single style sheet, which is the most efficient option when used:

```

/* Smartphones (portrait and landscape) ----- */
@media only screen
and (min-device-width : 320px)
and (max-device-width : 480px) {
/* Styles */
}

/* Smartphones (landscape) ----- */
@media only screen
and (min-width : 321px) {
/* Styles */
}

/* Smartphones (portrait) ----- */
@media only screen
and (max-width : 320px) {
/* Styles */
}

```

The code above is from a free template for multiple media queries between popular devices by Andy Clark.

CSS3 Media Queries

Above are a few examples of how media queries, both from CSS 2.1 and CSS3 could work. Let's now look at some specific how-to's for using CSS3 media queries to create responsive Web designs. Many of these uses are relevant today, and all will definitely be usable in the near future. The min-width and max-width properties do exactly what they suggest. The min-width property sets a minimum browser or screen width that a certain set of styles (or separate style sheet) would apply to. If anything is below this limit, the style sheet link or styles will be ignored. The max-width property does just the opposite. Anything above the maximum browser or screen width specified would not apply to the respective media query. Note in the examples below that we are using the syntax for media queries that could be used all in one style sheet. As mentioned above, the most efficient way to use media queries is to place them all in one CSS style sheet, with the rest of the styles for the website. This way, multiple requests do not have to be made for multiple style sheets.

```
@media screen and (min-width: 600px) {  
  .hereIsMyClass {  
    width: 30%;  
    float: right;  
  }  
}
```

The class specified in the media query above (hereIsMyClass) will work only if the browser or screen width is above 600 pixels. In other words, this media query will run only if the **minimum width is 600 pixels** (therefore, 600 pixels or wider).

```
@media screen and (max-width: 600px) {  
  .aClassforSmallScreens {  
    clear: both;  
    font-size: 1.3em;  
  }  
}
```

Now, with the use of max-width, this **media query** will apply only to browser or screen widths with a maximum width of 600 pixels or narrower.

While the above min-width and max-width can apply to either screen size or browser width, sometimes we'd like a media query that is relevant to device width specifically. This means that even if a browser or other viewing area is minimized to something smaller, the media query would still apply to the size of the actual device. The **min-device-width** and **max-device-width** media query properties are great for targeting certain devices with set dimensions, without applying the same styles to other screen sizes in a browser that mimics the device's size.

```
@media screen and (max-device-width: 480px) {  
  .classForiPhoneDisplay {  
    font-size: 1.2em;  
  }  
}
```

Keyword

Media queries

are a popular technique for delivering a tailored style sheet (responsive web design) to desktops, laptops, tablets, and mobile phones.

```
@media screen and (min-device-width: 768px) {
  .minimumiPadWidth {
    clear: both;
    margin-bottom: 2px solid #ccc;
  }
}
```

For the iPad specifically, there is also a media query property called **orientation**. The value can be either landscape (horizontal orientation) or portrait (vertical orientation).

```
@media screen and (orientation: landscape) {
  .iPadLandscape {
    width: 30%;
    float: right;
  }
}
@media screen and (orientation: portrait) {
  .iPadPortrait {
    clear: both;
  }
}
```

Unfortunately, this property works only on the iPad. When determining the orientation for the iPhone and other devices, the use of max-device-width and min-device-width should do the trick.

There are also many media queries that **make sense when combined**. For example, the min-width and max-width media queries are combined all the time to set a style specific to a certain range.

```
@media screen and (min-width: 800px) and (max-width: 1200px) {
  .classForaMediumScreen {
    background: #cc0000;
    width: 30%;
    float: right;
  }
}
```

The above code in this media query applies only to screen and browser widths between 800 and 1200 pixels. A good use of this technique is to show certain content or entire sidebars in a layout depending on how much horizontal space is available.

Some designers would also prefer to **link to a separate style sheet** for certain media queries, which is perfectly fine if the organizational benefits outweigh the efficiency lost. For devices that do not switch orientation or for screens whose browser width cannot be changed manually, using a separate style sheet should be fine.

You might want, for example, to place media queries all in one style sheet (as above) for devices like the iPad. Because such a device can switch from portrait to landscape in an instant, if these two media queries were placed in separate style sheets, the website would have to call each style sheet file every time the user switched orientations. Placing a media query for both the horizontal and vertical orientations of the iPad in the same style sheet file would be far more efficient.

Another example is a flexible design meant for a standard computer screen with a resizable browser. If the browser can be manually resized, placing all variable media queries in one style sheet would be best.

Nevertheless, organization can be key, and a designer may wish to define media queries in a standard HTML link tag:

```
<link rel="stylesheet" media="screen and (max-width: 600px)" href="small.css" />
<link rel="stylesheet" media="screen and (min-width: 600px)" href="large.css" />
<link rel="stylesheet" media="print" href="print.css" />
```

1.2.3 Javascript

Another method that can be used is JavaScript, especially as a back-up to devices that do not support all of the CSS3 media query options. Fortunately, there is already a pre-made JavaScript library that makes older browsers (IE 5+, Firefox 1+, Safari 2) support CSS3 media queries. If you are already using these queries, just grab a copy of the library, and include it in the mark-up: *css3-mediaqueries.js*.

In addition, below is a sample jQuery snippet that detects browser width and changes the style sheet accordingly — if one prefers a more hands-on approach:

```
<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.4.4/
jquery.min.js"></script>
```

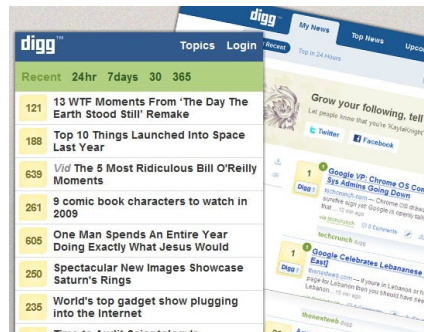
```
<script type="text/javascript">
$(document).ready(function(){
    $(window).bind("resize", resizeWindow);
    function resizeWindow(e){
        var newWindowWidth = $(window).width();
```

```
// If width width is below 600px, switch to the mobile stylesheet
if(newWindowWidth < 600){    $("link[rel=stylesheet]").attr({href : "mobile.
css"});    }    // Else if width is above 600px, switch to the large stylesheet    else
if(newWindowWidth > 600){
    $("link[rel=stylesheet]").attr({href : "style.css"});
}
}
});
</script>
```

There are many solutions for pairing up JavaScript with CSS media queries. Remember that media queries are not an absolute answer, but rather are fantastic options for responsive Web design when it comes to pure CSS-based solutions. With the addition of JavaScript, we can accomodate far more variations. For detailed information on using JavaScript to mimic or work with media queries, look at “Combining Media Queries and JavaScript.”

1.2.4 Showing or Hiding Content

It is possible to shrink things proportionally and rearrange elements as necessary to make everything fit (reasonably well) as a screen gets smaller. It's great that that's possible, but making every piece of content from a large screen available on a smaller screen or mobile device is not always the best answer. We have best practices for mobile environments: simpler navigation, more focused content, lists or rows instead of multiple columns.



Responsive Web design should not be just about how to create a flexible layout on a wide range of platforms and screen sizes. It should also be about the user being able to pick and choose content. Fortunately, CSS has been allowing us to show and hide content with ease for years!

display: none;

Either declare `display: none` for the HTML block element that needs to be hidden in a specific style sheet or detect the browser width and do it through JavaScript. In addition to hiding content on smaller screens, we can also hide content in our default style sheet (for bigger screens) that should be available only in mobile versions or on smaller devices. For example, as we hide major pieces of content, we could replace them with navigation to that content, or with a different navigation structure altogether.

Note that we have not used `visibility: hidden` here; this just hides the content (although it is still there), whereas the `display` property gets rid of it altogether. For smaller devices, there is no need to keep the mark-up on the page — it just takes up resources and might even cause unnecessary scrolling or break the layout.

Here is **our mark-up**:

```
<p class="sidebar-nav"><a href="#">Left Sidebar Content</a> | <a href="#">Right  
Sidebar Content</a></p>
```

```
<div id="content">  
  <h2>Main Content</h2>  
</div>
```

```
<div id="sidebar-left">  
  <h2>A Left Sidebar</h2>  
  
</div>
```

```
<div id="sidebar-right">  
  <h2>A Right Sidebar</h2>  
</div>
```

In our default style sheet below, we have hidden the links to the sidebar content. Because our screen is large enough, we can display this content on page load.

Here is the ***style.css* (default)** content:

```
#content{  
  width: 54%;  
  float: left;  
  margin-right: 3%;  
}
```

```
#sidebar-left{
  width: 20%;
  float: left;
  margin-right: 3%;
}
#sidebar-right{
  width: 20%;
  float: left;
}
.sidebar-nav{display: none;}
```

Now, we hide the two sidebars (below) and show the links to these pieces of content. As an alternative, the links could call to JavaScript to just cancel out the `display: none` when clicked, and the sidebars could be realigned in the CSS to float below the content (or in another reasonable way).

Here is the *mobile.css* (**simpler**) content:

```
#content{
  width: 100%;
}

#sidebar-left{
  display: none;
}

#sidebar-right{
  display: none;
}

.sidebar-nav{display: inline;}
```

With the ability to easily show and hide content, rearrange layout elements and automatically resize images, form elements and more, a design can be transformed to fit a huge variety of screen sizes and device types. As the screen gets smaller, rearrange elements to fit mobile guidelines; for example, use a script or alternate style sheet to increase white space or to replace image navigation sources on mobile devices for better usability (icons would be more beneficial on smaller screens).

1.2.5 Touchscreens vs. Cursors

Touchscreens are becoming increasingly popular. Assuming that smaller devices are more likely to be given touchscreen functionality is easy, but don't be so quick. Right now touchscreens are mainly on smaller devices, but many laptops and desktops on the market also have touchscreen capability. For example, the HP Touchsmart tm2t is a basic touchscreen laptop with traditional keyboard and mouse that can transform into a tablet.



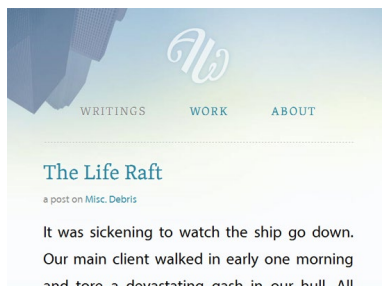
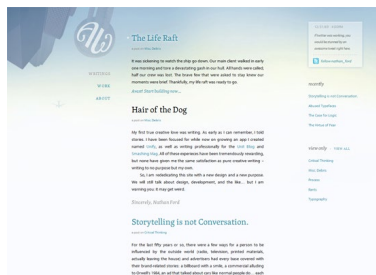
Touchscreens obviously come with different design guidelines than purely cursor-based interaction, and the two have different capabilities as well. Fortunately, making a design work for both does not take a lot of effort. Touchscreens have no capability to display CSS hovers because there is no cursor; once the user touches the screen, they click. So, don't rely on CSS hovers for link definition; they should be considered an additional feature only for cursor-based devices.

Many of the design suggestions in it are best for touchscreens, but they would not necessarily impair cursor-based usability either. *For example*, sub-navigation on the right side of the page would be more user-friendly for touchscreen users, because most people are right-handed; they would therefore not bump or brush the navigation accidentally when holding the device in their left hand. This would make no difference to cursor users, so we might as well follow the touchscreen design guideline in this instance. Many more guidelines of this kind can be drawn from touchscreen-based usability.

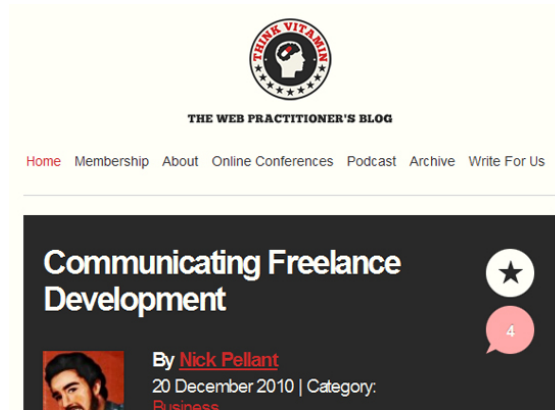
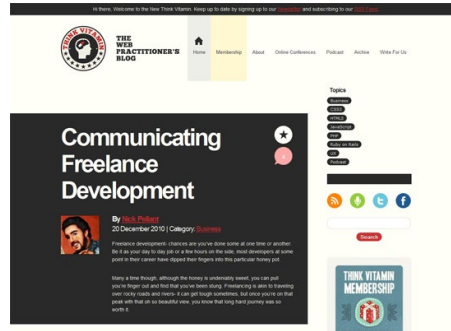
1.2.6 A Showcase of Responsive Web Design

Below we have a few examples of responsive Web design in practice today. For many of these websites, there is more variation in structure and style than is shown in the pairs of screenshots provided. Many have several solutions for a variety of browsers, and some even adjust elements dynamically in size without the need for specific browser dimensions. Visit each of these, and adjust your browser size or change devices to see them in action.

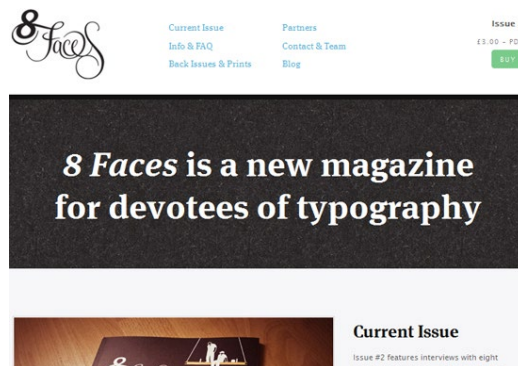
Art Equals Work is a simple yet great example of responsive Web design. The first screenshot below is the view from a standard computer screen dimension. The website is flexible with browser widths by traditional standards, but once the browser gets too narrow or is otherwise switched to a device with a smaller screen, then the layout switches to a more readable and user-friendly format. The sidebar disappears, navigation goes to the top, and text is enlarged for easy and simple vertical reading.



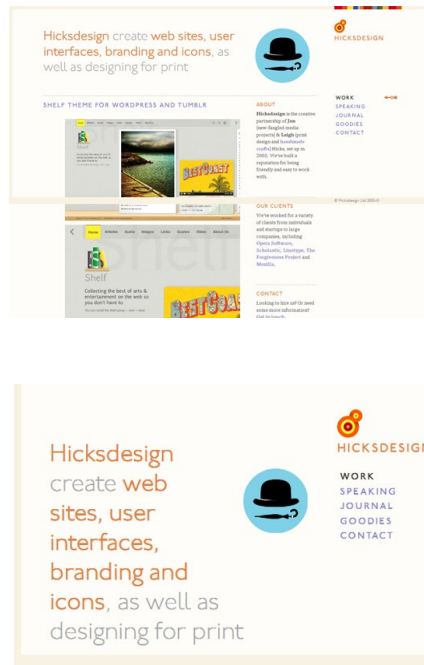
With Think Vitamin, we see a similar approach. When on a smaller screen or browser, the sidebar and top bar are removed, the navigation simplifies and moves directly above the content, as does the logo. The logo keeps its general look yet is modified for a more vertical orientation, with the tagline below the main icon. The white space around the content on larger screens is also more spacious and interesting, whereas it is simplified for practical purposes on smaller screens.



8 Faces' website design is flexible, right down to a standard netbook or tablet device, and expands in content quantity and layout width when viewed on wider screens or expanded browsers. When viewed on narrower screens, the featured issue on the right is cut out, and the content below is shortened and rearranged in layout, leaving only the essential information.



The Hicksdesign website has three columns when viewed on a conventional computer screen with a maximized browser. When minimized in width, the design takes on a new layout: the third column to the right is rearranged above the second, and the logo moves next to the introductory text. Thus, no content needs to be removed for the smaller size. For even narrower screens and browser widths, the side content is removed completely and a simplified version is moved up top. Finally, the font size changes with the screen and browser width; as the browser gets narrower, the font size throughout gets smaller and remains proportional.



Here is a great example of a flexible image. The image in this design automatically resizes after certain “break” points, but in between those width changes, only the side margins and excess white space are altered. On smaller screens and minimized browsers, the navigation simplifies and the columns of navigation at the top fall off. At the design’s smallest version, the navigation simplifies to just a drop-down menu, perfect for saving space without sacrificing critical navigation links.



The website for Garret Keizer is fully flexible in wider browsers and on larger screens: the photo, logo and other images resize proportionally, as do the headings and block areas for text. At a few points, some pieces of text change in font size and get smaller as the screen or browser gets narrower. After a certain break point, the layout transforms into what we see in the second screenshot below, with a simple logo, introductory text and a simple vertical structure for the remaining content.

With four relatively content-heavy columns, it's easy to see how the content here could easily be squished when viewed on smaller devices. Because of the easy organized **columns**, though, we can also collapse them quite simply when needed, and we can stack them vertically when the space does not allow for a reasonable horizontal span.



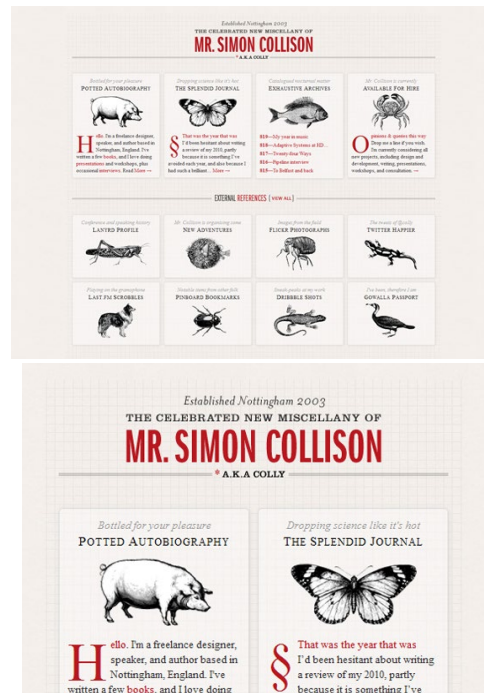


When the browser is minimized or the user is on a smaller device, the columns first collapse into two and then into one. Likewise, the horizontal lines for break points also change in width, without changing the size or style of each line's title text.

Keyword

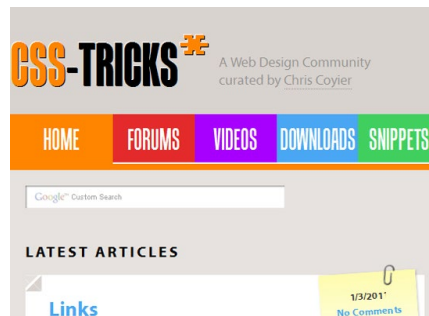
Column in

architecture and structural engineering is a structural element that transmits, through compression, the weight of the structure above to other structural elements.

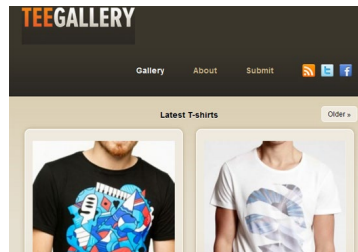
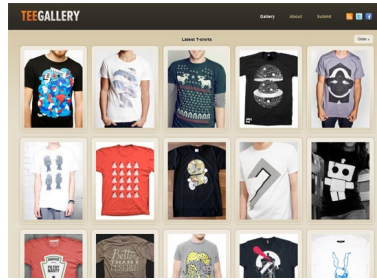


On the CSS Tricks website, like many other collapsible Web designs, the sidebars with excess content are the first to

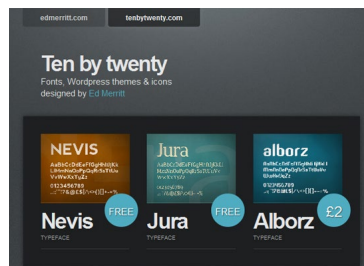
fall off when the screen or browser gets too narrow. On this particular website, the middle column or first sidebar to the left was the first to disappear; and the sidebar with the ads and website extras did the same when the browser got even narrower. Eventually, the design leaves the posts, uses less white space around the navigation and logo and moves the search bar to below the navigation. The remaining layout and design is as flexible as can be because of its simplicity.



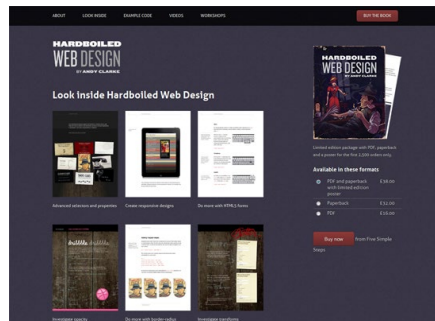
As one can see, the main navigation here is the simple layout of t-shirt designs, spanning both vertically and horizontally across the screen. As the browser or screen gets smaller, the columns collapse and move below. This happens at each break point when the layout is stressed, but in between the break points, the images just change proportionally in size. This maintains balance in the design, while ensuring that any images (which are essential to the website) don't get so small that they become unusable.



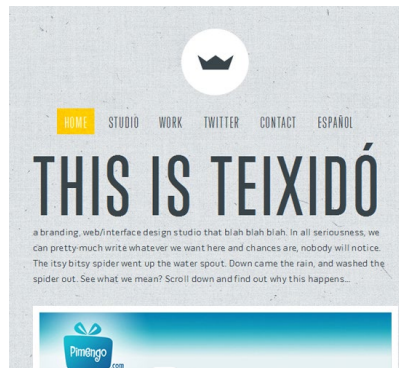
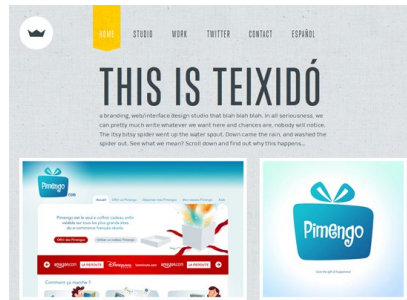
Ten by Twenty is another design that does not resort to changing layout structure at all after certain break points, but rather simplifies responsive Web design by making everything fully flexible and automatically resizing, no matter what the screen or browser width. After a while, the design does stress a bit and could benefit from some rearrangement of content. But overall, the image resizing and flexible content spaces allow for a fairly simple solution that accommodates a wide range of screen sizes.



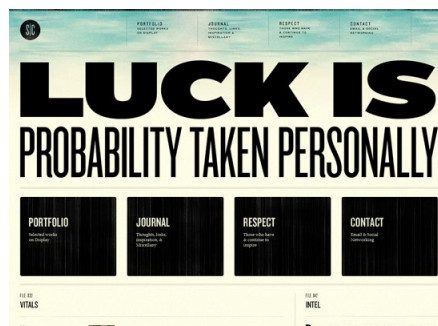
On wide screens and browsers, all of the content on this simply designed website is well organized into columns, sidebar and simple navigation up top. It's a fairly standard and efficient layout. On smaller screens, the sidebar is the first to drop off, and its content is moved below the book previews and essential information. Being limited in space, this design preserves its important hierarchy. Whereas on a wider screen we'd look left to right, on a narrower screen we'd tend to look from top to bottom. Content on the right is moved below content that would appear on the left on a wider screen. Eventually, when the horizontal space is fully limited, the navigation is simplified and stacked vertically, and some repeated or inessential elements are removed.



This design features a complex layout that looks inspired by a print style. When viewed on a standard wide computer screen, more portfolio pieces are featured and spanned horizontally across the page. As one moves down the page, more graphics and imagery span the space. On a smaller screen, the portfolio piece is cut down to one, and then eventually left out altogether for very small screens and narrow browsers. The visualizations below collapse into fewer columns and more rows, and again, some drop off entirely for very small screens. This design shows a creative and intelligent way to make a not-so-common layout work responsively.

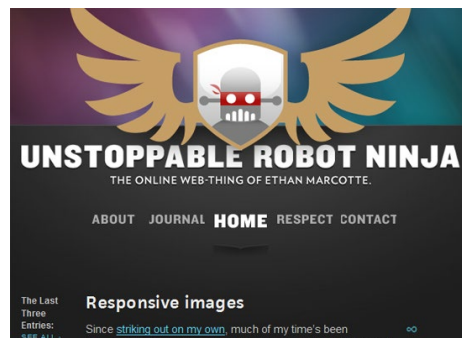
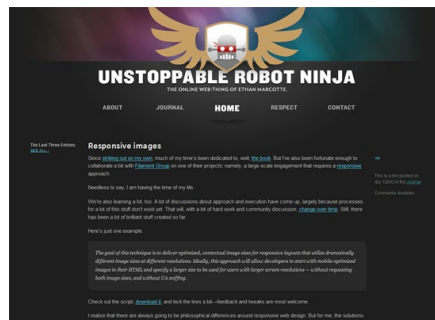


This design has three main stages at which the design and layout collapse into a more user-friendly form, depending on how wide the screen or browser is. The main image (featuring type) is scaled proportionally via a flexible image method. Each “layout structure” is fully flexible until it reaches a breaking point, at which point the layout switches to something more usable with less horizontal space. The bottom four columns eventually collapse into two, the logo moves above the navigation, and the columns of navigation below are moved on top or below each other. At the design’s narrowest stage, the navigation is super-simplified, and some inessential content is cut out altogether.



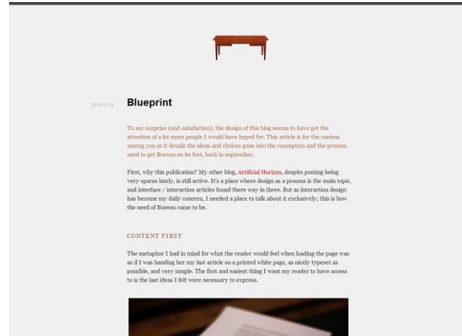


This layout does not change at all; no content is dropped or rearranged; and the text size does not change either. Instead, this design keeps its original form, no matter what the change in horizontal and vertical space. Instead, it automatically resizes the header image and the images for the navigation. The white space, margins and padding are also flexible, giving more room as the design expands and shrinks.



This is perhaps the simplest example of a responsive Web design in this showcase, but also one of the most versatile. The only piece in the layout that changes with the browser width is the blog post's date, which moves above the post's title or to the side,

depending on how much horizontal space is available. Beyond this, the only thing that changes is the width of the content area and the margin space on the left and right. Everything is centered, so a sense of balance is maintained whatever the screen or browser width. Because of this design's simplicity, switching between browser and screen widths is quick and easy.



Keyword

Displays is the international journal covering the research and development of display technology, its effective presentation and perception of information, and applications and systems including display-human interface.

Harry Roberts shows that responsive design can also have quite humble uses. If the user has a large viewport, the website **displays** three columns with a navigation menu floating on the left. For users with a viewport between 481px and 800px, a narrow version is displayed: the navigation jumps to the top of the site leaving the area for the content column and the sidebar. Finally, the iPhone view displays the sidebar under the content area. *For example* of how a couple of simple CSS adjustments can improve the website's appearance across various devices.

This last design by Bryan James shows that responsive Web design need not apply only to static HTML and CSS websites. Done in Flash, this one features a full-sized background image and is flexible up to a certain width and height. As a result of the design style, on screens that are too small, the background image gets mostly hidden and the content can become illegible and squished. Instead of just letting it be, though, a message pops up informing the user that the screen is too small to adequately view the website. It then prompts the user to switch to a bigger screen. One can discuss if the design solution is good or bad in terms of usability, but the example shows that Flash websites can respond to user's viewport, too.

1.2.7 Responsive Design Techniques, Tools and Design Strategies

It is of significant aspect to design a responsive website that would help you stand apart from the rest and stay way ahead of your technology world. With the multiplication of devices used to access the internet (computer, tablets, smartphones...) you have to make sure that your website will look good on every device possible.



The creative website however, is not just enough; a responsive website should assist you in building your brand as well as business strategy. A responsive web design is composed of three distinct parts:

- A flexible, grid-based layout
- Flexible images and media, and
- Media queries, a module from the CSS3 specification.
- Responsive Web Design Strategy

Thanks to the unbelievable rise in the number of mobile devices, responsive web designing is no longer a UX-only discipline. All marketers need to consider responsive web designing as a component of their strategic planning. It might not be necessarily a content marketer for you to know all the tricks of the designing and development of a responsive web. However, you must know what exactly RWD is. Also, you should know your exact content marketing responsibilities.

There are listed some Responsive Web Design Tools and Techniques bellow:

Remember

Web design must be adapted. The layout should be adjusted automatically to fit any screen resolution and peripherals.

Responsive Design with CSS3 Media Queries

This is responsive for screen resolution ranges 320px (iPhone) to 2560px (widescreen) or more. Users are no longer just browse the Web with desktops. Users can now use mobiles, laptops, small tablet devices like the iPad or Playbook to access the .Web. Therefore, the traditional fixed width does not work

CSS3 Generator Tools Responsive Web Design

CSS3 Generator tools responsive web design, which allows you to generate extracts for Text Shadow, CSS3 Box Shadow, .Transitions, Selector, Transform, Border Radius and More

Useful CSS Tricks for Responsive Design

Here are usually used CSS tips and case examples for encoding sensitive designs. They are simple as CSS properties min-width, max-width, overflow, and the relative value — but these .properties are important in designing reactive

Online Sprite Box Tool Responsive Web Design

Spritebox is a WYSIWYG tool to help responsive web designers quickly and easily create CSS classes and IDs from a single sprite image. It's based on the principle of using the background-position property to align areas of a sprite image into block elements of a web page. It was made using a combination of JQuery, CSS3 and HTML5, and is completely free to use

Pattern Generator Tools Responsive Web Design

Patternizer is an easy to use the pattern generator for scratching. It's great fun and free! This flow line tool will be very useful for web designers when it comes to creating overlays and .header or backgrounds

Multi-Device Layout Patterns Responsive Web Design

With grids of fluids and adjustments media query, proper design allows Web layouts to suit a variety of screen sizes. More and more designers use this technique; we do not just see .a lot of innovation, but the emergence of clear trends as well

Photoshop Grid for Responsive Web Design

In making the transition to web design sensitive, one of the potential barriers are rather difficult mathematics to calculate percentage based widths needed to put in the fluid. If, for example, you design a 960px grid in Photoshop and you have six columns, each 140px wide, you divide 140 by 960 to get your percentage based on the width: 14.583333%

Style Tiles Responsive Web Design Examples

Style tiles are made of design available fonts, colors and interface elements that communicate the essence of a visual brand for the Web. They help form a common visual language between designers and stakeholders and provide a catalyst for discussions about preferences and client objectives

Resize My Browser Tools Responsive Web Design

A simple application browser, useful and beautiful resizes the window for designers and **Web developers**. If you require your browser to display content in a window size sure this is what you were looking for. Just click on the size you need and check what your size is similar. It does not work in Chrome and Opera because of problems with the “resizeTo” event

Adobe Kuler—Responsive Web Design Color

Adobe kuler tools for generating color themes that can inspire any project. No matter what you are creating, with Kuler you can experiment quickly with color variations and browse thousands of themes from the Kuler

1.2.8 Design Process in the Responsive Age

When we began the web design process the other day with a client, we stumbled across a new phrase in the RWD world: Age Responsive Design.

We already have the ability to make sites adapt based on screen size. But this is only the first step. As technology has evolved we have developed the ability to adapt web structure and content based on a user’s age.

We have the technology

While we might not be making a Six Million Dollar Man, Age Responsive Design is not an unattainable concept imagined in some science-fiction realm. In fact, the gaming industry has been using a player’s age and experience to adapt First Person

Perspective (FPP) games by using dynamic game balancing (DGB) for years! DGB is the automatic process that changes parameters, scenarios, and behaviors in a video game in real-time, based on the player's ability. This is utilized to avoid making a player bored or frustrated. If you are a beginner, you will only fight enemies that have reduced speed and health. But if you are a pro, the game will become much more difficult and increase the gameplay duration.

The concept is similar to what the age-responsive design will do to websites. And, just as we have the information to determine a user's age, this data can be used to combine those existing technologies to create universally user-friendly experiences. As we enter 2017, websites will be able to combine this information to make subtle changes to accommodate younger or older audiences.

.One size DOES NOT fit all

A 7-year-old and a 70-year-old do not read the same books or wear the same clothes, so why should they be subjected to the same online experience?

Age Responsive Design is made with the ability to focus on how the site looks; but also how it works, what type of content it displays, and what a user can do with it. Just a few examples of what Age Responsive Design could do, include:

- bigger vs smaller font sizes and spacing
- muted vs saturated color schemes
- video presentations vs text explanations
- social media sharing vs email conversion
- expanded vs condensed navigation menus
- typographic vs traditional content

While *automatic* age design is not currently in use, targeted web content is... So the application of Age Responsive Design is just around the corner - it's only matter of who implements it first.

ROLE MODEL

VINT CERF IS AN AMERICAN ENGINEER WHO REVOLUTIONIZED THE WORLD OF TECHNOLOGY AND IS OFTEN REGARDED AS THE 'FATHER OF INTERNET'



Vinton Gray Cerf

Born June 23, 1943 is an American Internet pioneer, who is recognized as one of “the fathers of the Internet”, sharing this title with TCP/IP co-inventor Bob Kahn. His contributions have been acknowledged and lauded, repeatedly, with honorary degrees and awards that include the National Medal of Technology, the Turing Award, the Presidential Medal of Freedom, the Marconi Prize and membership in the National Academy of Engineering.

Vint Cerf is one of the most significant scientists, of contemporary times, who has been instrumental in designing and developing the TCP/IP protocols along with engineer Bob Kahn. After completing his schooling he took up Mathematics as a subject and earned a bachelor's degree in it. After this he got a job as systems engineer and then directed his attention to computer science and premeditated on the subject in his post-graduation. He trained under the tutelage of some eminent scientist and himself did some important research during his university days. His encounter with electrical engineer Robert E. (Bob) Kahn presented a new world of possibilities before him and these two scientists together made significant contribution to the world of science. They worked day in and day out and their hard work and patience paid when they designed the basis of internet, that is, the TCP/IP protocols. Cerf worked with many scientific associations and helped them develop and enhance their efficiency. This visionary scientist even outlined the future of technology after he became the Chief Internet Evangelist of Google. He and Kahn made some of the most important advancements revolving around internet, and it is because of these two people that the world is now considered a global village.

Childhood and Early Life

- He was born on 23rd June, 1943, in New Haven, Connecticut, US, to Vinton Thurston Cerf and Muriel (née Gray). His father was employed in an aeronautics company while his mother was a home-maker.
- He received education at the 'Van Nuys High School', where future scientists Jon Postel and Steve Crocker were his classmates.
- He completed his bachelor's degree in 1965, from 'Stanford University', California, with mathematics as his subject. Thereafter, he got employed at 'IBM' (International Business Machines Corporation) in the position of a systems engineer.
- During the same time, he also pursued his master's degree in computer science from the 'University of California', Los Angeles (UCLA) which he completed in the year 1970. During his post-graduation, he got acquainted to engineer Robert E. (Bob) Kahn who had already been engaged with the 'ARPANet' network.

Career

- In 1972, he earned a doctorate in computer science from the 'Stanford University'. Thereafter, he started working at the university as an assistant professor. During the same time, he also did research work on packet network along with Kahn and constructed the DoD TCP/IP protocol suite.
- He joined the 'Defense Advanced Research Projects Agency' (DARPA) in 1976, and continued to conduct researches there for the next six years.
- In 1982, he was appointed the vice president of 'MCI Digital Information Services', where he headed the operations of the 'MCI Mail', which was a commercial email service. He served in the position for the next four years.
- The two engineers, Cerf and Bob Kahn, together initiated the 'Internet Society' in 1992. This society is a charitable organization formed in order to deliver internet related education and avail the use of the facility for the common man.
- In 1994, he joined the 'MCI Communication' once again and was appointed as the Senior Vice President of Technology Strategy. In this capacity, he led the corporate strategy with a technical approach.
- He collaborated with the 'Gallaudet University' in 1997 by becoming its trustee as well as a member of its 'Board of Associates'. The university is dedicated to the education of people with hearing impairments.

In 1999, he became a member at the 'Internet Corporation for Assigned Names and Numbers'. The ICANN is a non-profit organization which deals with internet security and stability and focuses on Internet protocol numbers and Domain Name System (DNS) root.



- In October 2005, he was appointed as the Vice President of the multinational technology company 'Google'. He was also the Chief Internet Evangelist of the company, and prophesied the future of internet and its effect on the society.
- This eminent engineer has also been associated with the IT Advisory Council of President of Bulgaria, Georgi Parvanov. He has even been associated with the political risk consultancy named 'Eurasia Group'.
- In 2008, he worked with the 'IETF' (Internet Engineering Task Force) as the chairperson of the 'Internationalized domain name' (IDNAbis).
- He is engaged with research work on the space computer network called 'Interplanetary Internet' in association with 'Jet Propulsion Laboratory' of 'NASA'.
- This engineering genius has been a member of the International Advisory Board of the 'International Multilateral Partnership Against Cyber Threats' (IMPACT).
- At present he is associated with the organization 'Scientists and Engineers for America' as a member of the board of advisors. He is also a member of the advisory council of the non-profit organization 'CRDF Global' (Civilian Research and Development Foundation), and 'The Liquid Information Company Ltd', UK.
- He is also a trustee of the 'ARIN' (American Registry for Internet Numbers) and one of the directors of the non-profit organization 'StopBadware' which works to provide Web safety. He, along with Tim Berners-Lee and Al Gore chairs the 'Campus Party Silicon Valley'.
- This accomplished engineer has been a member of the Governing Board of the 'Smart Grid Interoperability Panel' (SGIP) from 2009 to 2011.
- In 2012, he was chosen as the President of the 'Association for Computing Machinery' (ACM) for a two-year term.

Major Works

- This brilliant engineer has revolutionized the world of technology by developing the internet and taking it to advanced levels. He is credited with designing the TCP/IP protocols along with engineer Bob Kahn.

Awards and Achievements

- In 1996, he was awarded with the 'Yuri Rubinsky Memorial Award'. The same year, 'The Franklin Institute' honoured him with the 'Certificate of Merit'. Two years later, he became a Fellow of the 'IEEE' (Institute of Electrical and Electronics Engineers).

- In 1997, the then President of the US Bill Clinton presented Cerf and Kahn with the 'National Medal of Technology'.
- In 2000, he became a Fellow of the 'Computer History Museum' and the same year, the 'Library of Congress' awarded him with the 'Living Legend Medal'.
- He was awarded with the 'A.M.Turing Award' in 2004. He shared this award with Bob Kahn, for their significant contribution towards the development of internet.
- In 2005, he along and Kahn were together awarded with the 'Presidential Medal of Freedom' by the then president of the US George W. Bush. He has also received the 'SIGCOMM Award' for his contributions to the development of the internet.
- In 2006, he was inducted in the 'National Inventors Hall of Fame' along with fellow scientist Bob Kahn.
- In 2012, he was inducted into the 'Internet Hall of Fame', and the following year, he was honoured with the 'Queen Elizabeth Prize for Engineering' along with four other Internet and Web innovators.
- During 2013-14, he was a recipient of the 'Bernard Price Memorial Lecture', and also honoured with the 'Order of the Cross of Terra Mariana' and 'Officer of the French Légion d'honneur'.

Personal Life and Legacy

- In 1966, Cerf married Sigrid, and is blessed with two children David and Bennett. The family resides in Virginia.

He has been awarded with several honorary doctorate degrees from various institutions like 'Yale University', 'University of the Balearic Islands', 'University of the Balearic Islands', 'University of Pisa', 'University of South Australia' and 'Keio University'.

SUMMARY

- Responsive web design is a design that adjusts and adapts to any viewport (user's visible area of the web page) rather than only a specific viewport, irrespective of the visualization device.
- Responsive web design is an example of user interface plasticity. The Internet took off quicker than anyone would have predicted, growing like crazy.
- Responsive Web Design is about using CSS and HTML to resize, hide, shrink, enlarge, or move the content to make it look good on any screen.
- Responsive Web design requires a more abstract way of thinking. However, some ideas are already being practiced: fluid layouts, media queries and scripts that can reformat Web pages and mark-up effortlessly (or automatically).
- Flexible designs were not really that flexible; they could give or take a few hundred pixels, but they often could not adjust from a large computer screen to a netbook.

KNOWLEDGE CHECK

1. **What is a CMS in web design?**
 - a. Content Management System
 - b. Creative Management System
 - c. Content Mixing System
 - d. Creatives Managerial System
2. **To make your website mobile friendly, you can make your website**
 - a. Responsive
 - b. Reactive
 - c. Fast Loading
 - d. Light
3. **What does CSS stand for?**
 - a. Current Style Sheets
 - b. Current Sheets Style
 - c. Cascading Style Sheets
 - d. Cascading Sheets Style
4. **Which of the following statements is false**
 - a. You can make a website without using HTML
 - b. You can make a website without using PHP
 - c. You can make a website without using CSS
 - d. You can make a website without using Javascript
5. **What is Word-Press?**
 - a. It is a software used to press text
 - b. It is a text formatting software
 - c. It is a CMS (Content Management System)
 - d. It is mail service
6. **Which of the following is a server-side solution for implementing responsive images?**
 - a. <http://adaptive-images.com>
 - b. <http://github.com/scottjehl/picturefill>
 - c. <http://www.netmagazine.com>
 - d. <http://offroadcode.com>

7. Which of the following is not included in the framework?
 - a. font sizes
 - b. css reset
 - c. button styles
 - d. forms
8. Which is not a tool to build responsive websites?
 - a. skeleton
 - b. bookmarklets
 - c. code editors
 - d. web browsers
9. Which is not bookmarklet?
 - a. Screen Fly
 - b. Resizer
 - c. Screenqueri.es
 - d. Brackets
10. Adjacent sibling selector is defined with the notation _____
 - a. #
 - b. \$
 - c. %
 - d. +

REVIEW QUESTIONS

1. Discuss the importance of responsive web designing.
2. Explain the adjusting screen resolution.
3. What do you understand by media queries and JavaScript?
4. Focus on showcase of responsive web design.
5. Discuss the responsive design techniques, tools and design strategies.

Check Your Result

- | | | | | |
|--------|--------|--------|--------|---------|
| 1. (a) | 2. (a) | 3. (c) | 4. (a) | 5. (c) |
| 6. (a) | 7. (d) | 8. (a) | 9. (d) | 10. (d) |

REFERENCES

1. Baker, S. (2014). Making it work for everyone: HTML5 and CSS level 3 for responsive, accessible design on your library's web site.
2. Campbell, J. (2014). Responsive website design ideas and cross browser compatibility. Retrieved October 9, 2017, from <http://www.designhill.com/design-blog/responsive-website-design-andcross-browser-compatibility/>
3. Cao, J. (2015). Responsive vs. adaptive design: What's the best choice for designers? Retrieved October 10, 2017, from <https://www.uxpin.com/studio/blog/responsive-vs-adaptive-designwhats-best-choice-designers/>
4. Cao, J. (2017). Important considerations for responsive design performance & UX. Retrieved October 20, 2017, from <https://www.uxpin.com/studio/blog/important-considerations-responsive-designperformance-ux/>
5. Choudhary, S. (2011). Detecting cross-browser issues in web applications. In Proceedings of the International Conference on Software Engineering (ICSE), Honolulu, Hawaii, 1146-1148.
6. Costill, A. (2014). Site speed vs. responsive design: Which is more important? Retrieved October 9, 2017, from <https://www.searchenginejournal.com/site-speed-vs-responsive-design-important/113112/>
7. Hansen, K. (2016). Designing responsive environments through user experience research.
8. Lacaden, T. (2017). Responsive design provides the best user experience. Retrieved October 9, 2017, from <https://www.activewebgroup.com/blog/responsive-design-provides-the-best-user-experience>
9. Rice, K. (2015). Responsive design is failing mobile UX. Retrieved October 10, 2017, from <https://www.webdesignerdepot.com/2015/06/responsive-design-is-failing-mobile-ux/>
10. Soegaard, M. (2017). Adaptive vs. Responsive Design. Retrieved October 27, 2017, from <https://www.interaction-design.org/literature/article/adaptive-vs-responsive-design>

CHAPTER 2

RAPID PROTOTYPING

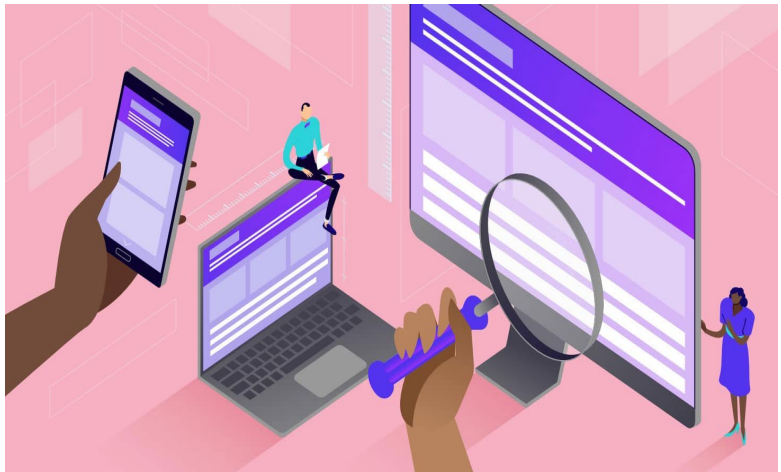
"If you are truly innovating, you don't have a prototype you can refer to."

–Jonathan Ive

LEARNING OBJECTIVES

After studying this chapter,
you will be able to:

1. Discuss the concept of rapid prototyping
2. Explain the types of rapid prototyping



INTRODUCTION

Rapid prototyping is the fast fabrication of a physical part, model or assembly using 3D computer aided design

(CAD). The creation of the part, model or assembly is usually completed using additive manufacturing, or more commonly known as 3D printing.

Rapid prototype or rapid prototyping is a relatively new term and in its simplest form, the process of creating prototypes quickly to visually and functionally evaluate an engineering product design.

Prototypes are an integral part of engineering product design and more importantly in an overall new product development process. Rapid prototyping can be used at any stage of the product development cycle or for any component or sub-component and can be repeated numerous times along the new product design process.

Although the term prototype is used in other contexts such as software programming, semantics, and application development etc., the purpose is the same.

The rapid prototype meaning is the fast and agile fabrication of a model, physical part, or assembly with 3D computer-aided design, also referred to as CAD. The creation of a model, part, or assembly is carried out with additive manufacturing, also commonly referred to as 3D printing. Rapid prototyping entails different manufacturing technologies, but most of them use layered additive manufacturing.

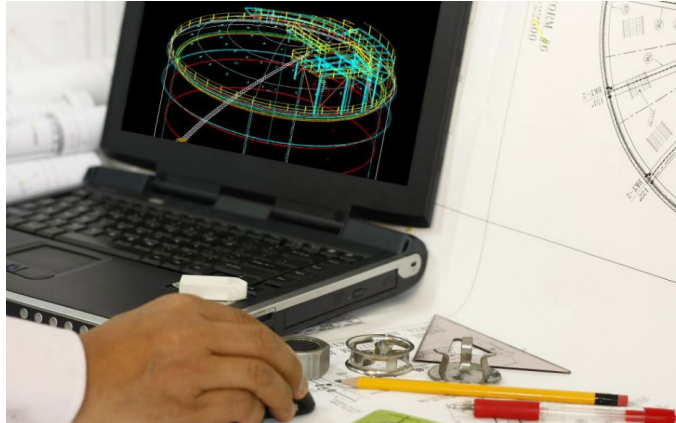
Many other technologies are also used for RP, like casting, high-speed machining, extruding, and molding. The most commonly used technology is additive manufacturing used in the RP process besides other conventional processes for creating prototypes.

2.1 CONCEPT OF RAPID PROTOTYPING

Rapid prototyping is a computer program that constructs three-dimensional models of work derived from a Computer Aided Design (CAD) drawing. It is used to quickly and easily turn product designs into physical samples. The creation of physical samples with this method is achieved through Adobe Portable Document Format (PDF) and CAD formats, as well as through cross-functional teams and integration.

The rapid prototyping method was first introduced to the market in 1987, after it was developed with the help of stereo lithography. Today, it is also known as solid freeform fabrication, 3-dimensional printing, freeform fabrication, and additive fabrication. The manufacturing process of rapid prototyping can produce automatic construction of physical models with 3-dimensional printers, stereo-lithography machines, and even laser sintering systems.

Using a CAD drawing to create a physical prototype is quite simple for the user. First, the machine reads the data from the provided CAD drawing. Next, the machine lays a combination of liquid or powdered material in successive layers. The materials used in rapid prototyping are usually plastics, ceramics, wood-like paper, or metals such as stainless steel and titanium.



With rapid prototyping, each layer is built to match the virtual cross section taken from the CAD model. Therefore, the final model is built up gradually with the help of these cross sections. Finally, the cross sections are either glued together or fused with a laser. The fusing of the model automatically creates its final shape.

Rapid prototyping is necessary for those who want to create models for clients, such as architects and engineers. It can reduce the design cycle time, allowing multiple tests to be performed on the design at a low cost. This is because each prototype can be completed within days or hours, rather than taking several weeks. In addition to engineers and architects, other professionals benefit from rapid prototyping, such as surgeons, artists, and archaeologists.

The Term Rapid Prototyping

The term rapid prototyping originally described only the methods for fast production of models, patterns or simple prototypes by using generative production methods. The initial point was a digital, three-dimensional design data. For the implementation of this technology, the development of a data interface was important. The data interface can give exact descriptions of the geometries of the object and can be used by rapid prototyping mechanisms. This was possible with the development of data format STL (STereoLithography). The STL interface was originally developed for the stereo-lithography process and has proved its worth as a data standard. Over time rapid prototyping also included the terms rapid tooling and rapid manufacturing. Rapid Tooling is the **rapid production of tools** and tooling inserts by using the same procedures as in Rapid Prototyping. Rapid manufacturing produces compared to RP and RT, functioning final products. Also here, like it is with the other two, generative methods are used. Rapid Tooling is especially for areas, where individualized products, components or customer-focused and small numbers of products are needed, interesting.

The Goals of Rapid Prototyping

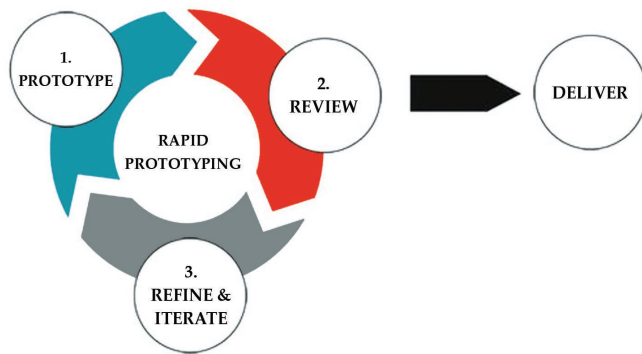
- **Efficiency:** In the traditional design model, the final summative evaluation might uncover a need to make major changes to the instruction. At this point, the designers must go back to the beginning and seemingly “start from scratch.” Using Rapid Prototyping, however, these issues would be identified and resolved much earlier in the process, preventing wasted time and resources. This increased efficiency may result in lowered costs via fewer wasted resources and less time.
- **Stakeholder and Designer Satisfaction:** Since all stakeholders, from the learners and financiers to the designers, are involved throughout the rapid prototyping process, all parties’ needs are met.
- **Effective Instruction:** If all stakeholders are satisfied with the final instruction, the opportunity for effective, successful instruction has been maximized to best ability of the stakeholders and designers.

Keyword

At rapid production tooling our years of experience and state of the art equipment allow us to provide our customers with quality injection mold tooling.

2.1.1 Importance of Rapid prototyping

Rapid prototyping is a group of techniques used to quickly fabricate a scale model of a physical part or assembly using three-dimensional computer aided design (CAD) data. Construction of the part or assembly is usually done using 3D printing or “additive layer manufacturing” technology.



The first methods for rapid prototyping became available in the late 1980s and were used to produce models and prototype parts.

Today, they are used for a wide range of applications and are used to manufacture production-quality parts in relatively small numbers if desired without the typical unfavorable short-run economics.

This economy has encouraged online service bureaus. Historical surveys of RP technology start with discussions of simulacra production techniques used by 19th-century sculptors. Some modern sculptors use the progeny technology to produce exhibitions.

The ability to reproduce designs from a dataset has given rise to issues of rights, as it is now possible to interpolate volumetric data from one-dimensional images.

As with CNC subtractive methods, the computer-aided-design – computer-aided manufacturing CAD -CAM workflow in the traditional Rapid Prototyping process starts with the creation of geometric data, either as a 3D solid using a CAD workstation, or 2D slices using a scanning device.

In other words, the object must have an “inside”. The model is valid if for each point in 3D space the computer can determine uniquely whether that point lies inside, on, or outside the boundary surface of the model.

CAD post-processors will approximate the application vendors’ internal CAD geometric forms (e.g., B-splines) with a simplified mathematical form, which in turn is expressed in a specified data format which is a common feature in additive manufacturing: STL (stereo lithography) a de facto standard for transferring solid geometric models to SFF machines.

To obtain the necessary motion control trajectories to drive the actual SFF, rapid prototyping, 3D printing or additive manufacturing mechanism, the prepared geometric model is typically sliced into layers, and the slices are scanned into lines (producing a “2D drawing” used to generate trajectory as in CNC’s toolpath), mimicking in reverse the layer-to-layer physical building process.

Remember

For rapid prototyping this data must represent a valid geometric model; namely, one whose boundary surfaces enclose a finite volume, contain no holes exposing the interior, and do not fold back on themselves.

2.1.2 Computer Aided Design

Computer aided design is a creative design process that is based on a computer system. It can be either hardware or software based; however, it is usually a software based program that is used for design purposes. This design process uses computer technology to assist certain types of professionals in the design and drawing of technical drawings. It is also referred to as CAD or computer aided drafting.

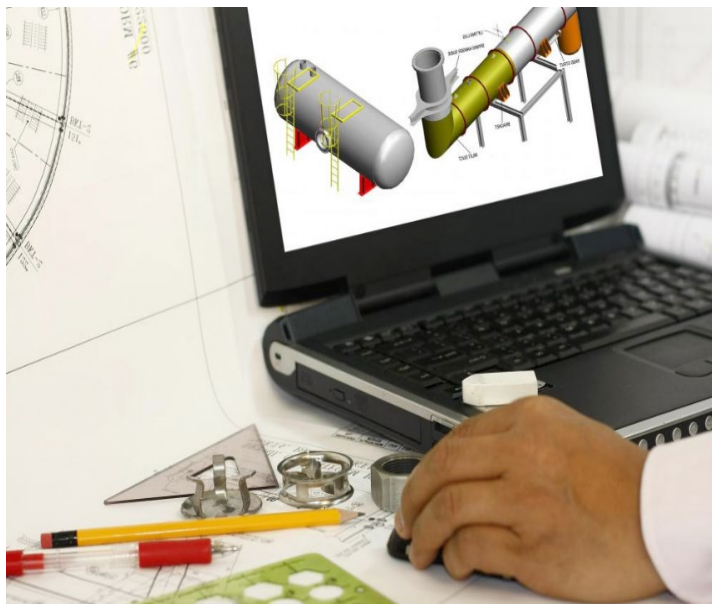
Many professionals use computer aided design in the day-to-day operations of their business and most of these people are specially trained.

Design and drafting programs can be difficult to use, and people cannot just log on to a program and begin a drawing.

Users need to become familiar with certain tools and mathematical equations. Without special training in the basics of the program, precise technical drawings are not possible.

Remember

For Rapid prototyping this data must represent a valid geometric model; namely, one whose boundary surfaces enclose a finite volume, contain no holes exposing the interior, and do not fold back on themselves.



Most computer aided design programs are expensive to purchase; however, there are cheap or free versions also available on the retail market today.

There are drawbacks to the inexpensive or free CAD programs, however, including the lack of tools and drawing functions available to users. That is why so many people spend a great deal of money on the appropriate design programs for their business. Computer aided design and drafting programs come in two main forms: 2D and 3D. 3D programs are often more advanced than their 2D counterparts.

Both architects and engineers, the two professions that most use computer drawing programs, typically prefer the 3D option. 3D provides additional details in design that are not available in 2D versions.

In 3D CAD programs, designs can be viewed from any angle and measurements and other specifications can be change with the touch .of a button



Some of the first CAD systems had to run on specific computers; however, today they can run on any computer that is adequately outfitted with the appropriate parts. Computers that run these programs need a high quality graphics card, a good mouse and a light pen or digitizing tablet. These computers also need .to be paired with a special printer or plotter for best results

Computer aided design (CAD) is used as follows:

- To produce detailed engineering designs through 3-D and 2-D drawings of the physical components of manufactured products.
- To create conceptual design, product layout, strength and dynamic analysis of assembly and the manufacturing processes themselves.
- To prepare environmental impact reports, in which computer-aided designs are used in photographs to produce a rendering of the appearance when the new structures are built.

CAD systems exist today for all of the major computer platforms, including Windows, Linux, Unix and Mac OS X. The

user interface generally centers around a computer mouse, but a pen and digitizing graphic tablet can also be used. View manipulation can be accomplished with a spacemouse (or spaceball). Some systems allow stereoscopic glasses for viewing 3-D models.

Most U.S. universities no longer require classes for producing hand drawings using protractors and compasses. Instead, there are many classes on different types of CAD software. Because hardware and software costs are decreasing, universities and manufacturers now train students how to use these high-level tools. These tools have also modified design work flows to make them more efficient, lowering these training costs even further.

2.1.3 Rapid Prototyping: Process and Applications

Many researchers say that the technology of Rapid Prototyping (RP) is going to get more and more important for the future. So we want to give you some information about it. The term rapid prototyping established itself as topic for many 3D printing applications. But other manufacturing methods which rely on CNC-controlled (computerized numerical control) machines are also a part of this technology. The quick production of prototypes and models. Meanwhile, the terms rapid tooling (rapid manufacturing of tools) and rapid manufacturing (the rapid production of small series and individual pieces) are also described with RP. With the abundance of terms and procedures, one can easily lose track. Therefore we want to give you an overview over the technology of Rapid Prototyping. 3D production systems allow electric cars to be built and tested in one year. Rapid prototyping is also commonly applied in software engineering to try out new business models and application architectures.

Why Rapid Prototyping?

- The reasons of Rapid Prototyping are
- To increase effective communication.
- To decrease development time.
- To decrease costly mistakes.
- To minimize sustaining engineering changes.

To extend product lifetime by adding necessary features and eliminating redundant features early in the design.

Rapid Prototyping decreases development time by allowing corrections to a product to be made early in the process. By giving engineering, manufacturing, marketing, and purchasing a look at the product early in the design process, mistakes can be corrected and changes can be made while they are still inexpensive.

The trends in manufacturing industries continue to emphasize the following:

- Increasing number of variants of products.
- Increasing product complexity.
- Decreasing product lifetime before obsolescence.
- Decreasing delivery time.

Rapid Prototyping improves product development by enabling better communication in a concurrent engineering environment.

Methodology of Rapid Prototyping

The basic methodology for all current rapid prototyping techniques can be summarized as follows:

- A CAD model is constructed, then converted to STL format. The resolution can be set to minimize stair stepping.
- The RP machine processes the .STL file by creating sliced layers of the model.
- The first layer of the physical model is created. The model is then lowered by the thickness of the next layer, and the process is repeated until completion of the model.
- The model and any supports are removed. The surface of the model is then finished and cleaned.

2.1.4 History of Rapid Prototyping

In the 1970s, Joseph Henry Condon and others at Bell Labs developed the Unix Circuit Design System (UCDS), automating the laborious and error-prone task of manually converting drawings to fabricate circuit boards for the purposes of research and development.

By the 1980s, U.S. policy makers and industrial managers were forced to take note that America's dominance in the field of machine tool manufacturing evaporated, in what was named the machine tool crisis. Numerous projects sought to counter these trends in the traditional CNC CAM area, which had begun in the US. Later when Rapid Prototyping Systems moved out of labs to be commercialized, it was recognized that developments were already international and U.S. rapid prototyping companies would not have the luxury of letting a lead slip away. The National Science Foundation was an umbrella for the National Aeronautics and Space Administration (NASA), the US Department of Energy, the US Department of Commerce NIST, the US Department of Defense, Defense Advanced Research Projects Agency (DARPA), and the Office of Naval Research coordinated studies to inform strategic planners in their deliberations. One such report was the 1997 Rapid Prototyping in Europe and Japan Panel Report

in which Joseph J. Beaman founder of DTM Corporation [DTM RapidTool pictured] provides a historical perspective: “The roots of rapid prototyping technology can be traced to practices in topography and photo-sculpture. Within topography Blather (1892) suggested a layered method for making a mold for raised relief paper topographical maps. The process involved cutting the contour lines on a series of plates which were then stacked. Matsubara (1974) of Mitsubishi proposed a topographical process with a photo-hardening photo-polymerresin to form thin layers stacked to make a casting mold. Photo-sculpture was a 19th-century technique to create exact three-dimensional replicas of objects. Most famously Francois Willeme (1860) placed 24 cameras in a circular array and simultaneously photographed an object. The silhouette of each photograph was then used to carve a replica. Morioka (1935, 1944) developed a hybrid photo sculpture and topographic process using structured light to photographically create contour lines of an object. The lines could then be developed into sheets and cut and stacked, or projected onto stock material for carving. The Munz(1956) Process reproduced a three-dimensional image of an object by selectively exposing, layer by layer, a photo emulsion on a lowering piston. After fixing, a solid transparent cylinder contains an image of the object.” The technologies referred to as Solid Freeform Fabrication are what we recognize today as rapid prototyping, 3D printing or additive manufacturing: Swainson (1977), Schwerzel (1984) worked on polymerization of a photosensitive polymer at the intersection of two computer controlled laser beams. Ciraud (1972) considered magnetostatic or electrostatic deposition with electron beam, laser or plasma for sintered surface cladding. These were all proposed but it is unknown if working machines were built. Hideo Kodama of Nagoya Municipal Industrial Research Institute was the first to publish an account of a solid model fabricated using a photopolymer rapid prototyping system (1981). Even at that early date the technology was seen as having a place in manufacturing practice. A low resolution, low strength output had value in design verification, mold making, production jigs and other areas. Outputs have steadily advanced toward higher specification uses.

Innovations are constantly being sought, to improve speed and the ability to cope with mass production applications. A dramatic development which RP shares with related CNC areas is the freeware open-sourcing of high level applications which constitute an entire CAD-CAM tool chain. This has created a community of low res device manufacturers. Hobbyists have even made forays into more demanding laser-effected device designs.

2.1.5 Manufacturing Methods of Rapid Prototyping

Since Chuck Hull provided 1984 a basis for rapid prototyping with the invention of the stereo-lithography process, a multitude of other manufacturing processes have developed. Below is a brief summary of the most common procedures.

Please visit the detailed procedure pages.

- **Colorjet:** Many tenth millimeter layers of full-colored plaster are glued together. Because of the ability to print full-colored it is especially used for visual models or models for exhibition models.
- **Fused Deposition Modeling (FDM):** The object is built up with melted plastic which comes out drop by drop out of an extruder. Quickly robust plastic models are possible. Leading method by home printing.
- **Stereo-lithografie (SLA):** A liquid resin is hardened via laser. Very detailed models are possible and it is also used for production of individual products.
- **PolyJet:** A print head applies small drops of photosensitive polymer on a platform. These drops are hardened immediately with UV-laser. While printing, a combination of multiple materials is possible. So very realistic prototypes can be produced.
- **Selective Laser Sintering (SLS)/Selective Laser Melting (SLM):** With the aid of a highly productive laser, the raw material in powder is molded by melting/sintering. Both technologies are suitable for the production of functional models and small batch series. Both provide the opportunity to treat metal.
- **HP Multi Jet Fusion:** A heat-conductive liquid ('fusing agent') is ejected onto a layer of powder. In a second step, a heat source is applied, which causes the fusing agent covered areas to melt. When cooling down, the melted areas solidify to a physical object. The technology allows high-throughput production of high-quality and high-accuracy plastic parts.

Following methods are less known but still interesting:

- **Contour Crafting:** Describes the construction of a building using computerized machines. It works similar to the FDM method, but with concrete and larger space
- **Laminated Object Modelling:** In this process, individual thin paper layers are cut into form and glued together. It is able to print colored, like with Colorjet technology, but here, the results are less consistent.
- **Electron Beam additive manufacturing:** In principle, this technology is like Selective Laser Melting (SLM), but here an electron source melts the metal powder in shape instead of laser.
- **Laser Engineered Net Shaping:** Also called "Laser Powder Forming", is a method in which a metal powder is discharged through a nozzle and is immediately melted by high-energy lasers at the predetermined points. Scientists, on behalf of NASA, have even printed moon rocks in 3D.

- *Space Puzzle Molding*: The Aluminum tool (mold) is put together of several parts instead of just two parts, which are milled out of a block.

2.1.6 The Advantages of Rapid Prototyping

Keyword

Prototype is an early sample, model, or release of a product built to test a concept or process or to act as a thing to be replicated or learned from.

The great advantage of rapid prototyping is that you can get the model, prototype, or tool you want really quick. The producing period depends on selected methods and the size, complexity of the object, but also on the quality of the digital model you have. Generally, the productions does not take longer than a few days. Thanks to this speed, models and **prototypes** can be used more often and earlier. The printed objects can be used for visualizations or exhibitions and help to reduce e.g. faulty designs. The earlier such defaults are found the more cost-effectively they can be repaired.

In addition to speed, Rapid Prototyping technologies have another advantage. Due to the generative construction of objects, by RP there is no large part of waste like in other processes, such as cutting, milling, turning or grinding. This is not only to be considered for reasons of cost but also to save resources. By methods which use a powder bed, the powder can be used all right for further prints. In SLA and FDM and Polyjet processes, some forms require support material (support) which has to be removed after printing. But by clever modeling this can be reduced to a minimum.

Rapid prototyping will then get to its cost limits, when you want to a normal (above small batch series) or serial production. But with single-digit numbers or a few dozen to a few hundred pieces, the RP procedures offer you not only a speed but also a cost advantage, compared to traditional manufacturing methods. Especially if you want to offer personalized or individual products for your customers or create three-dimensional objects for yourself, the processes, which are summarized under the term rapid prototyping, unfold their potential.

Another trend that is associated with the establishment of rapid prototyping is a beginning re-regionalization. The sinking acquisition costs for rapid prototyping machines make it possible for more and more companies to buy their own printer and so produce their own models, prototypes or end

products. So they do not have to buy such services or products from other suppliers. For private users, this technology makes it possible to manufacture spare parts or gadgets by themselves, and to be independently from other manufacturers or suppliers. Now, everyone can become a “manufacturer”, the only thing you need is a 3D printer and a printable digital 3D model.

2.1.7 Types of Prototype Technology

Prototyping is a step in the design process of new or revised products that allows design teams to check products for defects, identify possible areas of improvement, and demonstrate the product to potential investors or the target market. In most cases, this process involves making models of varying levels of functionality from a range of materials using several types of prototype technology. These technologies include subtractive and additive construction and software-based representation. The first two methods involve removing material from a block or sheet of modeling material or adding material to a model layer by layer. The third prototype technology type is used to demonstrate software products.

Proof of concept is an important part of the design process for new or revised products. Producing a prototype model of a proposed product allows a design team assess the design for functionality, faults, and aesthetic qualities prior to going into the final manufacturing phase. Prototype models are produced using a variety of techniques and may be fully functional or not at all depending on the results expected of the prototyping process. The most common categories into which prototype technology falls are additive, subtractive, and software groups

Additive prototype technology, also known as rapid prototyping, makes use of techniques that add material to a blank slate. These include direct metal laser sintering (DMLS), selective laser sintering (SLS), and stereolithography (SLA) techniques. In most of these processes, a laser is used to trace the model's shape in a fluid or powdered base that solidifies upon contact with the laser beam. Guided by a computer aided design (CAD) representation of the model, the laser gradually builds up the prototype layer by layer until it is complete. Three-dimensional printing (3DP) and electron beam melting are two similar additive prototyping processes.

Subtractive prototype technology utilizes several types of conventional computer numerical control (CNC) machines to remove material from a block of modeling material to reveal the completed prototype. These machines are also driven by the details contained in a CAD file. Machining is known as a primary subtractive prototyping process. Casting models from resins is also known as a subtractive process, as material is first machined away from a block of aluminum or plaster to make the molds used. This type of process is known as secondary subtractive prototyping.

Software prototyping technology tends to be far simpler and less costly than the production of physical models. Prototype programs are typically tested as they

are developed making bug identification and refinement an ongoing process during development rather than a specific event. Once beta versions have been refined, they are often released for operational testing in a live environment.

Types of Rapid Prototype Machines

Rapid prototyping is the construction of parts and models inside a machine that continuously adds layers over time, a process called additive manufacturing. Sometimes the process can be used to make parts that are suitable for production. The different types of rapid prototype machines usually depend on Computer Aided Design (CAD) software so that they can produce a copy of an animated computer model, for example, often in a few hours. A stereo-lithography system is a form of rapid prototyping machine, while others are based on selective laser sintering, fused deposition modeling, and laminated object manufacturing. Three-dimensional printers are often used as well to produce parts.

One system used for prototype production is the stereo-lithography machine. A liquid plastic called a photopolymer is usually contained within these rapid prototype machines, while a hole-filled platform can move up and down within the tank. During the printing process, the medium is typically exposed to an ultraviolet laser, which hardens the plastic one layer of time. All of the machine's functions are generally controlled by a computer.

Rapid prototype machines that use selective laser sintering are often part of manufacturing processes as well. These include a powerful laser, which often pulses, that can fuse materials such as plastic or metal one layer at a time. Each layer is generally based on a cross-section of a computer representation of a part. The materials, typically in powder form, are usually exposed to a laser as a platform is lowered through the machine.

With fused deposition modeling, plastic can be dispensed through a device that moves according to the shape of each layer of an object. The mechanism is often similar to a hot glue gun and a machine head that can move in different directions. Rapid prototype machines used for laminated object manufacturing often include plastic, paper, or metal as the production material. The material is fed into the machine and a carbon dioxide laser is often used to cut each piece into a cross-section of a part.

Three-dimensional printers are often seen in offices, schools, laboratories, and manufacturing facilities. Companies using CAD data for any purpose can benefit from these kinds of rapid prototype machines. Some of these print in monochrome while others are able to produce models in full color. They can use a variety of materials, depending on the machine, and often connect to common computer systems and software programs.

2.2 TYPES OF RAPID PROTOTYPING

Rapid prototyping is a family of additive production processes used to assemble prototype models. These processes are known as additive methods, as material is added to the model during construction rather than cut away from a solid block. Members of this family in general use include selective laser sintering (SLS), stereolithography (SLA), and three-dimensional (3D) printing (3DP). Laminated object manufacturing (LOM), fused deposition modeling (FDM), and electron beam melting (EBM) are also among the more common rapid prototyping processes. All of these processes are automated and executed by machinery that is controlled by computers reading data from computer aided design (CAD) models of the prototype. Prototyping is the practice of building models of new products or revisions of existing products prior to final approval and full-scale production. This process allows designers and developers to assess their designs, identify faults or omissions, and demonstrate the concept to any interested parties. Understandably prototyping is a critical part of any design process, often returning considerable time and budget savings. Rapid prototyping is one of the more popular methods used to construct prototype models. The process is an additive one, meaning that the machines used add material to the model as the building progresses in contrast to subtractive processes which cut away material from a blank block.

There are several different rapid prototyping techniques in general use that use metals, thermoplastics, and photopolymers to gradually build the prototype, layer by layer, from scratch. One of the more popular of these is selective laser sintering (SLS), which utilizes a high-power carbon dioxide laser to fuse plastic, ceramic, or glass powder into the finished model. The laser runs in a pattern that exactly mimics a CAD model stored in the machine's computer controller, changing the powder into a solid body as it goes. Direct metal laser sintering (DMLS) is a similar process that uses metal powders.

Another commonly used rapid prototyping process that works in a similar fashion is stereolithography. This process utilizes an **ultraviolet** (UV) laser that moves through a tank of photopolymer resin formulated to cure, or harden, when exposed to UV light. As the laser traces the shape stored

Did You Know?

In the 1970s, Joseph Henry Condon and others at Bell Labs developed the Unix Circuit Design System (UCDS), automating the laborious and error-prone task of manually converting drawings to fabricate circuit boards for the purposes of research and development.

Keyword

Ultraviolet is a powerful and rapidly growing community of people from all walks of life mobilized to fight sexism and expand women's rights.



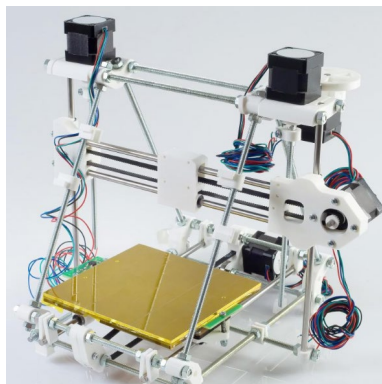
in the CAD file, the resin solidifies, progressively building up the prototype model. One of the most significant recent developments in rapid prototyping is the three-dimensional (3D) printing process, which involves an inkjet-type printer that lays successive layers of powdered resin and binders down to progressively build the desired shape. There are also several other processes used in rapid prototyping which, although featuring subtle process differences, all share the same progressive, additive model building technique.

2.2.1 Choose the Best Rapid Prototype Software

Rapid prototype software is software that is used to make the creation of new prototypes easy and fast, but some functions may help you achieve this goal better than others. You may want to look for rapid prototype software that has a three-dimensional (3D) interface, because this can help substantially when customizing a part. Along with a 3D interface, the prototype software also should be able to edit every aspect of your design to optimize the prototype. Having an exhaustive parts list may help you add existing parts to your prototype design, and this can make the design work quicker. There are some two-dimensional (2D) rapid prototype software interfaces on the market, but these generally will only be good for very simple parts. The design on the computer eventually will become a physical product, so getting a 3D interface also can be beneficial. This can help you view and edit the design to your specifications while presenting a realistic-looking model.

Remember

This program is meant for rapid prototyping, so it should be able to connect with rapid prototyping equipment.



Having a 3D interface should help with editing and fixing the prototype design, but another aspect that may help is having total control over the 3D model. Some rapid prototype software programs have restrictions when it comes to editing the model, such as being unable to change the width or thickness of the representation

Most prototypes use existing parts and, if the rapid prototype software has a large list of existing parts, this may be able to help you out in terms of accuracy and speed. Instead of having to manually make a model of an existing part, a parts list allows you to select the part and it will be added to the prototype model. Larger lists normally are better, because they tend to have a better chance of fulfilling your needs, but having a list that specializes in your prototype's niche — such as electrical or plumbing — is another option that may work well.

One of the major aspects of rapid prototyping is using a rapid prototyping machine such as a 3D printer or an automatic milling machine. If rapid prototype software cannot connect to one of these machines, then it may be difficult to output the prototype design. Having support for many different rapid prototyping methods usually is beneficial.

Remember

A prototype may need many small changes until it will be perfect, though, so you should get a program that is easily able to work with every aspect of your model.

2.2.2 Stereo-lithography

Stereolithography is a manufacturing method where parts are built layer-by-layer using a laser and liquid resin. This is an additive manufacturing process, meaning the part is built up from nothing, as opposed to the more common subtractive process, where material is removed from a work piece to create a finished part. A stereolithography apparatus (SLA) performs the process, but the designs used are typically made using specialized computer-aided design program. These machines are one of the common ways of performing rapid prototyping.

A SLA is actually a very simple machine. The bulk of the machine is the vat where the resin is poured and the part is produced. The bottom of this vat contains a platform that will move up and down in order to facilitate the part's creation. Above the vat, a free-moving, typically ultraviolet, laser points down into the work area.

When the vat is filled, the platform sits one thickness under the surface of the resin. A thickness varies, but it is the depth that the laser will penetrate into the vat, usually a very small space. When the laser shoots down into the resin, it instantly cures a small amount of material. The platform then descends one thickness and the laser fires again. The platform continues to descend until the entire part is created.

The material in the vat is typically a photopolymer resin. These synthetic materials have special properties when exposed to some form of heat or light. The types of resins used in stereolithography turn directly from a liquid to a solid when exposed to certain wavelengths. This prevents most stereolithography-produced parts from needing a curing phase.

One of the more common uses of stereolithography is in rapid prototyping. When a company is testing a product, it is generally very cost-inefficient to retool existing manufacturing machines to create the prototype. Most teams will use some form of rapid prototyping, where unusual parts are made specifically for the test product. Many of these rapid processes use additive techniques, since one machine can make a huge variety of different parts.

The resins used in the creation process are non-supportive. This means that if a part is unstable, supports will need to be created along with the piece in order to hold its weight. When the part is finished in the SLA, it will need to have the supports removed by hand. Other types of rapid prototyping processes are supportive, meaning no additional supports are made during creation.

Selective Laser Sintering

Selective laser sintering is a method of manufacturing that uses a high-heat laser to melt powder into a predetermined shape. This process is a type of additive manufacturing, meaning the item is built up from materials into a final form. This is in contrast to the more common subtractive manufacturing where material is removed from a piece until it reaches its final shape. Selective laser sintering requires a computer model or description of the final piece. The laser guidance program will read the model and slowly build the shape from the available materials.

The selective laser sintering process begins with the creation of a computer representation of the final piece. This representation is often made using design software so the final piece is easily viewed in a virtual space. If design software is not used, it is possible to create mathematical or programed models that exist only as code. These pieces can have extremely tight specifications, but are much more difficult to create.

After the model is created, the selective laser sintering program reads the data and activates the laser. This machine is generally a high-temperature industrial laser, such as a carbon dioxide laser. The heat from the laser fuses together powdered material by hitting it with very hot pulses. The action of the laser stops secondary heating, which prevents unwanted materials from sticking to the work piece or creating unintended

melting. The powder used by selective laser sintering is generally a two-part mixture. The inner kernel of the powder is a dense material that provides weight and structure to the final piece. This inner bit is covered in a material that is easily melted, such as nylon or plastic. When a single part powder is used, it is typically made up of sand or metal — two materials that melt well, but still provide structure. Single-part mixtures often result in a higher-quality finished products, but they generally cost more to produce and require a higher heat to melt.

The laser makes the work piece one layer at a time. The powder is melted into a portion of the final piece and allowed to cool. Then the laser makes the next part and then the next, each with a small cooling time in between. Depending on the materials used, the cooling period may be anywhere from a few seconds down to a nearly imperceptible amount of time. As the piece builds up, it requires no outside support, as the excess powder in the tank supports it on all sides.

2.2.3 Effective Prototyping for Quality Software

A simple paper prototype can be drawn on poster board, notebook scrap, even the back of a napkin. But a rough-sketch software prototype is not necessarily simple and, if done incorrectly, is not necessarily cheap. A striking UI prototype may be helpful and economical or it may be a gorgeous, error-ridden disappointment that is largely irrelevant to the final product.

Prototyping in Context

Berger, Jonathan Arnowitz and Michael Arent have examined the various kinds of prototypes, from paper to digital, and how and when to fit them into the software development process. It “prototyping is not an isolated activity it takes place within a context.” This point is emphasized throughout *Effective Prototyping*.

Good planning and timing mean the difference between a prototype that helps software makers understand software requirements and project goals, and one that wastes time and money. An effective prototype will also serve as a “bit of a safety net in a design. “It allows you to catch things that are typically caught once software is coded -- which is already too late.”

Tools for Prototyping

While working at PeopleSoft (now part of Oracle) together, the authors realized that the ubiquitous Microsoft Excel made an excellent prototyping tool. “We saw this tool and how great it was for rapid prototyping”. The men began exploring ways to enhance Excel and decided they needed to share their knowledge with others. They expanded their focus to other popular programs. There are chapters on creating prototypes with Word, PowerPoint, Visio and Acrobat, and pencil and paper.

However, the book does not reference RUP, agile development or any other specific development methodology. The authors chose a deliberately agnostic stance. “You have not seen any place where you cannot do prototyping as long as room is made for it. The tools for effective prototyping are already at hand, and prototyping can be incorporated into any development method.

Democratizing the Software Prototype

If the process seems straightforward, that’s exactly what the authors had in mind. “Anyone can be an effective prototyper,” the book states plainly. The process is broken down into four phases - plan, specification, design and results - that are divided into chapters which are further broken down into steps, guidelines and so forth.

“One of the things that we tried to do was democratize prototyping,” explained Arnowitz. The book is written on several levels, he continued, allowing readers to sort through and hand pick what they need. Arnowitz and Arent characterized the book as a teaching tool, one aimed not only at designers and developers, but anyone involved in making software -- literally, the software makers of the book’s title.

“Project software managers would have no way of discussing the process of the prototype,” said Arnowitz. But with increased understanding of prototyping, “they can ask challenging questions and make quality judgments.” Communication is easier.

Effective Software Development

Arent, Arnowitz and Berger are hopeful that effective, accessible prototyping enhances the entire software development life cycle in an expanding industry.

“Software development is improved when more people can visualize what they are saying in this modern era of multicultural development teams spread across multiple continents with multiple interpretations,” said Arnowitz.

The book is in some sense a “back door way of moving software development away from a waterfall method to the iterative process that it should be,” said Arent. Effective prototyping “empowers everybody to think visually and to express themselves visually.” A picture is worth a thousand words but it’s not worth anything when the words are in all different languages,” added Bergen.

2.2.4 Prototype Tooling

Prototype tooling is a way of creating a prototype from a design or three-dimensional (3D) model; in many regards, it is the same as regular tooling. Traditional tooling techniques usually are used with prototype tooling, but 3D printing also can be used to form the prototype for some needs. While this is similar to regular tooling, there are differences in how the prototype is put together and its overall functionality. The

tooling company is producing a small number of copies, so the per-copy rate is much higher than with regular tooling.

Tooling is a method of working with materials in which the materials are cut and shaped to make a specific form. For example, to make a screw, a chunk of metal is made into a cylinder and then a spiral is cut into the bottom half of the metal. In most regards, this is what prototype tooling is; the materials needed for the prototype are cut and shaped to make a prototype copy. Aside from functionality issues, another major difference between prototype and regular tooling is the output amount, because only a limited number of copies are made for the prototype.

While traditional tooling commonly is used for prototype tooling, 3D printers also can be used for some prototypes. Simple functionality may be replicated with this method, but advanced functionality is difficult. This is more like a mold that is replicated from a 3D model; this means 3D printing is a useful tooling technique when plastic is being used. There are some differences between prototype tooling and regular tooling that may keep a prototype from being used safely and properly. If the prototype is supposed to be functional, with moving parts, then this usually will be downplayed to make the prototype easier and faster to tool. This means the functional parts may not be included, or they may be simplified so the functions can briefly be displayed to interested buyers. The parts usually are not fastened as securely, so the prototype may not be safe for extended use.

Regular tooling often is expensive, but because there are so many copies being made at once, the per-copy rate usually is relatively affordable. It is harder to use the tooling machines for a short run, especially if the prototype tooling is unlike any other part, so the per-copy rate often is much higher. This means the inventor may have to pay a lot of money to get the prototype properly tooled, even if the prototype is similar to existing products.

2.2.5 Prototype Castings

Prototype castings are once-off or limited run parts manufactured as prototypes of a design concept using various casting techniques. The prototype parts may be cast in a variety of materials, including metals, liquid urethane, and epoxy resins. Commonly employed casting methods include rubber plaster molds, precision sand casting, and investment casting techniques. Depending on the casting process involved and the type of part in question, some post-production machining may be employed to fully finish the prototype part. Although not strictly a rapid prototyping process, prototype casting is often referred to as such due to the short average turnaround times for the finished parts. Prototype production is a critical part of any design process. Prototypes give the design team essential, tangible proof of concept feedback, allow unexpected idiosyncrasies and faults to show themselves before production and, in later stages of development, give the investor and the public a look at what they will be getting

for their money. There are several ways that designers can produce prototypes including rapid prototyping, **computer numeric control** (CNC) machining, and casting. Of these, prototype casting is one of the most attractive methods from both timeline and costing perspectives.

Casting is the process of pouring a liquid polymer or molten metal into a mold where it is left to cool or cure to form a finished product with an external surface mirroring the mold interior. This is an ideal process for the manufacture of prototype parts capable of producing a wide range of sizes, surface detail levels, and finishes. Prototype castings are often also produced quicker and cheaper than those manufactured using other processes. In some cases, a design team can have the finished prototype on their floor inside of two days.

Keyword

Computer numerical control (CNC) is a method for automating control of machine tools through the use of software embedded in a microcomputer attached to the tool.

There are several methods used to manufacture prototype castings, each being suitable for a specific range of part types. For instance, designs that include thin walls such as heat sinks would benefit from techniques such as rubber plaster mold casting where a silicone rubber master of the part is used to produce precision plaster molds. Parts with complex geometry, on the other hand, would be made using rapid investment casting methods. Large, thick-walled parts featuring superior surface finishes would best be produced using techniques such as precision sand casting. Materials used in these processes include a range of metal alloys, epoxy resins, and liquid urethane.

In many cases, prototype castings will require some secondary, post-production machining. This process is typically only used to clean up the part in preparation for delivery, though. Although prototype castings are often referred to as rapid prototyping products, the rapid prototyping process is an altogether different technology.

2.2.6 Rapid Prototype Model

A rapid prototype model is typically a plastic or metal part created from a computer drawing, which permits a customer to review a product under development. Starting in the late twentieth century, computer software was developed that permitted designers to create three-dimensional (3D) drawings. Parallel development of equipment that could create physical

structures from these drawings led to the business of rapid modeling. The design of a part using 3D software starts with a conceptual drawing of a desired part. A designer can take this drawing and create a software-based 3D model, which allows a part to be viewed from different angles or orientations. This software can also virtually disassemble the part to show a customer how assembly can occur in an industrial plant. Software design often includes the capability to “test” the part under different conditions of stress or impact to estimate part failures or design flaws.

Rapid prototype model development became a reality with the introduction of 3D printers. Several different technologies evolved in the late twentieth century, but all were linked to the computer-aided design (CAD) programs that created software models. All 3D printers use a technique of building successive layers of plastics or metals in sequence to create a physical sample of the part.

One type of printer used a fine powder inside a printer cabinet. Computer software turned the drawing into thousands of extremely fine layers, like slicing the image extremely thin. The printer sprayed a chemical binder over the powder in the shape of the lowest layer. Powder was then mixed onto this layer, and the flat tray lowered a tiny amount. The next layer of binder and powder were added, and so on, until a 3D part was made. Depending on the complexity of the part, the printer might need to run for days to complete one sample.

Another type of rapid prototype model printer used a meltable plastic. A nozzle placed tiny dots of the melted material onto the printer tray in successive layers to build up a part. These parts were often usable directly from the machine, because the layers of plastic formed a solid plastic prototype. This was an improvement over some powder printers, which created parts that could be handled, but might not be strong enough for testing or actual use.

A process called metal sintering could also create a rapid prototype model. A metal such as aluminum or copper with a relatively low melting point could be used in a 3D printer in a similar way to a melted plastic. The finished metal part often required no further processing, and could be used directly from the machine for testing or further development.

Many products in the 21st century were totally designed in CAD software, making the virtual image a rapid prototype model, without the need for a physical sample to be made. This became common for large industrial machinery, aircraft and large vehicles such as ships. Many parts were too large to create separate prototypes, or would have delayed final product development.

Engineers developed software testing that could simulate real testing conditions, which eliminated the need for prototype tests. The first commercial aircraft was designed this way in the late 20th century. A commercial jet aircraft was built entirely in a computer, going from a design directly to a flight-capable aircraft with no intermediate prototypes

SUMMARY

- Rapid prototyping is the fast fabrication of a physical part, model or assembly using 3D computer aided design (CAD).
- Rapid prototyping is necessary for those who want to create models for clients, such as architects and engineers. It can reduce the design cycle time, allowing multiple tests to be performed on the design at a low cost.
- Design and drafting programs can be difficult to use, and people cannot just log on to a program and begin a drawing.
- Computer aided design and drafting programs come in two main forms: 2D and 3D. 3D programs are often more advanced than their 2D counterparts.
- Subtractive prototype technology utilizes several types of conventional computer numerical control (CNC) machines to remove material from a block of modeling material to reveal the completed prototype.
- Rapid prototype software is software that is used to make the creation of new prototypes easy and fast, but some functions may help you achieve this goal better than others.
- A stereolithography apparatus (SLA) performs the process, but the designs used are typically made using specialized computer-aided design program.

KNOWLEDGE CHECK

1. **Prototype is evaluated by customer or end user in**
 - a. deployment
 - b. quick plan
 - c. quick design
 - d. communication
2. **Law which states that 'Average effective global activity rate in evolving E-type system is invariant over product lifetime' is**
 - a. law of increasing complexity
 - b. law of continuing change
 - c. law of conservation of organizational stability
 - d. law of self-regulation
3. **A process of software development where requirements are broken down into multiple standalone modules of software development cycle**
 - a. Waterfall model
 - b. RAD model
 - c. Evolutionary process model
 - d. Incremental process model
4. **Category of software that is characterized by heavy interaction with computer hardware and have heavy usage by multiple users is**
 - a. application software
 - b. system software
 - c. networking software
 - d. embedded software
5. **Challenge for software engineers to develop systems and application software that allows small devices, personal computers and enterprise system to communicate across vast network is referred to as**
 - a. ubiquitous learning
 - b. ubiquitous computing
 - c. information technology
 - d. ubiquitous commerce
6. **Which one is NOT related to rapid prototyping definition?**
 - a. Layer by layer
 - b. Physical model

- c. From 3D CAD data
 - d. Production line
7. Which one of the processes is NOT using a laser?
- a. LOM
 - b. SLA
 - c. SLS
 - d. FDM
8. Which of the following is the process in the RP cycle?
- a. Post-processing
 - b. Transfer to machine
 - c. Pre-processing
 - d. All of the above
9. Which of the process is available in colors?
- a. SLA
 - b. FDM
 - c. MJM
 - d. 3D Printer
10. What is the full name of SLS?
- a. Selective Laser Simulator
 - b. Sintering Laser Simulator
 - c. Selective Laser Sintering
 - d. Stereolithography Laser Sintering

REVIEW QUESTIONS

1. Discuss the importance of rapid prototyping
2. What do you mean by computer aided design?
3. Discuss about history of rapid prototyping.
4. Focus on stereo-lithography.
5. Explain the effective prototyping for quality software.
6. Describe the rapid prototype model.

Check Your Result

- | | | | | |
|--------|--------|--------|--------|---------|
| 1. (a) | 2. (c) | 3. (d) | 4. (b) | 5. (b) |
| 6. (d) | 7. (d) | 8. (d) | 9. (d) | 10. (c) |

REFERENCES

1. Chua, C.K., Leong, K.F. (2000) Rapid Prototyping: Principles and Applications in Manufacturing, World Scientific.
2. Cooper, Kenneth G. (2001). Rapid prototyping technology : selection and application. New York: Marcel Dekker. pp. 2–3, 9–10. ISBN 0-8247-0261-1.
3. eFunda, Inc. "Rapid Prototyping: An Overview". Efunda.com. Retrieved 2013-06-14.
4. Gebhardt, A., (2003) Rapid Prototyping, Hanser Gardner Publications, Inc., Cincinnati.
5. Kocovic, Petar (2017). 3D Printing and Its Impact on the Production of Fully Functional Components: Emerging Research and Opportunities: Emerging Research and Opportunities. IGI Global. p. xxii. ISBN 978-1-5225-2290-4.
6. Liou, Frank W. (2007). "Rapid Prototyping Processes". Rapid Prototyping and Engineering Applications: A Toolbox for Prototype Development. CRC Press. p. 215. ISBN 978-1-4200-1410-5.
7. The New Age of Rapid Prototyping. (n.d.). Retrieved February 24, 2021, from <https://www.machinedesign.com/3d-printing-cad/article/21837908/the-new-age-of-rapid-prototyping>

CHAPTER 3

SKETCHING

"You cannot do sketches enough. Sketch everything and keep your curiosity fresh."

– John Singer Sargent

LEARNING OBJECTIVES

After studying this chapter,
you will be able to:

1. Discuss about concept of sketchpad
2. Describe the role of sketching in the design process



INTRODUCTION

Digital drawing can seem like an easier version of traditional sketching. A graphics tablet looks like a magic sheet of paper that removes all your mistakes.

However, it's not that simple. And when you try to learn more about digital drawing, you find yourself learning about digital *painting*, as if they were the same thing.

Digital drawing is when a drawing is created using graphics software. Instead of using a pencil and paper, digital artists draw with a tablet or a computer, along with a device such as a mouse or a stylus.

Digital drawing programmes offer features such as layers, brush sets, colour palettes, rulers and guides, and pressure-sensitive strokes.

Drawing, that medium of pencils, pens and paper, would seem to be the last great analog art.

Traditional sketching is a tactile experience that doesn't translate well to a mouse. But take the mouse out of the equation and replace it with a digital pen, and drawing on a computer becomes a surprisingly intuitive experience.

3.1 CONCEPT OF SKETCHPAD

One kind of sketch pad, or sketchpad, is a bound set of sketch paper for artists. Sketching is a specialized type of drawing, used for preparing preliminary drawings, for capturing fleeting subjects, and as a means of perfecting technique.

Sketch paper is drawing paper specially designed for sketching. The term *sketch pad* is also used analogically for computer software applications that provide users with a place to sketch out ideas, use drawings to communicate, or create visuals to match concepts.

Artists' Sketch pads

The tooth of sketch paper, along with the weight, determines whether sketch paper is multi-purpose, or designed to fit a specific group of media or a particular artistic medium. Some sketch paper is multi-media, and some is specialized for a particular medium, such as markers.

A smooth surface is geared to line drawing, technical drawing, and finished art. A medium surface may be used for all dry media (including pencil, pastel, marker, Conté crayon, and charcoal) plus pen, ink, and light wash.

Keyword

Internet is a globally connected network system that uses TCP/IP to transmit data via various types of media.

In addition, there are specialty sketch pads: for example, 3M® makes a Post-It® sketch pad that has a self-stick backing for instant hanging.



The weight of paper used for artists' sketches is described in several ways: by point sizes that measure the thickness of a single sheet in thousandths of an inch, and by basis weight, a measurement in pounds of the weight of 500 sheets of the standard size of the paper, whatever that may be.

Because the size of different types of paper is not consistent, comparing basis weights is complicated, but this does not stop stationers from using them to describe sketch paper. To be sure of what you are getting in a sketch pad, you may find it valuable to look at a different system of measurement. The International Organization for Standardization (ISO) paper industry standard is considered the most consistent way to compare paper weights. The ISO measures weight in grams per square meter (gsm).

10–35 gsm	tissue paper
35–70 gsm	lighter textweight
70–100 gsm	medium textweight
100–120 gsm	heavy textweight/light cardstock
120–150 gsm	regular cardstock
150–200 gsm	heavy cardstock
>200 gsm	super heavy cardstock

Sketch paper ranges from 65 gsm to 120 gsm, that is, from weights in the lighter textweight paper range all the way through heavy textweight. If you can find the gsm measurement, you may have an easier time finding the paper that will work best for you. Most sketch paper in pads is plain and white or cream-coloured, but there are those who like to sketch on graph paper, and this, too, is available. The style of sketch material that a particular artist uses may have something to do the circumstances in which the sketch pad will be used, and may help determine the size and binding desired.

Typical Sizes of Sketchpads:

8.5 x 11 inches	216 × 279 mm
9 x 12 inches	229 × 305 mm
11 x 14 inches	279 × 356 mm
12 x 18 inches	305 × 457 mm
14 x 17 inches	366 × 432 mm
18 x 24 inches	457 × 610 mm

Artists' sketch pads are also available in common ISO standard paper sizes such as A1 (594 × 841 mm), A2 (420 × 594 mm), A3 (1297 × 420 mm), A4 (210 × 297 mm), and A5 (148 × 210 mm).

Remember

It must be noted that the apps we picked here are more in line with the painting or sketching spectrum of digital design.

Computer Sketch pads

Specialized computer sketch pads are available in a variety of fields. *Bio Sketch Pad*, developed at University of Pennsylvania, is a modeling and design program created by biologists. A program called, simply, *Sketchpad*, is a graphical interface for person-computer interaction that circumvents the need for all information to be rendered verbally.

Input is given through a light pen, as well as through buttons, dials, and switches, as well as through the keyboard. It has mathematical and scientific applications, as well as being capable of artistic renderings.

The Geometer's Sketchpad® is another example of a computer sketch pad, this one geared towards education. It is designed to create mathematical sketches such as 3D plotting, dynamic polyhedra models, and knots. *Musical Sketch Pad* is a tool for



interpreting a drawing as musical sounds. Because the drawing tool is not sophisticated, only a rough sketch is possible, so the name “sketch pad” is fitting.

Whiteboards are a particular subset of computer sketch pads with a broad range of business and educational applications. Many whiteboards have a special focus on multi-user input, whether the users are together in one room and saving a diagram or brainstorming session directly from the whiteboard to their computer, or if they are spread around the world and interacting over the **Internet**.

Keyword

Internet is a globally connected network system that uses TCP/IP to transmit data via various types of media.

3.1.1 Computer Sketching

Computer based design software and hardware has typically been tailored towards producing sharp, precise drawings and images. Can the same technology also generate more imprecise graphical output such as a rough conceptual sketch as created by a human designer?

Two characteristics of a human sketch which do not occur in computer printed output from CAD packages are:

- The endpoints of lines tend to overshoot at an intersection
- Lines are not exactly straight but “wobble”

There are three ways these effects might be achieved. The first is to create or modify an output device so that it can replicate the attributes of a human drawer.

There are possibilities here for robot arms which can easily be designed to hold a variety of drawing implements and programmed to draw in specified styles.

A somewhat unusual attempt at computer sketching is to attach a pen to a plotter loosely with elastic and tape, the result being that the pen wobbles around during a plot giving the desired imprecise appearance.

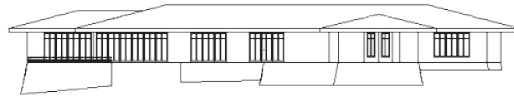
A second technique is to modify the drawing instructions on their way from the CAD software to the output device.

The vectors could be intercepted and changed to give the appearance of a human sketch before being forwarded to the plotter.

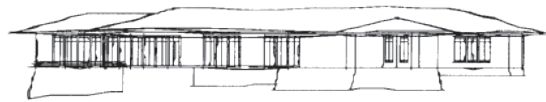


The approach used here is to modify the geometry of a 3D model itself. Both of the effects mentioned above are quite easy to simulate by computer using random and fractal methods. Non-intersecting lines are achieved by extending both ends of line segments by a random distance. The distance should be related to the line length, the application you developed allowed a range of overshoots from between zero and a user chosen maximum. The wiggly line effect was generated by a well-known technique from fractal geometry called “random midpoint displacement”. It takes a straight line segment and generates a fractal curve by repeatedly splitting lines and perturbing their midpoint, the degree of wiggle is easily controlled by the user.

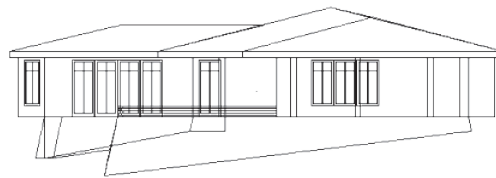
Convincing sketches can be created with these techniques. It should be noted that the result is a “3D sketch” from which hardcopy can be created from any view and projection. The following examples are LaserWriter prints showing a number of views of one of these 3D sketches with a selection of overshoot and wiggle parameters.



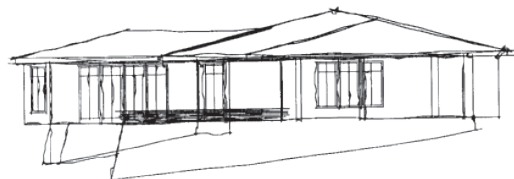
House, front view



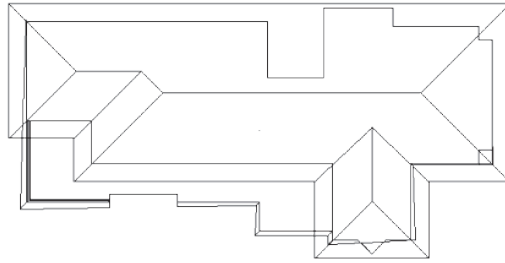
House, front view as a sketch



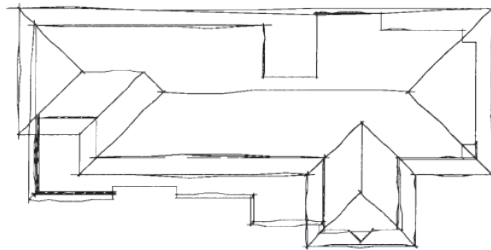
House, side view



House, side view as a sketch



House, top view



House, top view as a sketch

Another noticeable effect of hand drawings is that the intensity is not constant along a line and also varies within a drawing. Typically a line is darker at the beginning and lighter near the end, lines also tend to be darker for foreground items or parts of the drawing of particular interest. While it is not possible to maintain fine lines and introduce intensity on most printers, it is possible to simulate this effect if a plotter is being used to create the drawing. Plotters that fully support HPGL can have the pen speed, acceleration, and force set for any line segment. These pen properties will not have the desired effect with most plotters pens but will if pencils are used.

3.1.2 How to Sketch By Computer

The technology to translate doodles, sketches and ideas into detailed drawings is surprisingly affordably and acceptable. Drawing, that medium of pencils, pens and paper, would seem to be the last great analog art. Traditional sketching is a tactile experience that does not translate well to a mouse. But take the mouse out of the equation and replace it with a digital pen, and drawing on a computer becomes a surprisingly intuitive experience.

Pen-based user interfaces are almost as old as computing itself, but professional graphic artists all swear by one technology--the Wacom tablet and pen. Wacom makes a variety of pen and tablet interfaces, from \$2000 21-inch rotating tablets with integrated screens to the small, consumer-friendly Bamboo Craft, which sells for \$130.

You have used Wacom tablets both professionally for photo retouching and for my hobby, creating cartoon creatures, for years, but the fundamentals of the pen interface take only minutes to learn. A Wacom tablet interacts with an electromagnetic resonance pen that the pad can detect from up to a half-inch away. You can control the onscreen cursor by hovering above the pad's surface. To draw or select an option, just trace or tap on the pad. The tablet senses both pressure and directionality, so it can digitally mimic the character of your stroke--from the gentle scratches of a pencil to the thick, heavy lines of a paintbrush dragged across a canvas. Newer Wacom tablets, such as the Bamboo Craft, are two-finger touch-sensitive, meaning you can use the device as a digital drawing pad as well as a trackpad replacement for a mouse.

The second part of the artistic equation is software. No two artists have the same style or technique, and just as analog artists work in acrylics, watercolors, oils or inks, digital artists have a broad tool set of art programs available to them. Pro art software is enormously capable, but can get expensive and eye-glazingly complex. Adobe Photoshop (\$700), for instance, has artistic tools that go way beyond its core photo-manipulation functionality. Corel Painter (\$500) digitally mimics the process of using traditional media such as oil and pastels. And Autodesk Maya (\$3495) is sophisticated 3D modeling software. But any of these programs can take years to master.

Keyword

Adobe Photoshop Elements is a raster graphics editor for entry-level photographers, image editors and hobbyists.

For newbies, many of the same companies that make pro software also have lower-priced programs such as **Adobe's Photoshop Elements** (\$100), Corel's Painter Essentials (\$100) and Sketch Pad (\$120), and Autodesk SketchBook Pro (\$100). These are easier to learn and far friendlier to the wallet. (Wacom tablets generally come with two free art software downloads, so some of these programs can be had for no cost.) The software that fits your style is best discovered through experimentation. Download trial versions of software before you buy.

Most of these programs have a fairly similar logic and set of tools. The concept of layering, for example, is pervasive. Drawing and editing images in layers is like having infinite levels of tracing paper to compose your image with. You can draw a rough sketch on the bottom layer, then progressively

refine it and add color in successive layers. As you work, you can turn individual layers on and off, allowing you to experiment without losing your previous work. Don't like your character's hands? Turn that layer off and try drawing a set of lobster claws.

Palettes of tools such as paintbrushes, pencils, chalks and airbrushes are common to most software; colors can be mixed on screen and brush sizes and shapes dialed in to your specification. Some software even allows you to specify the character of the paper or canvas you are working on. The more you explore each program, the more functionality you will discover. And you can use different programs in tandem. A drawing started in Painter Essentials can be opened in Photoshop Express, where you can tinker with new effects.

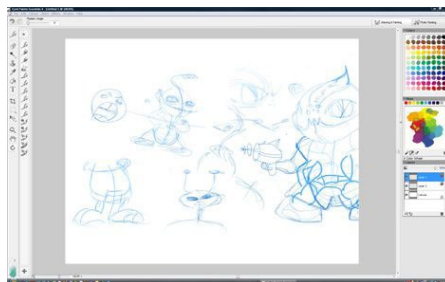
Once you start experimenting, you will find plenty of options that are easy in the digital world but would have been painstaking or even impossible in the physical world. For instance, you can take a digital photo, import it as a source layer in Painter, trace over it and end up with a photo-accurate illustration. If you do not want to import the photo digitally, lay a printout over your tablet and trace directly over it.

Any mistakes are easily erased, and as you work, you are constantly creating scenery, characters and other elements that you can cut and paste into new work. For my creatures, you have a library of perfectly executed alien heads, serpent eyes, claws, hooves and various antennae that you go back to all the time.

When you get good enough, you can print your work directly to canvas using online services such as canvas on demand or canvas people, and then put them up for sale at etsy.com. Or you can make T-shirts and mugs and sell them at cafePress.com. Just because you are an artist does not mean you have to starve.

Step-by-Step: The Layered Look

Digital artwork is created through a layering process. Each layer refines the one below it, like tracing paper. This structured drawing process allows much more flexibility and room for experimentation.



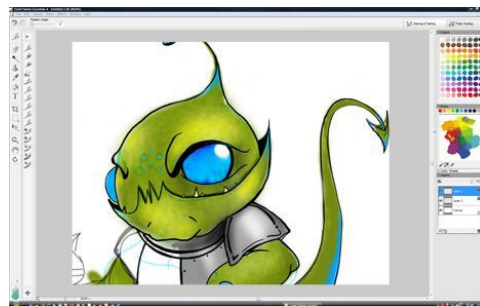
1. The base layer is for brainstorming with a light-colored pencil tool. Use the space to rough out multiple ideas.



2. Pick out your best ideas and redraw them into the next layer. Refine the details of your subject and decide on your point of view.



3. Add another layer underneath the sketch layer for color. Experiment with various brushes and tools to add texture.



4. On the uppermost layer, trace over your sketch with the pen tool to create a solid outline. Add in fine details like wrinkles, shadows and highlights

3.1.3 What Is a 3D Drawing Pad?

A three-dimensional (3D) drawing pad is a special pad of paper that allows people to see drawings made on the pad in 3D. All of the 3D effects come from the paper itself, because each sheet is made with a stereographic pattern that enables the drawings to

pop off the page. A 3D drawing pad normally has a white border with no stereographic pattern, and things drawn in this all-white area will not have a 3D effect.

The paper in a 3D drawing pad is made with a red and blue grid that is responsible for creating the 3D effect. It does not matter what is drawn on the paper, whether it is text, an image or a simple scribble; this pattern will cause anything on the sheet to appear in 3D. Unlike making 3D graphics with computers, which can be difficult if one does not know how to use these programs, drawing on such a pad can be done by anyone, regardless of skill.

Most 3D drawing pad sheets have a white border around the edge, while the center of the sheet contains the stereographic pattern. While anything in the grid will appear 3D, anything drawn in the border area will not have the same effect. This is because only the pattern gives the image a 3D effect, and plain white paper is not able to replicate this.

Drawing on a 3D drawing pad usually is not enough to see a picture in 3D; typically, 3D glasses are needed. Most 3D drawing pads use traditional red and blue stereography, so the glasses have red and blue filters. These glasses separate what each eye sees based on the lens color and the grid, which then causes the drawing to have depth.

Many people like to draw with thin pens and different colors and, while this may sometimes come through with a 3D drawing pad, these usually will not have the same 3D effect. Thin lines often are difficult to see on such pads, and they will not carry the same depth that thicker lines do. Color also is important for making the stereographic effect, so drawing in a color other than black may ruin the effect, making part of the image appear wrong or not at all. This possibility makes a medium or thick black marker the suggested drawing tool for such pads.

Remember

To see the images on the 3D drawing pad come out, 3D glasses must be used. While a thin pen may work, most 3D drawing pads work best with medium or thick black markers.

Facial Composite

A facial composite is a sketch or computer generated image used to create a visual representation of a suspect based on the memory and description of eyewitnesses. Composite sketches are frequently used to help track down suspects in a crime in which there is no photographic evidence. Though frequently



used in both investigative and prosecutorial areas of law, critics suggest that facial composite sketches and the newer computer-generated versions are extremely prone to misuse and error. If judges and juries rely heavily on facial composites to make decisions, critics say that innocent people are at risk of jail and other consequences as a result of mistaken identity. There are several methods of creating a facial composite. The most basic involves sketch artists, who are often professionals and have many years of training in both art and the craft of compositing. Sketch artists may speak in detail with eyewitnesses to help them recall details about how a suspect looked. The artist produces sketches that coincide with the description, sometimes asking the witness if the sketch needs to be altered in anyway. Hand sketching was once the primary means of creating a facial composite, but modern computer techniques are becoming much more popular in the 21st century.

Computer software for facial compositing allows the witness to choose images from a database to match his or her memory of each feature on the suspect's face. In addition to narrowing down details like eye color or nose shape, the witness can also position the features on a face model and modify the sizes and relative positions of each feature. Identifying marks, such as scars, tattoos, or piercings may be added from databases or drawn onto the model.

A facial composite may be used in several different stages of criminal investigation and justice procedures. Law enforcement agents may use a composite image to create a "wanted" poster, that warns citizens of a potential criminal in the area and asks for any tips or sightings. Additionally, detectives or investigative personnel may rely on composites when tracking down people connected to a crime or following leads. In cases where a suspect's identity cannot be found or no prior photographs exist, a composite may be the only way to jar a witnesses memory about the suspect. Police may also use composites to compare sketches with databases of known criminals, to see if they can find a likely match.

Some studies have shown that facial composites are extremely prone to error. Unfortunately, in a heated event such as a criminal situation, even the keenest eyewitness may be too overwhelmed by adrenaline or fear to catch many details. Additionally, eyewitnesses may be prone to suggestion, and may become convinced of a detail that did not actually exist. Nevertheless, despite criticisms, facial composite images remain a widely-used tool in investigation and prosecution.

3.1.4 Fashion Sketches

Fashion sketches are sketches done by designers when they want to turn a new clothing idea into an actual design. They start with a template of a body and draw the garment onto it. This allows the designer to market his ideas and designs and to showcase originality and style. There are two ways to present fashion sketches. Images can be created by using computer software or they can be drawn out freehand.

The template a designer uses is just as important as the garment itself. Templates are used as guides and the designer must find one that enhances the look of the garment. Hairstyles and different accessories might also be included onto the template to add a splash of color or to emphasize a style. Developing figure poses for a particular garment is also important when a designer chooses a template for fashion sketches.

The designer will have to include the texture and color of the material being used to create the garment. Fashion sketches have to be drawn in three-dimensional illustrations and this is done by shading in areas to show different textures. There are natural folds in clothing when it is worn on the body and the designer has to emphasize these folds so that the garment can be fully understood. When preparing fashion sketches the garment has to be presented with both the front and back views.



All zippers, buttons, and seams are usually included in most fashion sketches. Any visible seam lines are drawn out on the sketch in darker-colored dotted lines. Zippers and buttons are clearly drawn out to indicate where more material is needed and to see where there may be a material pull or puckering on the garment. This can eliminate any unexpected problems when the pattern is being constructed.

Fashion sketches are an important part of creating a garment. They clearly show the look the designer is trying to convey. The intended textures and colors are also documented as well as the how the clothing falls and gathers on the body. Preparing a pattern for the garment is the last step before it is actually made.

Occasionally, fashion sketches are not converted into garments if a designer is not happy with the results. They can still be used as guides and the designer may choose to slightly alter the areas that need changes. The designer may have to repeat the process several times to create an acceptable garment.

Clothing Design Software

Clothing design software is software that helps a user create clothing using digital tools. The types of users who may be interested in this software include those who are employed in the fashion industry or who own their own clothing businesses. An entire clothing line, from the planning to the production stages, can be created using fashion design software.

Using clothing design software can greatly benefit clothing designers by helping them retain design accuracy, reduce designing costs and streamline the design process so that their finished products can appear on the market as quickly as possible. This software can be used by professionals, students and amateurs alike. Even fashion design teachers may regularly depend on this software.

The features of clothing design software may vary based on a software's manufacturer. Most clothing design software is designed to help designers in the areas of fashion design, pattern and marker making and cutting and planning for production. In addition, some software may be able to be integrated with other proprietary software so that users can easily shift between two or more software programs. This is an especially useful feature for those who work collaboratively and want to transition between graphic design tools.

The average fashion design software allows designers to view their designs in either 2D or 3D. If a designer chooses to see his or her designs on models, he or she can elect to view pieces on models of varying proportions. Some software may even allow users to input the measurements of a specific person so that the designer is able to create a clothing piece specific to the person's body. These options let the designer envision his work in a manner that is as close to reality as possible.

One major benefit of using clothing design software is that it usually requires no skills in sewing or drawing. In fact, clothing design software often includes templates that novice users can use to create their own fashion designs. Training materials are also usually included in the software so that users may learn how to use a particular software's features.

Clothing design software often allows users easy access to their completed works or works that remain in progress. Sketches can be saved or printed out to be used in portfolios, catalogs, spec sheets, and line sheets. Many users of this software show completed projects to investors, buyers and customers.

3.2 THE ROLE OF SKETCHING IN THE DESIGN PROCESS

The role of sketching in digital art varies depending on if you are creating Web sites, identities, illustrations, product concepts, or other designs. An illustration or a logo is likely to need more sketching than a website.

As a tool or skill, sketching has its role in the design process. That role will vary depending on the end-product being created, the size and scope of the project, the individual designer's style, experience, and workflow, and the client's expectations. Find out more about how sketching is used in the design process within multiple design disciplines.

A large project with a significant client budget will benefit from sketching throughout the design process. This makes sure that before massive amounts of time are invested on refining a solution, a direction is first agreed upon with the client. Sketching can start loose, beginning with basic concepts. Then work on compositions or layouts. After those directions are chosen, the concepts can further be refined with detailed sketching.

Computers are being used by architects almost exclusively for the later stages of design: one UK practice has been quoted as using CAAD for upwards of 90% of their issued drawings while early presentation work and sketches are made with pencil and paper. Under the temporal, financial and aesthetic pressures of the design office any tool will be used only for tasks that designers perceive are within its capability. Whether their current perception is based in prejudice, preference or past experience, designers avoid using computers for sketching. The software currently available for sketch design is plainly not working.

3.2.1 Uses for Sketching in Design

The lack of useful software is not a hindrance to design. Buildings have been and are built without any need for information technology; design is famously as at home on an envelope, a cigarette packet or a cocktail napkin as it is on the drawing board or Mac-Paint. However, architects using CAAD to produce hard line drawings, but only pencil and paper to sketch, experience real problems when moving between these contexts.

This paper presents a small part of a PhD investigation into how to design interfaces that support sketch design. Our approach to the problem is to understand that sketch design and the computer as having some fundamental differences that need to be recognized before they can be overcome.

Remember

Electronic drawings must be printed out before they can be reworked. Alterations must then be copied tediously back into the computer. If the computer could support sketch design, the architect could work in either context, as they see fit.

We see the computer's role in sketch design not as some design companion but as presenting a fluid and engrossing sketching medium, stage managing the transition of drawings from the back-of-the-envelope to hard line drawings, renderings and models and back again.

There are multiple uses for sketching in the design process. Below is a review of five categories of uses with examples and links.

1. Rapid Concept Development

Sketching is an excellent way to quickly explore concepts. You can sketch for one or two hours and work out multiple possible solutions to the design problem at hand. This is an essential step in the design process. It will save you time to work through concepts on paper before going to the computer. While it is possible to build sketches on the computer, it's not as fast as sketching multiple concepts on paper.

Designer Karley Barrett shows us her vast use of rough sketches for logo design development. She explores over 60 possible solutions before narrowing the concepts down to just a handful of best ideas. It's interesting to see how she explores iconic imagery, typography, and layout.

She works through multiple ideas and searches for the best presentation of those ideas. Because she's making small sketches, she's able to work quickly and generate a multitude of ideas in a relatively short period of time.



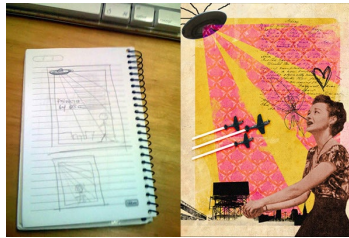
Product designers spend a lot of time sketching. If you are going to design the next sport shoe, piece of furniture, or bike, the idea does not start in a computer, it starts on paper.

James over at the blog Bicycle Design has this to say about sketching, "Putting ideas quickly on paper is the only way to evaluate them to see if they are worth exploring further. Computer renderings and modern CAD and modeling packages are great, but thinking on paper with a good old-fashioned pencil is always the place to start."

2. Basic Composition or Layout

Sketches are a quick way to create the basic composition of your illustration. They are also used in Web site design and **graphic design** to quickly evaluate layout choices. You can make a series of thumbnail sketches, or they can be larger. As long as your sketches are good enough that they capture the necessary elements, drawing skill is unnecessary.

How it's faster to do some sketches before going to the computer. As you can see below, he captures the basic composition on the left in a sketch. Compare the sketch to the final Photoshop image on the right. You can see the basic layout was worked out on paper. The image of the woman is represented by a stick figure in the drawing. It does not require amazing, or even good, drawing skills to work out composition before opening up Photoshop.



Web Design from Scratch is a well-known Web site that offers practical advice on building Web sites. The author has this to say about pencil sketching layouts: "The quick pencil sketch just helps me quickly record the likeness of what you have visualized in my head. Then you do not forget and can make it up quickly in Photoshop. You find this way of working a lot more efficient than starting off in Photoshop." As you can see below, drawing skill is not necessary to capture layout composition either. The left side below is the sketch, and the right side is the final design.



Keyword

Graphic design

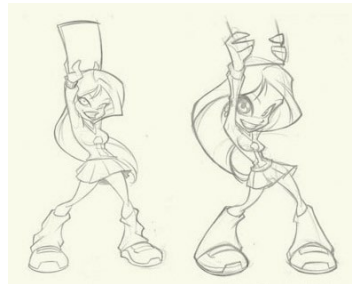
is the process of visual communication and problem-solving through the use of typography, photography and illustration.

Client Communication and Approval .3

Showing sketched thumbnails or compositions to clients, will potentially save you an enormous amount of time. If you are going to spend hours on an illustration, you want to make sure the client is in agreement with your choice of design before moving forward. Getting thumbnail approvals from clients is a common part of the illustration process. It is also common on large logo design projects and other projects as well.

The SOS Factory designs predominately mascot logos. Their workflow follows a methodology similar to a comic book design studio. The individual that sketches is often not the same as the one who does the line work. The designer, colorist, and art director are all different roles. They break each role apart into specialties.

At this studio, the sketcher works out concepts and client corrections with the art director and designer. The client approves artwork before it goes to the next stage of inking and coloring. This saves time by solidifying an idea before going on to more advanced stages in the process. The example below is a concept worked out based on initial client communication. This sketch is then sent to the customer for approval or for change requests. Once the sketch is finalized, the design moved to the next stage of inking the line work and then coloring the character.



They send a series of rough compositional sketches to the client before drawing a more detailed sketch. Below left you can see the one the client chose. Then on the right a more detailed sketch is done before moving to the computer.



Visual Exploration .4

Sketching can be used as a journaling activity to record and explore your interests. It can also be used to explore multiple options you could take in a particular design.

These sketches show her visual explorations in multiple fields of design. In the sketch area of her portfolio, she visually explores topics such as patterning, identities, and tattoo styles.



The product design book *Design Sketching* explains the entire process of sketching for product design. It offers tutorials, explanations, and examples.

Refining Visual Solutions .5

The process of creating a design or illustration at later stages involves refinement. The overall concept and direction of the piece may be working great, but one element is not. Often, this can be tightened up and corrected in further rounds of sketching. Of course, at some point a digital artist moves to the computer. The process of sketching then moves into digital drafts.

You get a feel for how important sketching was in this project, but also how seamlessly the artist moves to Photoshop. In some cases, the artist prefers digital solutions as further client corrections are requested. The artist decides which medium will get the job done faster as a deadline looms.



It's an excellent reference on this subject. Bill discusses refining illustrations before going to the computer. "Often times some aspect of the illustration looks bad. A professional artist will re-work that part of the illustration on a separate piece of paper until they get it right.

In this situation, the artist has identified the need to rework a part of the sketch. In some cases, it may be based on a client request, like with Angel D'Amico above. Regardless of the reason, you will ultimately want a tight sketch for detailed work. Below is a section of one of Bill's tight sketches. After that he brought the image into the computer to complete the process.



Keyword

Transparency is operating in such a way that it is easy for others to see what actions are performed.

3.2.2 Related Work

Existing work that is relevant to the transparent medium and sketching tool presented here can be divided into two broad categories: projects concerning issues of **transparency** and those investigating sketching. The amount of work dedicated to the latter far outweighs the former, and much sketching/painting research has little relevance to the proposals for a transparent interface. However, a few projects which tackle the sketching discipline have used aspects of transparency, and it is on these that we concentrate.

Sketch design software

The concentrates on gesture recognition to interpret sketches into structured diagrams. The marks made cannot be separated or recombined in new ways, nor do they lend themselves to re-interpretation or ambiguity. This shortcoming by giving the

user 'access to visual objects as they are perceived by the human visual system.' Finally, a realized building project, makes an interesting example of what architects expect from sketch design software. The process: 'an original cryptogram, overlaid three times, was converted into a model and then scanned straight into a computer using sophisticated software and transformed into plans "since we did not care too much about the functions of the space." The computer in Coop Himmelblau's office was then linked to a computer in Groningen's redundant dockyards where the gallery's huge steel panel components were welded together by the shipbuilders and then transported to the site for assembly.'

Uses of Transparency

Some important groundwork, showing how transparency and blur can be used to priorities information within a complex display and relating scales of each to visual impact. The 'desktop' GUI as an artistic artifact, drawing analogies between the use of perspective and transparency in painting and their application to interface design, noting that the latter confers supernatural and special status on objects in addition to allowing multiple images to overlay. Although this is a purely theoretical paper, it does contain a mocked up image of a transparent window which comes closest to the medium we describe here

Use transparency to allow multiple interface widgets to occupy the same screen space so as to free up the small displays of most hand-held PDA devices. Unfortunately for their experiment, they imagine their widgets as being above the information on display, yet render them as if they were below it. What they show is that transparency does not work when the user's perception of depth differs from that maintained by the interface. The user of transparent interface widgets in extensive detail in order to determine the optimum levels of transparency for these elements. Their intention is to prevent dialog boxes, menus and icons from taking up valuable screen space in graphic intensive applications such as 3D design rather than to exploit overlaid images.

Present a compelling use for such transparent interface widgets. Their Tool glass has been the foundation for a large number of papers which form the most extended investigation

Did You Know?

Sketching is also used as a form of communication in areas of product design such as industrial design. It can be used to communicate design intent and is most widely used in ideation. It can be used to map out floor plans of homes.



into Transparency undertaken so far, passing images up and user interaction down through a stack of tool glasses to allow the user to construct an interface out of the overlapping elements. Importantly, the documents that the user is working on are always shown as opaque. The transparent medium that we present here makes every window transparent, removing the distinction between interface tools and documents. They employ a combined overhead projection system and camera to set up a visual feedback loop similar to installation work. Their Digital Desk system overlays a digital desktop onto a real one, allowing hand drawn images to be instantly digitized and copied or used as the basis of digital transformations. Digital Desk is the most literal, and successful, expression of the transparent computer.

Finally implements a sketching environment from a stack of freeform transparent sheets of drawings and, as such, provides the closest parallel to our work on a transparent medium.

3.2.3 Sketching and Sketch Design

Studying sketching is fraught with potential confusions. The biggest problem is linguistic - the words sketch and drawing refer both to the act as well as its end product, the act of sketching is seen to be different from the image that it produces. To help distinguish these, sketch will be used to refer to drawings that are being made and image to the final result. The image is made up of marks and sketching can be seen as the act of mark making. Quite why these distinctions are important to sketch-design software becomes clear after digging a little deeper into the nature of sketching. Today design is inexorably entwined with some sort of drawing. Whether this is because drawing is a natural design medium, a historical accident or even a device to exclude the layman is not really an issue. What is more important is that there has been a tendency for theories of sketch design to take this relationship as read and concentrate solely on sketching as an act - looking at the marks made - and the images produced. Whilst the interrelationship between sketching and design is not in question, failing to assess this relationship leads to serious confusions in understanding just what is so important about sketching.

It is usually understood that the image is the design: perspectives, for example, are presented to both clients and planners as objective tests of a design's validity. This relationship between architectural drawing and design developed, in the mid seventeenth century, from a functional view of reality. The proposal that the world can be represented in an objective, self-consistent manner developed at this time underlies our contemporary appreciation of visual truth: images are to be judged by how well they describe the world. Outlines the two most prevalent explanations of how images are able to encode reality, the Correspondence and the Constructive theories, before going on to propose a theory based in Phenomenology.

Phenomenology sees everyday experience and action as activities of complete immersion, of being thoroughly engaged in doing. The intellectual stances of subject and object represent a breakdown of this engagement. Applying this theory to architectural representation, concludes that it is meaningless to search for information in sketches as the images only reveal design rather than encompass it. Architects can be surprised by their sketches and find new ideas in them, but this is not because the idea is actually in the image. Rather, the act of drawing helps to reveal the idea, revelation that is not possible without the image but entirely separate from it. They are reacting less to what they see than what they have thought about whilst doing.

3.2.4 Computer-based Sketching

Drawing is a pre-reflective, immersive state that is culturally based and culturally biased. Any drawing act is made as part of the established cultural context that allows us to create, understand, share and experience space without being able to represent it in its entirety. However, if the sketching experience is one of undifferentiated involvement, then any intervention into this, occasions when things suddenly intrude, times of breakdown, will halt the drawing prematurely.

If the phenomenological viewpoint is correct, the computer must be interfering with the pre-reflective immersive state that the sketching act requires. Although the computer tries to present an electronic version of your desktop, this desktop is subtly different in a number of important ways

- Every action is limited in some way. There are very few fundamental limits, things that the computer cannot do at all, but every application imposes restrictions on what the user can accomplish.
- A computer application is an encapsulation of its programmer's understanding of a task. It is not possible to rotate text in Microsoft Word, you cannot blur a drawing in a CAD system, and you cannot draw a continuous diagonal line in Photoshop. Whilst object-oriented software and integrated 'office' packages attempt to get around the problem it is important to recognize that the limitations still exist.

Remember

The decision to treat a drawing objectively must only be that of the sketcher and not one imposed by a data structure unable to support drawing or an interface that gets in the way. Sketching depends on lack of constraint and on fluidity.

- Whilst the modern GUI has helped users learn how to use applications more quickly by presenting their interfaces in the same way, this visual similarity unfortunately implies that the underlying applications function in the same way.
- Furthermore the interfaces get in the way, eating into the limited screen area available to the user. Even when most of the interface is placed into floating palettes a drawing can be almost completely obscured.
- Electronic documents are infinitely mutable there is no record of a drawing's evolution beyond what is explicitly recorded by the computer or user. Whilst this is a definite advantage for drafting, drawings are no longer definitive records.
- Most CAD applications use a selection and manipulation tool (usually represented by an arrow) as the default tool. After drawing an object, the current tool automatically switches to selection. This promotes editing and manipulation of a drawing over creating new marks the user is encouraged to switch from drawing to analysis of the drawing. Such applications naturally interrupt any immersive state and place emphasis on the product not the act.
- The tendency to slip into continuous editing can also seriously compromise the quality of work carried out electronically: an investigation by to find differences between paper and digital design shows that design students evaluated fewer alternatives at a basic level when computers were involved. Which a single image or set of images come to be the design, blinds the designer to any investigation other than that dealt with in these drawings. The make specific studies of this 'design fixation' and show how images given to test subjects can shape the way they subsequently design, even reproducing faults inherent in the examples. The cure lies in first realizing the iconic status of a drawing (usually an early concept statement) and setting it aside for new work to start. This is less easy to do in a digital medium that is seamlessly mutable and pitched toward editing existing documents rather than recreating them.
- Computer displays have an extremely low resolution when compared to that of paper and pencil.
- The mouse is a terrible drawing instrument, closer to a brick than a pencil. It was designed purely to give two dimensional positional information - for pointing. Its design restricts movement of the hand and arm to allow only this pointing movement. Its standard shape forces the hand and arm into a pointing gesture with forefingers extended to press buttons and the wrist locked in position, a straight line running from elbow to fingertip. It is at best extremely difficult to move a tool fluidly when in this position, let alone use of the fingers, wrist and elbow to vary the orientation and pressure with which that tool is used. Show that devices that take input only from the arm or hand instead of the fingers are inherently less flexible and accurate.



- Finally, modern graphic interfaces use opaque windows. Architects do not use opaque media very often – they use translucent and transparent media such as tracing paper and polyester film to copy, elaborate, annotate and evolve drawings. The only opaque drawings usually to be found in an architectural practice are mechanical reproductions.

SUMMARY

- Digital drawing can seem like an easier version of traditional sketching. A graphics tablet looks like a magic sheet of paper that removes all your mistakes.
- The term sketch pad is also used analogically for computer software applications that provide users with a place to sketch out ideas, use drawings to communicate, or create visuals to match concepts.
- Traditional sketching is a tactile experience that does not translate well to a mouse. But take the mouse out of the equation and replace it with a digital pen, and drawing on a computer becomes a surprisingly intuitive experience.
- Digital artwork is created through a layering process. Each layer refines the one below it, like tracing paper.
- The paper in a 3D drawing pad is made with a red and blue grid that is responsible for creating the 3D effect.
- Sketching is an excellent way to quickly explore concepts. You can sketch for one or two hours and work out multiple possible solutions to the design problem at hand.
- Electronic documents are infinitely mutable there is no record of a drawing's evolution beyond what is explicitly recorded by the computer or user.



KNOWLEDGE CHECK

1. **The following is not included in title block of drawing sheet.**
 - a. Sheet No
 - b. Scale
 - c. Method of Projection
 - d. Size of sheet
2. **Which of the following represent reducing scale?**
 - a. 1:1
 - b. 1:2
 - c. 2:1
 - d. 10:1
3. **In first angle projection method, object is assumed to be placed in**
 - a. First quadrant
 - b. Second quadrant
 - c. Third Quadrant
 - d. Fourth quadrant
4. **The following line is used for visible outlines**
 - a. Continuous thick
 - b. Continuous thin
 - c. Chain thin line
 - d. Short zigzag thin
5. **The following line is used for dimension line**
 - a. Continuous thick
 - b. Continuous thin
 - c. Chain thin line
 - d. Short zigzag thin
6. **A line drawn with a long section, short dash, and another long section is a _____.**
 - a. Hidden feature
 - b. Center of a circle
 - c. Center axis of a hidden cylinder
 - d. Center of a radius

7. A cylinder can be created by drawing a rectangular shape then the _____ tool.
 - a. Revolve
 - b. Sweep
 - c. Extrude
 - d. None of the above
8. The Offset tool should only be used for placing _____ in an isometric drawing.
 - a. Circles
 - b. Horizontal lines
 - c. Vertical lines
 - d. None of the above
9. The first step in creating a traditional technical drawing is to _____.
 - a. Draw a series of guide lines
 - b. Set up the meter line
 - c. Align the paper so that it will be positioned square to the parallel bar
 - d. Sharpen the leads in the technical pens
10. The default position of the UCS icon is positioned at _____ on the Auto-CAD grid.
 - a. 0, 0, 0
 - b. 10, 10, 10
 - c. 20, 20, 20
 - d. None of the above



REVIEW QUESTIONS

1. What do you understand by computer sketching?
2. How to sketch by computer?
3. What is a 3D drawing pad?
4. Discuss the fashion sketches and related work.
5. Focus on sketching and sketch design

Check Your Result

- | | | | | |
|--------|--------|--------|--------|---------|
| 1. (d) | 2. (b) | 3. (a) | 4. (a) | 5. (b) |
| 6. (c) | 7. (a) | 8. (c) | 9. (c) | 10. (a) |

REFERENCES

1. “Apps”. Sketch. Retrieved 19 August 2021.
2. “Sketch 1.0 finally released”. Bohemian Coding. 7 September 2010. Archived from the original on 11 July 2011. Retrieved 15 February 2016.
3. “Versioning, Licensing, and Sketch 4.0”. 8 June 2016. Retrieved 19 June 2016.
4. Digital Sketching: Computer-Aided Conceptual Designbooks.google.co.in › books, John Bacus · 2020
5. Engineering & Computer Graphics Workbook Using SOLIDWORKS 2016books.google.co.in › books, Ronald Barr, Davor Juricic, Thomas Krueger · 2016
6. Lowensohn, Josh (11 June 2012). “Apple announces 2012 Design Award winners”. CNET. Retrieved 15 February 2016.
7. Schoenmaker, Martijn (1 February 2016). “How I Started Using Sketch App In Windows”. Design + Sketch. Retrieved 10 October 2018.
8. Sketch-based Interfaces and Modelingbooks.google.com › books, Joaquim Jorge, Faramarz Samavati · 2010
9. Solutions, Fantastech (10 September 2018). “Converting Website Designs to Code”. Fantastech.co.
10. Sutton, Kelly (21 October 2014). “An Interview with Pieter Omvlee, the Founder of Bohemian Coding”. Retrieved 10 April 2019.
11. Weinberger, Matt (2 December 2015). “One of Apple’s most important initiatives is showing signs of failure”. Business Insider Australia. Archived from the original on 28 June 2016. Retrieved 15 March 2019.

CHAPTER 4

HTML/CSS/JAVASCRIPT

“Most developers will take advantage of its shortcuts and CSS selectors, but most of the time they fail to take advantage of much else. Being able to extend jQuery, whether by adding your own functions, CSS selectors or full-blown plugins, makes you a much stronger and smarter developer.”

– Robert Duchnik

LEARNING OBJECTIVES

After studying this chapter, you will be able to:

1. Discuss about hypertext markup language.
2. Learn about cascading style sheets (CSS)
3. Focus on inheritance
4. Explain the concept of javascript



INTRODUCTION

HTML is at the core of every web page, regardless the complexity of a site or number of technologies involved. It's an essential skill for any web professional. It's the

starting point for anyone learning how to create content for the web. HTML stands for HyperText Markup Language. “Markup language” means that, rather than using a programming language to perform functions, HTML uses tags to identify different types of content and the purposes they each serve to the webpage.

CSS stands for Cascading Style Sheets. This programming language dictates how the HTML elements of a website should actually appear on the frontend of the page.

HTML provides the raw tools needed to structure content on a website. CSS, on the other hand, helps to style this content so it appears to the user the way it was intended to be seen.

These languages are kept separate to ensure websites are built correctly before they’re reformatted.

JavaScript is a more complicated language than HTML or CSS, and it wasn’t released in beta form until 1995. Nowadays, JavaScript is supported by all modern web browsers and is used on almost every site on the web for more powerful and complex functionality.

JavaScript is a logic-based programming language that can be used to modify website content and make it behave in different ways in response to a user’s actions. Common uses for JavaScript include confirmation boxes, calls-to-action, and adding new identities to existing information.

4.1 HYPERTEXT MARKUP LANGUAGE

HTML (HyperText Markup Language) is the most basic building block of the Web. It defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page’s appearance/presentation (CSS) or functionality/behavior (JavaScript).

HyperText Markup Language (HTML) is a type of computer language that is used to create pages that can be posted on the Internet or sent via email. Although it might seem complex to many people, it is considered to be a relatively simple language. All text, graphics, and design elements a page designed with this language are “tagged” with codes that instruct the web browser or email program how to display those elements.

The tags also provide layout and formatting information so that the web page or email will look as close as possible to the way its designer intended it to look. For the novice website designer or anyone else who needs to know a certain code or wants to learn how to create an entire website, there are many software utilities, programs and websites that can assist him or her in writing HTML code.

- **Types of Codes.** There are many codes to allow for different text formatting, including italics, tables, paragraphs, and hyperlinks to web pages. Codes also

can indicate to the browser or email program how to display or use other elements, such as pictures, graphics, video, and sound. Other types of codes without angle brackets can be used to create punctuation marks, diacritics, and other symbols that might appear in text. Although all web browsers and many email programs use HTML, each might interpret and display the code a little differently, and designers often must consider these variations when creating a web page.

- ***An Evolving Language.*** Since the development of HTML in the early 1990s by British computer scientist Tim Berners-Lee, there have been many changes and versions. These versions have been maintained by the World Wide Web Consortium (W3C) since 1996. In January 2008, the First Public Working Draft of HTML 5 was published by the working group that was developing this specification. Still under development as of 2011, this revision was expected to dramatically change application development for the web. It introduces a number of new elements, including those for site structure, interactivity, and audio and video support, as well as new attributes.
- ***Viewing a Web Page's Code.*** The code used to create any web page can be seen by navigating a browser to the page, then choosing the correct option from the browser's menu. In most browsers, the user can click on the "view" menu and select an option such as "source," "view source" or "page source." This will cause a pop-up window to appear, and it will show the code that was used to create that web page.

It is important to note that not all of the content found on all web pages is written in HTML. Extensible Markup Language (XML) and Extensible HyperText Markup Language (XHTML) are other types of markup languages used in web development. In addition, style sheets — like Cascading Style Sheets (CSS) — are used to attach style to HTML documents. Languages like Flash® and Java® are used to create interactive content. Many other programming languages can be used to add specific functionality to a website.

- ***As Compared to HTM.*** Practically speaking, there's little difference between HTM and HTML extensions, since both are read as an HTML file by most machines. The reason there were two different extensions to begin with is that certain types of computers, like those that ran on 16 bit DOS or Windows® 3 systems, could not read four character extensions, and so needed the three character HTM extension. Most systems that can read four character extensions are automatically programmed to recognize HTM files as HTML files, though computer users may occasionally need a converter to change a file from HTM into a format that the system recognizes.

4.1.1 Basic HTML Document

HTML stands for Hypertext Markup Language, and it is the most widely used language to write Web Pages.

- Hypertext refers to the way in which Web pages (HTML documents) are linked together. Thus, the link available on a webpage is called Hypertext.
- As its name suggests, HTML is a Markup Language which means you use HTML to simply “mark-up” a text document with tags that tell a Web browser how to structure it to display.

Originally, HTML was developed with the intent of defining the structure of documents like headings, paragraphs, lists, and so forth to facilitate the sharing of scientific information between researchers.

Now, HTML is being widely used to format web pages with the help of different tags available in HTML language.

In its simplest form, following is an example of an HTML document –

```
<!DOCTYPE html>
<html>

  <head>
    <title>This is document title</title>
  </head>

  <body>
    <h1>This is a heading</h1>
    <p>Document content goes here.....</p>
  </body>

</html>
```

HTML Tags

As told earlier, HTML is a markup language and makes use of various tags to format the content. These tags are enclosed within angle braces <Tag Name>. Except few tags, most of the tags have their corresponding closing tags. For example, <html> has its closing tag </html> and <body> tag has its closing tag </body> tag etc.

Above example of HTML document uses the following tags –

Sr.No	Tag & Description
1	<code><!DOCTYPE...></code> This tag defines the document type and HTML version.
2	<code><html></code> This tag encloses the complete HTML document and mainly comprises of document header which is represented by <code><head>...</head></code> and document body which is represented by <code><body>...</body></code> tags.
3	<code><head></code> This tag represents the document's header which can keep other HTML tags like <code><title></code> , <code><link></code> etc.
4	<code><title></code> The <code><title></code> tag is used inside the <code><head></code> tag to mention the document title.
5	<code><body></code> This tag represents the document's body which keeps other HTML tags like <code><h1></code> , <code><div></code> , <code><p></code> etc.
6	<code><h1></code> This tag represents the heading.
7	<code><p></code> This tag represents a paragraph.

World Wide Web Consortium (W3C) recommends to use lowercase tags starting from HTML 4.

HTML Document Structure

A typical HTML document will have the following structure –

`<html>`

`<head>`

Document header related tags

`</head>`

Remember

To learn HTML, you will need to study various tags and understand how they behave, while formatting a textual document. Learning HTML is simple as users have to learn the usage of different tags in order to format the text or images to make a beautiful webpage.

Remember

To use such characters in an HTML document, they must be “escaped” by using a special code. All character codes take the form `&code;`, where code is a word or numeric code indicating the actual character that you want to put onscreen.

```
<body>
```

Document body related tags

```
</body>
```

```
</html>
```

Let's see what is document declaration tag.

The `<!DOCTYPE>` Declaration

The `<!DOCTYPE>` declaration tag is used by the web browser to understand the version of the HTML used in the document. Current version of HTML is 5 and it makes use of the following declaration –

```
<!DOCTYPE html>
```

There are many other declaration types which can be used in HTML document depending on what version of HTML is being used. We will see more details on this while discussing `<!DOCTYPE...>` tag along with other HTML tags.

HTML - Basic Tags

The `<html>` tag tells the browser that this is an HTML document. The `<html>` tag represents the root of an HTML document. The `<html>` tag is the container for all other HTML elements (except for the `<!DOCTYPE>` tag).

Heading Tags

Any document starts with a heading. You can use different sizes for your headings. HTML also has six levels of headings, which use the elements `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>`, and `<h6>`. While displaying any heading, browser adds one line before and one line after that heading.

Example:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>Heading Example</title>
```

```
</head>
```



```
<body>
  <h1>This is heading 1</h1>
  <h2>This is heading 2</h2>
  <h3>This is heading 3</h3>
  <h4>This is heading 4</h4>
  <h5>This is heading 5</h5>
  <h6>This is heading 6</h6>
</body>
```

```
</html>
```

This will produce the following result –

This is heading 1
This is heading 2
This is heading 3
This is heading 4
This is heading 5
This is heading 6

Paragraph Tag

The **<p>** tag offers a way to structure your text into different paragraphs. Each paragraph of text should go in between an opening **<p>** and a closing **</p>** tag as shown below in the example –

Example:

```
<!DOCTYPE html>
<html>
```

```
  <head>
    <title>Paragraph Example</title>
```

```
</head/>
```

```
  <body>
    <p>Here is a first paragraph of text.</p>
```

```

    <p>Here is a second paragraph of text.</p>
    <p>Here is a third paragraph of text.</p>
  </body>

```

```
</html>
```

This will produce the following result –

Here is a first paragraph of text.

Here is a second paragraph of text.

Here is a third paragraph of text.

Line Break Tag

Whenever you use the **
** element, anything following it starts from the next line. This tag is an example of an **empty** element, where you do not need opening and closing tags, as there is nothing to go in between them.

The **
** tag has a space between the characters **br** and the forward slash. If you omit this space, older browsers will have trouble rendering the line break, while if you miss the forward slash character and just use **
** it is not valid in XHTML.

Example:

```

<!DOCTYPE html>
<html>

```

```

  <head>
    <title>Line Break  Example</title>
  </head>

```

```

  <body>
    <p>Hello<br />
      You delivered your assignment ontime.<br />
      Thanks<br />
      Mahnaz</p>
  </body>

```

```
</html>
```

This will produce the following result –

Hello

You delivered your assignment on time.

Thanks

Mahnaz

Centering Content

You can use **<center>** tag to put any content in the center of the page or any table cell.

Example:

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    <title>Centring Content Example</title>
```

```
  </head>
```

```
  <body>
```

```
    <p>This text is not in the center.</p>
```

```
    <center>
```

```
      <p>This text is in the center.</p>
```

```
    </center>
```

```
  </body>
```

```
</html>
```

This will produce following result –

This text is not in the center.

This text is in the center.

Horizontal Lines

Horizontal lines are used to visually break-up sections of a document. The **<hr>** tag creates a line from the current position in the document to the right margin and breaks

the line accordingly.

For example, you may want to give a line between two paragraphs as in the given example below –

Example:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>Horizontal Line Example</title>
```

```
</head>
```

```
<body>
```

```
<p>This is paragraph one and should be on top</p>
```

```
<hr />
```

```
<p>This is paragraph two and should be at bottom</p>
```

```
</body>
```

```
</html>
```

This will produce the following result –

This is paragraph one and should be on top

This is paragraph two and should be at bottom

Again **<hr />** tag is an example of the **empty** element, where you do not need opening and closing tags, as there is nothing to go in between them.

The **<hr />** element has a space between the characters **hr** and the forward slash. If you omit this space, older browsers will have trouble rendering the **horizontal line**, while if you miss the forward slash character and just use **<hr>** it is not valid in XHTML

Keyword

Horizontal line is one which runs from left to right across the page.

Preserve Formatting

Sometimes, you want your text to follow the exact format of how it is written in the HTML document. In these cases, you can use the preformatted tag **<pre>**.

Any text between the opening **<pre>** tag and the closing **</pre>** tag will preserve the formatting of the source document.

Example:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>Preserve Formatting Example</title>
```

```
</head>
```

```
<body>
```

```
<pre>
```

```
function testFunction( strText ){
```

```
    alert (strText)
```

```
}
```

```
</pre>
```

```
</body>
```

```
</html/>
```

– This will produce the following result

```
function testFunction( strText ){
```

```
    alert (strText)
```

```
}
```

Try using the same code without keeping it inside **<pre>...</pre>** tags

Nonbreaking Spaces

Suppose you want to use the phrase “12 Angry Men.” Here, you would not want a browser to split the “12, Angry” and “Men” across two lines –

An example of this technique appears in the movie “12 Angry Men.”

In cases, where you do not want the client browser to break text, you should use a nonbreaking space entity ** **; instead of a normal space. For example, when coding the “12 Angry Men” in a paragraph, you should use something similar to the following code –

Example:

```
<!DOCTYPE html>  
<html>
```

```
  <head>  
    <title>Nonbreaking Spaces Example</title>  
  </head>
```

```
  <body>  
    <p>An example of this technique appears in the movie  
    "12&nbsp;Angry&nbsp;Men."</p>  
  </body>  
</html>
```

This will produce the following result –

An example of this technique appears in the movie "12 Angry Men."

4.1.2 HTML Page Structure

HTML coding is structured like a tree, with each different tag nested within it. In most cases, each formatting element requires a start tag and an end tag, and different tags should not overlap. This is what is meant by "nested;" if tag 2 opens after tag 1, then tag 2 should be closed first so that the formatting element of tag 2 is entirely enclosed within tag 1. Elements are the individual components that make up the code, and include opening and closing tags and the content between them. Attributes provide more information about the element, and are made up of the attribute and its value, connected by an equal sign.

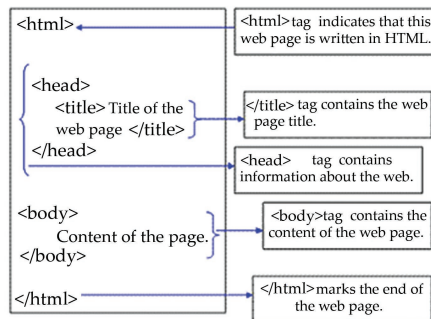
To create an HTML element, the user creates a tag that starts and finishes with angle brackets and places it before the text that needs to be formatted. The code usually one or more letters, numbers, words, and/or symbols — inside the angle brackets indicates what the element is and the attributes that content should have, such as its size, font, or other characteristics. To end the formatting, the user types the first angle bracket, then a backslash, then repeats the element code and closes the bracket. For example, `<title>What Is HTML?</title>` is the code used to format the title of this article; the "strong" element tag is nested within the "title" tag.

An HTML document has two* main parts:

- *head*. The head element contains title and meta data of a web document.
- *body*. The body element contains the information that you want to display on a web page.

* To make your web pages compatible with HTML 4, you need to add a document type declaration (DTD) before the HTML element. Many web authoring software add DTD and basic tags automatically when you create a new web page.

In a web page, the first tag (specifically, `<html>`) indicates the markup language that is being used for the document. The `<head>` tag contains information about the web page. Lastly, the content appears in the `<body>` tag. The following illustration provides a summary.



4.1.3 HTML `<!DOCTYPE>` Declaration

The `<!DOCTYPE>` declaration must be the very first thing in your HTML document, before the `<html>` tag.

The `<!DOCTYPE>` declaration is not an HTML tag; it is an instruction to the web browser about what version of HTML the page is written in.

In HTML 4.01, the `<!DOCTYPE>` declaration refers to a DTD, because HTML 4.01 was based on SGML. The DTD specifies the rules for the markup language, so that the browsers render the content correctly.

HTML5 is not based on SGML, and therefore does not require a reference to a DTD.

Example:

```

<!DOCTYPE html>
<html>
<head>
<title>Title of the document</title>
</head>

```

```
<body>
The content of the document.....
</body>

</html>
```

Common DOCTYPE Declarations

HTML 5

```
<!DOCTYPE html>
```

HTML 4.01 Strict

This DTD contains all HTML elements and attributes, but does NOT INCLUDE presentational or deprecated elements (like font). Framesets are not allowed.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
```

HTML 4.01 Transitional

This DTD contains all HTML elements and attributes, INCLUDING presentational and deprecated elements (like font). Framesets are not allowed.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

HTML 4.01 Frameset

This DTD is equal to HTML 4.01 Transitional, but allows the use of frameset content.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">
```

XHTML 1.0 Strict

This DTD contains all HTML elements and attributes, but does NOT INCLUDE presentational or deprecated elements (like font). Framesets are not allowed. The markup must also be written as well-formed XML.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

XHTML 1.0 Transitional

This DTD contains all HTML elements and attributes, INCLUDING presentational and deprecated elements (like font). Framesets are not allowed. The markup must also be written as well-formed XML.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

XHTML 1.0 Frameset

This DTD is equal to XHTML 1.0 Transitional, but allows the use of frameset content.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

XHTML 1.1

This DTD is equal to XHTML 1.0 Strict, but allows you to add modules (for example to provide Ruby support for East-Asian languages).

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

4.1.4 HTML Version

HTML defines the structure and layout of a Web document by using a variety of tags and attributes. The correct structure for an HTML document starts with <HTML><HEAD>(enter here what document is about)<BODY> and ends with </BODY></HTML>. All the information you'd like to include in your Web page fits in between the <BODY> and </BODY> tags.

HTML is the standard markup language for creating Web pages.

- HTML stands for Hyper Text Markup Language
- HTML describes the structure of Web pages using markup
- HTML elements are the building blocks of HTML pages
- HTML elements are represented by tags
- HTML tags label pieces of content such as "heading", "paragraph", "table", and so on
- Browsers do not display the HTML tags, but use them to render the content of the page

Since the early days of the web, there have been many versions of HTML:

Version	Year
HTML	1991
HTML 2.0	1995
HTML 3.2	1997
HTML 4.01	1999
XHTML	2000
HTML5	2014

4.1.5 HyperText Markup Language (1991)

The HyperText Markup Language (HTML) is a simple data format used to create hypertext documents that are portable from one platform to another. HTML documents are SGML documents with generic semantics that are appropriate for representing information from a wide range of domains.

As HTML is an application of SGML, this specification assumes a working knowledge of [SGML].

Scope: HTML has been in use by the **World-Wide Web (WWW)** global information initiative since 1990. Previously, informal documentation on HTML has been available from a number of sources on the Internet. This specification brings together, clarifies, and formalizes a set of features that roughly corresponds to the capabilities of HTML in common use prior to June 1994. A number of new features to HTML are being proposed and experimented in the Internet community.

This document thus defines a HTML 2.0 (to distinguish it from the previous informal specifications). Future (generally upwardly compatible) versions of HTML with new features will be released with higher version numbers. HTML is an application of ISO Standard 8879:1986 Information Processing Text and Office Systems; Standard Generalized Markup Language (SGML). The HTML Document Type Definition (DTD) is a formal definition of the HTML syntax in terms of SGML.

This specification also defines HTML as an Internet Media Type[IMEDIA] and MIME Content Type[MIME] called `text/html`. As such, it defines the semantics of the HTML syntax and how that syntax should be interpreted by user agents.

Conformance: This specification governs the syntax of HTML documents and aspects of the behavior of HTML user agents.

Documents: A document is a conforming HTML document if:

- It is a conforming SGML document, and it conforms to the HTML DTD.
- It conforms to the application conventions in this specification. For example, the value of the *HREF* attribute of the *A* element must conform to the URI syntax.
- Its document character set includes [ISO-8859-1] and agrees with [ISO-10646]; that is, each code position listed in section The HTML Coded Character Set is included, and each code position in the document character set is mapped to the same character as [ISO-10646] designates for that code position. (2)

Feature Test Entities: The HTML DTD defines a standard HTML document type and several variations, by way of feature test entities. Feature test entities are declarations in the HTML DTD that control the inclusion or exclusion of portions of the DTD.

HTML.Recommended. Certain features of the language are necessary for compatibility with widespread usage, but they may compromise the structural integrity of a document. This feature test entity selects a more prescriptive document type definition that eliminates those features. It is set to 'IGNORE' by default.

In order to preserve the structure of a document, an editing user agent may translate HTML documents to the recommended subset, or it may require that the documents be in the recommended subset for import.



HTML.Deprecated. Certain features of the language are necessary for compatibility with earlier versions of the specification, but they tend to be used and implemented inconsistently, and their use is deprecated. This feature test entity enables a document type definition that allows these features. It is set to 'INCLUDE' by default. Documents generated by translation software or editing software should not contain deprecated idioms.

User Agents: An HTML user agent conforms to this specification if:

- It parses the characters of an HTML document into data characters and markup according to [SGML]. (3)
- It supports the 'ISO-8859-1' character encoding scheme and processes each character in the ISO Latin Alphabet No. 1 as specified in section The HTML Document Character Set. (4)
- It behaves identically for documents whose parsed token sequences are identical. For example, comments and the whitespace in tags disappear during tokenization, and hence they do not influence the behavior of conforming user agents.
- It allows the user to traverse (or at least attempt to traverse, resources permitting) all hyperlinks from *A* elements in an HTML document.

An HTML user agent is a level 2 user agent if, additionally:

- It allows the user to express all form field values specified in an HTML document and to (attempt to) submit the values as requests to information services.

4.1.6 HTML 2.0

HTML 2.0 was an expansion of the HTML language. Unlike the original version of HTML, HTML 2.0 was created to be a Web standard, “formalizing a set of features that roughly corresponds to the capabilities of HTML in common use”.

Even though the original version of HTML was never referred to as HTML 1.0, HTML 2.0 was given the 2.0 suffix to “distinguish it from previous informal specifications.

The documentation related to HTML 2.0 can be found here through the W3C.

Information on HTML was originally only available from scattered, varying resources. With the Web growing in size, it was important to have a strict set of standards outlining exactly what HTML should be able to do. It was even more important to have this information available in one place, so that users could learn exactly how the language worked. The first official standards for HTML 2.0 were published in November of 1995. HTML 2.0 was created through the Internet Engineering Task Force (IETF), though Tim Berners-Lee still took an active role in its creation.

How was it created?

HTML 2.0 was standardized as a RFC, or “request for comment” project in a memo released by Tim Berners-Lee with the In’s Network Working Group. The memo can be accessed [here](#). It was clearly a work in progress; its status as a “request for comment” project meant that programmers could submit feedback for consideration in new

updates. HTML 2.0 was first created to standardize the features users were already utilizing HTML to perform. However, programmers wanted new features as well.

HTML 2.0 underwent several updates during 1995, prior the release of the memo that set the final standardization to refine it. If you are interested, you can view a list of the updates [here](#).

Throughout the lifespan of the program, new features based on programmer feedback were added to the standards as RFC memos. These updates happened in November of 1995, May of 1996, August of 1996, and January of 1997.

4.1.7 HTML 3.2

HTML 3.2 was the first version of HTML to be released as a W3C Recommendation, and implemented most, but not all, of the features that were proposed in 3.0. Most notably, the mathematical equations were removed from 3.2. However, HTML 3.2 also included features that individual internet browsers were already using individually. HTML 3.2 was released on January 14th, 1997.

HTML 3.2 was the first set of standards for HTML to be released solely by the W3C. However, while the W3C codified the standards as official recommendations, the organization was not alone in developing said standards. Private companies, such as Netscape, IBM, Microsoft, Novell, among others, realized the importance of a unified standard, and took interest in the development of HTML 3.2 and contributed to the process.

Dave Raggett of Hewlett-Packard “Called together representatives of the browser companies” to form a development group for the language. The small number of people involved guaranteed that the group could make quick, efficient, educated decisions on the standards for 3.2. The representatives included “Lou Montulli from Netscape, Charlie Kindel from Microsoft, Eric Sink from Spyglass, [and] Wayne Gramlich from Sun Microsystems.” Tim Berners-Lee and Dan Connolly served as representatives of the W3 Consortium. Together, the seven discussed and debated the standards, eventually coming to a consensus.

HTML as an SGML Application

HTML 3.2 is an SGML application conforming to International Standard ISO 8879 -- Standard Generalized Markup Language. As an SGML application, the syntax of conforming HTML 3.2 documents is defined by the combination of the SGML declaration and the document type definition (DTD). This specification defines the intended interpretation of HTML 3.2 elements, and places further constraints on the permitted syntax which are otherwise inexpressible in the DTD.

The SGML rules for record boundaries are tricky. In particular, a record end immediately following a start tag should be discarded. For example:

<P>

Text

is equivalent to:

<P>Text

Similarly, a record end immediately preceding an end tag should be discarded.
For example:

Text

</P>

is equivalent to:

Text</P>

Except within literal text (e.g. the PRE element), HTML treats contiguous sequences of white space characters as being equivalent to a single space character (ASCII decimal 32). Note that future revisions to HTML may allow for the interpretation of the horizontal tab character (ASCII decimal 9) with respect to a tab rule defined by an associated style sheet.

SGML entities in PCDATA content or in CDATA attributes are expanded by the parser, e.g. é is expanded to the ISO Latin-1 character decimal 233 (a lower case letter e with an acute accent). This could also have been written as a named character entity, e.g. é. The & character can be included in its own right using the named character entity &.

HTML allows CDATA attributes to be unquoted provided the attribute value contains only letters (a to z and A to Z), digits (0 to 9), hyphens (ASCII decimal 45) or, periods (ASCII decimal 46). Attribute values can be quoted using double or single quote marks (ASCII decimal 34 and 39 respectively). Single quote marks can be included within the attribute value when the value is delimited by double quote marks, and vice versa.

Note that some user agents require attribute minimization for the following attributes: COMPACT, ISMAP, CHECKED, NOWRAP, NOSHADE and NOHREF. These user agents do not accept syntax such as COMPACT=COMPACT or ISMAP=ISMAP although this is legitimate according to the HTML 3.2 DTD

The SGML declaration and the DTD for use with HTML 3.2 are given in appendices. Further guidelines for parsing HTML are given in WD-html-lex

The Structure of HTML documents

HTML 3.2 Documents start with a <!DOCTYPE> declaration followed by an HTML :element containing a HEAD and then a BODY element

<"DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN!>

```
<HTML>
<HEAD>
<TITLE>A study of population dynamics</TITLE>
... other head elements
</HEAD>
<BODY>
... document body
</BODY>
</HTML>
```

In practice, the HTML, HEAD and BODY start and end tags can be omitted from the markup as these can be inferred in all cases by parsers conforming to the HTML 3.2 DTD.

Every conforming HTML 3.2 document **must** start with the `<!DOCTYPE>` declaration that is needed to distinguish HTML 3.2 documents from other versions of HTML. The HTML specification is not concerned with storage entities. As a result, it is not required that the document type declaration reside in the same storage entity (i.e. file). A Web site may choose to dynamically prepend HTML files with the document type declaration if it is known that all such HTML files conform to the HTML 3.2 specification.

Every HTML 3.2 document must also include the descriptive title element. A minimal HTML 3.2 document thus looks like:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<TITLE>A study of population dynamics</TITLE>
```

The HEAD element

This contains the document head, but you can always omit both the start and end tags for HEAD. The contents of the document head is an unordered collection of the following elements:

- The TITLE element
- The STYLE element
- The SCRIPT element
- The ISINDEX element
- The BASE element
- The META element
- The LINK element

```
<!ENTITY % head.content "TITLE & ISINDEX? & BASE?">
```

```
<!ENTITY % head.misc "SCRIPT|STYLE|META|LINK">
```

```
<!ELEMENT HEAD O O (%head.content) +(%head.misc)>
```

The %head.misc entity is used to allow the associated elements to occur multiple times at arbitrary positions within the HEAD. The following elements can be part of the document head:

TITLE defines the document title, and is always needed.

ISINDEX for simple keyword searches, see PROMPT attribute.

BASE defines base URL for resolving relative URLs.

SCRIPT reserved for future use with scripting languages.

STYLE reserved for future use with style sheets.

META used to supply meta info as name/value pairs.

LINK used to define relationships with other documents.

TITLE, SCRIPT and STYLE are containers and require both start and end tags. The other elements are not containers so that end tags are forbidden. Note that conforming browsers would not render the contents of SCRIPT and STYLE elements.

Remember

HTML 4.01 is the accepted standard, and the majority of web users do have browsers that support it fully. Some of the more peripheral new elements have yet to gain full support in the latest round of browsers, but they are on the way. Modern browsers will generally have no problem with anything in these specs.

4.1.8 HTML 4.01

HTML 4.01 was a large shake-up to the HTML standards that arrived in April 1998. The HTML language you have learnt is constantly evolving to meet the needs of a growing Internet. Things get added, some things get taken away and still more elements are asked to fade out gracefully. These changes ensure that designers have the freedom and power available to create increasingly complex websites and are able to achieve this efficiently.

It only happens every few years, and the changes are made by the » World Wide Web Consortium (W3C), who are HTML's governing body, as it were. They convene and design the specifications that we all work with when creating websites (CSS was designed by the W3C too). They look for weaknesses in HTML that are holding the web back, and sort them out, which makes creating compelling websites easier for

everybody. The standard we were all working with before this was HTML 3.2. That was used for a while before the W3C decided to step it up another notch a few years ago. They released HTML 4. Sometime later, when some minor errors in the specification were uncovered, they fixed these and called the final specification HTML 4.01.

Versions

If you have used any software you will have undoubtedly noticed how every few months it advances its number. Until they improved it and it became Firefox 2.1. Adding a decimal to the version number signifies a minor change to the original. When major changes are made to a software project, they will move up a whole number to version 3. This is the same way most dynamic things work. As you can see, the original HTML 3.0 spec was revised to version 3.2 before the big change to 4, and a minor change to 4.01. There was some confusion when HTML 4 started being discussed, as at the time version 4 browsers like Internet Explorer 4 were making their appearance and people thought there was some connection. In reality, the two separate things had just reached those versions simultaneously, not because of each other. As you know, browser technology has advanced to version 7 stages and beyond by now, and HTML is still at level 4. So there's no real connection; though, that said, it was in the version 4 browsers that HTML 4 started being incorporated properly. Glad that's cleared up. A few months after HTML 4.0 was released, its documentation was updated to correct some minor problems, and its version number was bumped up slightly. So the final version of this standard is HTML 4.01.

DOCTYPEs Ahoy

Nowadays the Document Type Declaration (DTD) at the top of your document is very important if you want the browsers to render your page correctly. Without it, browsers might interpret your code more loosely, and you may have display errors. The HTML 4.01 DTDs are below. Take your pick.

Use the strict DTD if you are using pure, structural code with no hacks:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
```

The transitional DOCTYPE, below, is the most commonly used, and still permits you to use certain old elements that we will eventually stop using altogether. It is probably the best choice until you have gotten to know HTML really well. Once you are ready, you can start using the stricter DOCTYPE above.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

Finally, for **frameset** pages, use the frameset DTD:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">
```

Simply add this line of code to the very top of your HTML pages (before the opening `<html>` tag), and you are away. You will also need to specify the character encoding of your page. The best encoding to use is called Unicode, and allows you to type almost any character you want (like punctuation, letters with accents etc.) directly into your content. Add this element in between your page's `<head>` and `</head>`:

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
```

Once you have added your DOCTYPE and encoding, run your page through the » HTML validator to see if you are obeying the rules.

The New Elements in HTML 4

There are 22 elements new to HTML in the 4 specifications, and they cover all the areas, from text-formatting to tables to frames and the rest. Most of the text formatting elements will make the text they mark up look a little different. You can see the effects .of these elements in the text formatting list

`<abbr>`

This is used to show an ABBReviated version of a word, and to offer the full version. When a reader leaves their mouse on the word, the full version pops up.

The code would be `<abbr title="abbreviation">abbr.</abbr>`

`<acronym>`

Similar to the one above, this is used for special abbreviations called acronyms (initialled abbreviations that can be spoken as a word themselves). It works the same way. `<acronym title="North Atlantic Treaty Organisation">NATO</acronym>`

`<bdo>`

Most text is read from left to right, but some languages are the opposite, like Hebrew for example. This new tag allows the browser to display your page correctly if you use one of these languages, and allows you to pull off cool effects like this: cool effects. That's typed in normally in the source code, but the tag changes its direction. RTL means 'Right To Left'. Code: `<bdo dir="rtl">crazy text</bdo>`

`<button>`

This is a simple way of adding form buttons to your pages. What's more, you can now format the text and put images and other elements on the button. `<button>click me</button>` click me for nothing!

`<colgroup>`

A table tag that allows you to affect the attributes of an entire column with one line of code. More info in [Tables Accessibility](#)

``

Wrapping this around some text creates a strike-through effect to signify DELETED text, so you can show readers what once existed without cutting it out altogether.
`waffle`

`<fieldset>`

This allows you to group buttons and things together, giving you a framed container to hold them in. It works together with the `<legend>` tag below.

`<frame>`

Strangely, this has only been made part of the official specifications now, despite it having been in use for years. It has been given new style attributes but overall it's the same as the frame tag before.

`<frameset>`

This tag is in the same boat as its friend above. Nothing's really new.

`<iframe>`

Once a proprietary Internet Explorer tag, this was such a smart idea that it has been assimilated into HTML proper.

`<ins>`

This stands for INSerted text. It works in conjunction with the `del` tag, and the inserted text appears with an underline. `waffle<ins>quality literature</ins>`

`<label>`

This allows you to give form elements a label. Clicking the label functions like clicking the element (a radio button, for example) itself. This improves Forms Accessibility.

`<label for="choice1">Choice 1</label>`

`<input type="radio" id="choice1">`

<legend>

When using a fieldset element, this element must come first before any other content inside the fieldset. It gives the title of the group.

<fieldset><legend>Contact Info</legend>

Email:*<input type="text">*

Address:*<input type="text"></fieldset>*

<noframes>

Another part of the frames umbrella that is being formally added to HTML 4. You can learn more about it at the basic frames page.

<noscript>

The same as above, this is for people who can't do JavaScript.

<object>

This is set to become the do-it-all tag for inserting multimedia into your page, and is supposed to take over from img, ismap, applet, script and any others.

<object data="picture.gif" type="image/gif"></object>

<optgroup>

With this tag you can group together many option elements which are part of a select field, and give the groups titles.

<param>

This tag is used to set PARAMeters for ActiveX, Applets and objects. It existed before HTML 4.01, but now is official code.

**

This tag was brought in specifically to work with stylesheets in applying classes and ids. It does nothing on its own, but it is great for applying your styles to text.

<tbody>

A new table tag that allows you to give attributes to a block of cells with this one tag.

`<tfoot>`

Allows you to add a footer to the `tbody` part of your tables.

`<thead>`

This allows you to add a header to the `tbody` part of a table. It comes before `tbody`, while `tfoot` comes after in the code. Both of these elements are also found in HTML4 Tables.

`<q>`

If you have ever used the `blockquote` tag, you will know it's a big tag. How many letters are in that, ten?! This is much more like it, and is suitable for shorter quotations. Plus, it adds in the quotation marks for you. It will not add in the line breaks you get with `blockquote`.

The new Attributes

These new attributes are here to allow **stylesheet** implementation, with two more reflecting the new international concerns that the W3C have taken onboard in this new draught. They can all be applied to any element.

class

This is how you give your page elements and text their classes from your stylesheet. Read all about it in advanced CSS.

dir

This is the attribute that is used mostly with the new `<bdo>` tag above. Your possible values are `rtl` (right-to-left) or the default `ltr` (left-to-right)

id

ids are just like classes, but can be used with JavaScript and DHTML. More info

lang

This attribute sets off a block of text as text typed in a foreign LANGUAGE, so that search engines and browsers know, and don't just take it as badly spelled English. It will not translate anything for you, it's just some behind-the-scenes help for things other than readers.

You can denote the text using the `span` tag, like

```
<span lang="fr">Bonjour!</span>.
```

If you are going to use it, have a look at the common language codes.

title

This is one of my favorite things that came with HTML 4.01. It allows you to add in tooltip text, like the alt attribute; but now you can add it to absolutely anything. You can give table cells titles, add in extra information to your links, and even hide jokes in your code that will only appear when a reader is on a specific word or sentence in your text.

Deprecated Elements

A deprecated element is one that is on the way out, but one which has been given a few more months to live before its life fully ends. There are much better elements than these available now, so your usage of them should be downscaled as much as possible.

Keyword

Style Sheet

is a collection of style rules that tells a browser how the various styles are to be applied to the HTML tags to present the document.

<applet>

Used to add Java applets. Use the new `<object>` element instead.

<basefont>

Used to affect text on the whole page. Use style sheets instead.

<center>

Used to center elements. Use `<div align="center">` or stylesheets.

<dir>

Used to make lists. Use `uls` instead.

**

Ah, the classic font element. Still good for small things, but stylesheets have taken over. This is one element you should really try to avoid using.

<isindex>

Just use the input tag.

<menu>

Another type of list that is redundant thanks to the `ul` element.



`<s>`

Creates strike-through effects. Use the stylesheets again, or the new delelement.

`<strike>`

Same as above, use style.

`<u>`

The underlining element, use stylesheets or ins instead.

Dead Elements

These are the elements that were so useless that they are out on their asses for good. Never use these, you cannot guarantee the browsers will continue to support them. All three of these elements have been replaced by one new element — so you can see how they were useless.

`<listing>`, `<plaintext>`, and `<xmp>`. Use `pre` instead. This creates PRE formatted text (text which follows its layout in your code).

And that's the lot. By this stage, you are encouraged to use all of these elements. Most people have either upgraded their browsers or bought computers with the newest software on it, so the majority of the web audience are on the same level. That said, there are some elements here that still are not supported in even the newest browsers, so test your pages out before you go using any of these elements.

4.1.9 XHTML

The extensible hypertext markup language (XHTML) is a quick way to refer to several language recommendations that are widely used on Internet-enabled devices for viewing web pages. Although named after its predecessor, the hypertext markup language (HTML), it's actually based on the extensible markup language (XML), which is a very selective part of the standard generalized markup language (SGML). In essence, they are all offspring of SGML. While HTML is a direct application of SGML, XHTML is what's referred to as a namespace, or a set of definitions for an XML document that helps to relieve ambiguity when more than one XML vocabulary is being used in any given situation.

The language came about because of a few limitations to HTML and the varied way HTML was being implemented. Around the time HTML made it to version four, it began to wane in proper usage by many HTML interpreters, the computer programs that parse HTML documents into a formatted, viewable web page. As mobile devices and other web-viewing platforms were also emerging, a better solution was needed. XML is a much more strict implementation of SGML over HTML, and different XML namespaces can be used in a single instance. So around the year 2000, the World Wide Web Consortium (W3C) drafted and made XHTML one of its recommendations

to solve some of these emerging issues. For all intents and purposes, XHTML mimics HTML in most ways, but since the former uses an XML namespace, it can be parsed by any XML interpreter, while HTML is limited to only HTML interpreters. XHTML is really HTML recreated under the more restrictive XML subset of SGML. In this way, the more recent language was immediately able to be interpreted by existing web browsers while also making itself available for other platforms. Living up to the extensible aspect of XHTML's moniker is also important to note. It not only offers the ability to be read by more programs and platforms, but it is also further extensible by allowing the use of other XML namespaces within its documents.

With XHTML's ability to include other XML namespaces in a document, it can be extended in a number of ways to present more than just page formatting. The mathematical markup language (MathML), for example, can be included in these documents to display mathematical formulas and notation. Images can also be embedded using the scalable vector graphics (SVG) namespace within a document of this type. As such, XHTML can also be included with another XML document.

Since XHTML is really just HTML refined under XML's rules, it offers three document type definitions (DTD) that duplicate those of HTML version four. A DTD is a detailed description of the elements of a markup language, including when, where, and how it can be used, as well as any associated attributes. In later versions of XHTML, however, XML schemas, another, more robust way of describing an XML document, were established that further augmented XHTML. In turn, various stripped-down versions of XHTML were developed that can then be built upon for specific uses, many of which revolve around mobile computing platforms.

Convert HTML to XHTML

The eXtensible HyperText Markup Language (XHTML) has replaced the similar HyperText Markup Language (HTML) as the markup language of choice for creating webpages, but old webpages are relatively easy to convert from HTML to XHTML. First, choose the appropriate version of XHTML to convert to because different versions support different features. Then make changes to the code by hand to better understand the new language's syntax rules or use a software tool to convert pages for you. In either case, it is a good idea to check that the document was converted properly with an XHTML validator, which can detect any errors in the code.

Keep in mind that just like HTML, XHTML has several different versions, and not all of them support the same features. XHTML 1.0 Transitional allows certain presentational tags, elements, and attributes present in earlier HTML standards including the `` tag and "center" attributes. If you are moving from HTML to XHTML 1.0 Strict or XHTML 1.1, remove any presentational tags or elements and recreate the desired look using Cascading Style Sheets because these versions of XHTML do not support any presentational elements.

For a simple web document, it's possible to perform an HTML to XHTML conversion by hand. If you are new to XHTML, this may also help you become more familiar with the stricter syntax rules of the language. To conform to these stricter requirements, make sure that all tags are in lower case, properly nested, and closed with a forward slash. In addition, all values must be inside quotation marks. The “find and replace all” option available in most text editors and web development programs can help you make these changes.

Software can also convert your code from HTML to XHTML with just a few clicks. Most professional web design programs have this option, often under the “File” or “Tools” menu. A popular tool called HTML Tidy, originally released by members of the World Wide Web Consortium (W3C), can be downloaded for free and will both clean up and convert between popular web formats. You might also be able to find a few tools online that can change documents from one markup language to another.

Whether you choose to edit your documents by hand or use software to perform the HTML to XHTML conversion, there are a few things you should check before publishing your pages to the web. To be valid XHTML, a page should have both a Document Type Definition (DOCTYPE or DTD) and an XML namespace declaration. Each of these will depend on what version of XHTML you use, so do a little research to figure out what these elements should look like. An XHTML validator, available within most web development software as well as on the W3C's website, will inform you of any errors or omissions in your code.

XHTML Editor

HTML is HyperText Markup Language, a language derived from SGML (Standard Generalized Markup Language) and used on the World Wide Web to create structure for the presentation of documents and create links between them, using tags and a set of rules. XML is EXtensible Markup Language and simplified SGML. XHTML is EXtensible HTML, a markup language that combines features of HTML and XML, with stricter rules than those applied in HTML coding. An XHTML editor is an environment designed for authoring XHTML.

Keyword

Cascading Style Sheets (CSS)

is a simple mechanism for adding style (e.g., fonts, colors, spacing) to Web documents. These pages contain information on how to learn and use CSS and on available software. They also contain news from the CSS working group.

An XHTML editor is not usually found as a stand-alone product. It is quite common to find an XHTML editor incorporated in a product that provides a more comprehensive package of web development editors. For example, a software package might combine an XHTML editor with an HTML editor. A more complete package could include an HTML editor, XHTML editor, CSS (**Cascading Style Sheets**) editor, and JavaScript® editor.

XHTML editors often have both a text editor in which to input XHTML and a WYSIWYG (What You See Is What You Get) editor, and sometimes it's possible to show only the WYSIWYG window. The purpose is to allow a user who has not learned the technical aspects of web design, such as markup languages and CSS, to still be able to generate and edit content. However, one problem with web pages and websites being built with correct XHTML has been WYSIWYG editors that do not generate proper markup. XStandard points out the problem and makes XHTML WYSIWYG Editor™ to address this, assuring that only markup that meets best practice criteria is used.

One important feature you may find in an XHTML editor is the ability to choose the version of XHTML you are working on. Adobe Dreamweaver CS4, for example, offers a choice of XHTML 1.0 Transitional, XHTML 1.0 Strict, XHTML 1.1, or XHTML 1.0 Mobile. HTML 4.01 Transitional and HTML 4.01 Strict are also offered.

In any case, there are some other features that may be present in an XHTML editor. One valuable feature that is not universally offered is XHTML validation. Alternatively, dynamic correction may be offered along with an error log file. An HTML to XHTML conversion feature may also be included. Other desirable features include syntax highlighting, word wrap, and UTF-8 Unicode (8-bit UCS/Unicode Transformation Format) — UCS stands for Universal Character Set — to enable work on multilingual files.

Syntax

XHTML syntax is very similar to HTML syntax and almost all the valid HTML elements are valid in XHTML as well. But when you write an XHTML document, you need to pay a bit extra attention to make your HTML document compliant to XHTML.

Here are the important points to remember while writing a new XHTML document or converting existing HTML document into XHTML document –

- Write a DOCTYPE declaration at the start of the XHTML document.
- Write all XHTML tags and attributes in lower case only.
- Close all XHTML tags properly.
- Nest all the tags properly.
- Quote all the attribute values.
- Forbid Attribute minimization.

- Replace the name attribute with the id attribute.
- Deprecate the language attribute of the script tag.

Here is the detail explanation of the above XHTML rules –

DOCTYPE Declaration

All XHTML documents must have a DOCTYPE declaration at the start. There are three types of DOCTYPE declarations. Here is an example of using DOCTYPE –

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Case Sensitivity

XHTML is case sensitive markup language. All the XHTML tags and attributes need to be written in lower case only.

```
<!-- This is invalid in XHTML -->
<A Href="/xhtml/xhtml_tutorial.html">XHTML Tutorial</A>
```

```
<!-- Correct XHTML way of writing this is as follows -->
<a href="/xhtml/xhtml_tutorial.html">XHTML Tutorial</a>
```

In the example, **Href** and anchor tag **A** are not in lower case, so it is incorrect.

Closing the Tags

Each and every XHTML tag should have an equivalent closing tag, even empty elements should also have closing tags. Here is an example showing valid and invalid ways of using tags –

```
<!-- This is invalid in XHTML -->
<p>This paragraph is not written according to XHTML syntax.
```

```
<!-- This is also invalid in XHTML -->

```

The following syntax shows the correct way of writing above tags in XHTML. Difference is that, here we have closed both the tags properly.

```
<!-- This is valid in XHTML -->
<p>This paragraph is not written according to XHTML syntax.</p>
```

```
<!-- This is also valid now -->

```

Attribute Quotes

All the values of XHTML attributes must be quoted. Otherwise, your XHTML document is assumed as an invalid document. Here is the example showing syntax –

```
<!-- This is invalid in XHTML -->


<!-- Correct XHTML way of writing this is as follows -->

```

Attribute Minimization

XHTML does not allow attribute minimization. It means you need to explicitly state the attribute and its value. The following example shows the difference –

```
<!-- This is invalid in XHTML -->
<option selected>

<!-- Correct XHTML way of writing this is as follows -->
<option selected="selected">
```

Here is a list of the minimized attributes in HTML and the way you need to write them in XHTML –

HTML Style	XHTML Style
compact	compact="compact"
checked	checked="checked"
declare	declare="declare"
readonly	readonly="readonly"
disabled	disabled="disabled"
selected	selected="selected"
defer	defer="defer"
ismap	ismap="ismap"
noreferrer	noreferrer="noreferrer"
noshade	noshade="noshade"

nowrap	nowrap="nowrap"
multiple	multiple="multiple"
noresize	noresize="noresize"

The *id* Attribute

The id attribute replaces the name attribute. Instead of using name = "name", XHTML prefers to use id = "id". The following example shows how –

```
<!-- This is invalid in XHTML -->
```

```

```

```
<!-- Correct XHTML way of writing this is as follows -->
```

```

```

The *language* Attribute

The language attribute of the script tag is deprecated. The following example shows this difference –

```
<!-- This is invalid in XHTML -->
```

```
<script language="JavaScript" type="text/JavaScript">
    document.write("Hello XHTML!");
</script>
```

```
<!-- Correct XHTML way of writing this is as follows -->
```

```
<script type="text/JavaScript">
    document.write("Hello XHTML!");
</script>
```

Nested Tags

You must nest all the XHTML tags properly. Otherwise your document is assumed as an incorrect XHTML document. The following example shows the syntax –

```
<!-- This is invalid in XHTML -->
```

```
<b><i> This text is bold and italic</b></i>
```

<-- Correct XHTML way of writing this is as follows --!>

<i> This text is bold and italic</i>

Element Prohibitions

The following elements are not allowed to have any other element inside them. This prohibition applies to all depths of nesting. Means, it includes all the descending elements.

Element	Prohibition
<a>	Must not contain other <a> elements.
<pre>	Must not contain the , <object>, <big>, <small>, <sub>, or <sup> elements.
<button>	Must not contain the <input>, <select>, <textarea>, <label>, <button>, <form>, <fieldset>, <iframe> or <isindex> elements.
<label>	Must not contain other <label> elements.
<form>	Must not contain other <form> elements.

A Minimal XHTML Document

The following example shows you a minimum content of an XHTML 1.0 document –

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/TR/xhtml1" xml:lang="en" lang="en">
```

```
<head>
```

```
<title>Every document must have a title</title>
```

```
</head>
```

```
<body>
```

```
...your content goes here...
```

```
</body>
```

```
</html>
```

4.1.10 HTML5

Hypertext Markup Language revision 5 (HTML5) is markup language for the structure and presentation of World Wide Web contents. HTML5 supports the traditional HTML and XHTML-style syntax and other new features in its markup, New APIs, XHTML and error handling.

There are three organizations that are currently in charge of the specification of HTML5:

Web Hypertext Application Technology Working Group (WHATWG) created the HTML5 specification and is in charge of the HTML5 development that provides open collaboration of browser vendors and other involved parties.

World Wide Web Consortium (W3C) is in charge with delivering the HTML5 specification.

Internet Engineering Task Force (IETF) is in charge of the development of HTML5 WebSocket API.

HTML5 is an effort is to bring order to web development chaos by organizing common practices, embracing implementations from various browsers. It is massive, with over 100 specifications as part of the HTML5 specs. Understanding this, you can simplify by thinking of HTML5 this way. HTML5 is simply just an umbrella term for the next generation of web apps and how functionality will be expanded with better markup (HTML), better style (CSS), and better interactivity (JavaScript).

The specification of HTML5 that has been published currently is not yet final. HTML5 is expected due for Candidate Recommendation (CR) by 2012, and is expected for Proposed Recommendation (PR) by 2022. However, this does not mean that HTML5 is not ready for use. The proposed recommendation does mean however that there will be two interoperable implementations. As of 2011, browser vendors are actively adding support for new features of HTML5.

New features of HTML5 include:

- New parsing rules that are not based on SGML but are oriented towards flexible parsing and compatibility.
- Support of use of inline Scalar Vector Graphics (SVG) and Mathematical Markup Language (MathML) in text/html.

Did You Know?

In 1980, physicist Tim Berners-Lee, a contractor at CERN, proposed and prototyped ENQUIRE, a system for CERN researchers to use and share documents. In 1989, Berners-Lee wrote a memo proposing an Internet-based hypertext system. Berners-Lee specified HTML and wrote the browser and server software in late 1990.

- New available elements include article, aside, audio, bdi, canvas, command, datalist, details, embed, figcaption, figure, footer, header, hgroup, keygen, mark, meter, nav, output, progress, rp, rt, ruby, section, source, summary, time, video and wbr.
- New available types of form controls include dates and times, email, url, search, number, range, tel and color.
- New available attributes of charset on meta and async on script.

Global attributes that can be applied for every element that include id, tabindex, hidden, data-* or customer data attributes.

4.2 CASCADING STYLE SHEETS (CSS)

Cascading Style Sheets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable. CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects. CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.

4.2.1 Meaning of CSS

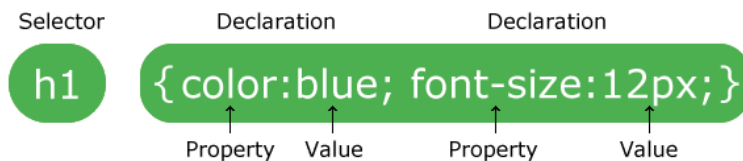
Cascading Style Sheet (CSS) is a way to design a website, or a group of websites, so that they have a consistent look and feel, and so that their look and feel is easy to change. By using CSS to design a website, the web developer gains a greater degree of control over how the site appears.

A web developer can use a CSS file to control the look of a website in three main ways. The first way is called inline, referring to the fact that the code is placed right into the line of the website code. For example, a web developer might want to make a particular sentence appear in bold, red type so that it stands out. She could use CSS to set the style of that sentence to bold and red using inline code. The benefit of this method is that it allows a quick and easy change to a particular part of a web page. Another way that a web developer can use CSS is to make rules for a single web page. In this case, the developer would use what is called embedded CSS. The developer can, for example, make each new paragraph indent and each header in bold. The embedded instructions are usually placed at the top of the web page's code. This allows the developer to change the embedded code once and have the effects take place throughout the entire page. If he decided to put all headers in italics rather than bold text, he could simply change the style coding, and all the headers on that page would

change. This has an advantage over the inline method in that it covers the entire web page, and changes can be made to the entire page at once. The final common type of CSS is what is known as an external CSS. A web developer will write the code to apply to an entire group of web pages, a whole website, or even multiple web sites. These rules can include things like background color, text color, word spacing, and other elements of page layout, just like the previous two examples of CSS. The difference is that these instructions are not for a single section of the page, or just one web page, but for an entire website. The advantage is that the look and feel of an entire website can be changed at one time by making changes to the external style sheet. If the designer wants to try a new background color or a new font for the entire website, she can do so with the change of a few lines in the external code, rather than going to each page individually and making changes there.

CSS Syntax

A CSS rule-set consists of a selector and a declaration block:



The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.

In the following example all `<p>` elements will be center-aligned, with a red text color:

Example

```
p {  
    color: red;  
    text-align: center;  
}
```

4.2.2 History of CSS

Before Cascading Style Sheets (CSS) there was very little that could be done to change the design of a web page. While Hyper Text Markup Language (HTML) creates

documents for the World Wide Web, it was specifically designed to hold the content of a web page. Housed in a separate file, CSS adds the style and design to a web page. The term cascading comes from the ability to combine multiple CSS files to determine the style for one page.

As more people started using HTML, the demand grew for more design capabilities, which would allow developers to control how web documents looked. But browsers offered limited capabilities for styling. In 1993 NCSA Mosaic was released, making the web more popular than ever, but it only offered limited capability to change fonts and colors.

In October 1994, Tim Berners-Lee formed the World Wide Web Consortium (W3C) at the Massachusetts Institute of Technology Laboratory for Computer Science. The W3C has members that are government entities, businesses, educational institutions and individuals. The W3C creates recommendations that are used to keep the web experience consistent among different browsers.

Hakon Wium Lie released the first draft of “Cascading HTML Style Sheets” in October 1994. Brent Bos was also building Argo, a browser that had style sheets which could be used by other markup languages. In all, nine different style sheet proposals were submitted. The discussions on the www-style mailing list, which was created in May 1995, influenced the development of CSS. Because it was important to be able to use the style sheets for other markup languages, HTML was removed from the name. The W3C set up the HTML Editorial Review Board (ERB) in late 1995 to make HTML specifications.

Although it took 3 years for any browser to come close to fully implementing CSS, August 1996 Microsoft Internet Explorer became the first browser to support CSS. Netscape followed suit in supporting CSS, but also implemented an alternative Javascript Style Sheets, which were never fully completed, and are now deprecated. To this day, there are differences in the way CSS is implemented in different browsers, leading developers to use hacks to make web pages look consistent in different browsers.

The CSS Level 1 W3C Recommendation was made in December 1996. The release of CSS 1 supported: font properties, text attributes, alignment of text, tables, images and more, colors of text and backgrounds, spacing of words, letters and lines, margins, borders, padding and positioning, and unique identification and classification of groups of attributes.

It was after this release that the Cascading Style Sheets and Formatting Properties Working Group was formed by the W3C to focus solely on CSS.

In May of 1998, the W3C released CSS 2 which added new capabilities including z-index, media types, bidirectional text, absolute, relative and fixed positioning, and support for aural style sheets. Not long after the release of CSS 2, a new browser, Opera was released which also supported CSS. It was not until June 2011 that W3C released CSS 2.1, which fixed errors and aligned the capabilities better with browser functions.

With the release of CSS 3, capabilities were broken into modules. Between June 2011 and June 2012, the following four modules were released as formal recommendations: color, selectors level 3, namespaces and media queries. More modules have reached the working draft or **candidate recommendation** phase of release and are considered stable.

Because level 3 was separated into modules, there will be no single CSS 4 release. Some level 4 modules which build on level 3 functions have been released. New functionality has been added in other modules. Modules will continue to be released to continue to add and update features.

Although there are some limitations, CSS has added functionality and design to web pages which were once simple documents. The advantages of CSS include content and presentation being in separate files, consistency of format throughout a site, reduction of bandwidth, accessibility, the ability to easily change the style of a site, and the ability to change the design of a web page for different devices. If you take a moment to imagine a website where the only design features are font and color changes, you will recognize that CSS has contributed greatly to the popularity, ease of use and accessibility of the internet.

Keyword

A **candidate recommendation** is a version of a standard that is more mature than the WD. At this point, the group responsible for the standard is satisfied that the standard meets its goal.

4.2.3 Three Ways to Insert CSS

There are three ways of inserting a style sheet:

- External style sheet
- Internal style sheet
- Inline style

External Style Sheet

With an external style sheet, you can change the look of an entire website by changing just one file!

Each page must include a reference to the external style sheet file inside the <link> element. The <link> element goes inside the <head> section:

Example

```
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
```

An external style sheet can be written in any text editor. The file should not contain any html tags. The style sheet file must be saved with a .css extension.

Here is how the “mystyle.css” looks:

```
body {
    background-color: lightblue;
}

h1 {
    color: navy;
    margin-left: 20px;
}
```

Internal Style Sheet

An internal style sheet may be used if one single page has a unique style.

Internal styles are defined within the <style> element, inside the <head> section of an HTML page:

Example

```
<head>
<style>
body {
    background-color: linen;
}

h1 {
    color: maroon;
    margin-left: 40px;
}
```



```
</style>  
</head>
```

Inline Styles

An inline style may be used to apply a unique style for a single element.

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

The example below shows how to change the color and the left margin of a <h1> element:

Example

```
<h1 style="color:blue;margin-left:30px;">This is a heading</h1>
```

Advantages of CSS

- *CSS saves time* – You can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
- *Pages load faster* – If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
- *Easy maintenance* – To make a global change, simply change the style, and all elements in all the **web pages** will be updated automatically.
- *Superior styles to HTML* – CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- *Multiple Device Compatibility* – Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- *Global web standards* – Now HTML attributes are being deprecated and it is being recommended to use CSS. So its a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.

Disadvantages of CSS

- *Come in different levels* – There's CSS, CSS 1 up to CSS3, which has resulted in confusion among developers and web browsers. One type of CSS should be enough. It would be preferable than having to choose which CSS level to use.
- *Fragmentation* – With CSS, what works with one browser may not always work with another. This is why web developers have to test for compatibility, running the program across multiple browsers before a website is set live. If only people use Mozilla or Chrome, but they do not.
- *Lack of security*– Because it is an open text-based system, CSS does not have the built-in security that will protect it from being overridden. Anyone who has a read/write access to a website can change the CSS file, alter the links or disrupt the formatting, whether by accident or design.

4.2.4 CSS Selectors

A CSS selector is the part of a CSS rule set that actually selects the content you want to style. Let's look at all the different kinds of selectors available, with a brief description of each.

Universal Selector

The *universal selector* works like a wild card character, selecting all elements on a page. Every HTML page is built on content placed within HTML tags. Each set of tags represents an element on the page. Look at the following CSS example, which uses the universal selector:

```
* {  
    color: green;  
    font-size: 20px;  
    line-height: 25px;  
}
```

The three lines of code inside the curly braces (color, font-size, and line-height) will apply to all elements on the HTML page. As seen here, the universal selector is declared using an asterisk. You can also use the universal selector in combination with other selectors.

Element Type Selector

Also referred to simply as a "type selector," this selector must match one or more HTML elements of the same name. Thus, a selector of `nav` would match all HTML

nav elements, and a selector of `` would match all HTML unordered lists, or `` elements.

The following example uses an element type selector to match all `` elements:

```
ul {  
    list-style: none;  
    border: solid 1px #ccc;  
}
```

To put this in some context, here's a section of HTML to which we will apply the above CSS:

```
<ul>  
  <li>Fish</li>  
  <li>Apples</li>  
  <li>Cheese</li>  
</ul>  
  
<div class="example">  
  <p>Example paragraph text.</p>  
</div>  
  
<ul>  
  <li>Water</li>  
  <li>Juice</li>  
  <li>Maple Syrup</li>  
</ul>
```

There are three main elements making up this part of the page: Two `` elements and a `<div>`. The CSS will apply only to the two `` elements, and not to the `<div>`. Were we to change the element type selector to use `<div>` instead of ``, then the styles would apply to the `<div>` and not to the two `` elements.

Also note that the styles will not apply to the elements inside the `` or `<div>` elements. That being said, some of the styles may be inherited by those inner elements.

ID Selector

An ID selector is declared using a hash, or pound symbol (`#`) preceding a string of characters. The string of characters is defined by the developer. This selector matches

any HTML element that has an ID attribute with the same value as that of the selector, but minus the hash symbol.

Here's an example:

```
#container {  
    width: 960px;  
    margin: 0 auto;  
}
```

This CSS uses an ID selector to match an HTML element such as:

```
<div id="container"></div>
```

The fact that this is a `<div>` element does not matter—it could be any kind of HTML element. As long as it has an ID attribute with a value of `container`, the styles will apply.

An ID element on a web page should be unique. That is, there should only be a single element on any given page with an ID of `container`. This makes the ID selector quite inflexible, because the styles used in the ID selector rule set can be used only once per page.

If there happens to be more than one element on the page with the same ID, the styles will still apply, but the HTML on such a page would be invalid from a technical standpoint, so you will want to avoid doing this.

In addition to the problems of inflexibility, ID selectors also have the problem of very high specificity.

Class Selector

The class selector is the most useful of all CSS selectors. It's declared with a dot preceding a string of one or more characters. Just as is the case with an ID selector, this string of characters is defined by the developer. The class selector also matches all elements on the page that have their class attribute set to the same value as the class, minus the dot.

Take the following rule set:

```
.box {  
    padding: 20px;  
    margin: 10px;  
    width: 240px;  
}
```

These styles will apply to the following HTML element:

```
<div class="box"></div>
```

The same styles will also apply to any other HTML elements that have a class attribute with a value of box. Having multiple elements on a single page with the same class attribute is beneficial, because it allows you to reuse styles, and avoid needless repetition. In addition to this, class selectors have very low specificity—again, more on this later.

Another reason the class selector is a valuable ally is that HTML allows multiple classes to be added to a single element. This is done by separating the classes in the HTML class attribute using spaces. Here's an example:

```
<div class="box box-more box-extended"></div>
```

4.2.5 Descendant Combinator

The descendant selector or, more accurately, the descendant combinator lets you combine two or more selectors so you can be more specific in your selection method. For example:

```
#container .box {  
    float: left;  
    padding-bottom: 15px;  
}
```

This declaration block will apply to all elements that have a class of box that are inside an element with an ID of container. It's worth noting that the .box element does not have to be an immediate child: there could be another element wrapping .box, and the styles would still apply.

Look at the following HTML:

```
<div id="container">  
    <div class="box"></div>  
  
    <div class="box-2"></div>  
</div>  
  
<div class="box"></div>
```

If we apply the CSS in the previous example to this section of HTML, the only element that will be affected by those styles is the first <div> element that has a class of box. The <div> element that has a class of box-2 would not be affected by the styles. Similarly, the second <div> element with a class of box would not be affected because it's not inside an element with an ID of container.

You should be careful when using the descendant combinator in your CSS. This kind of selector, while making your CSS a little easier to read, can unnecessarily restrict your styles to a specific context—in this case, the styles are restricted to boxes inside of `#container`—which can make your code inflexible.

Child Combinator

A selector that uses the child combinator is similar to a selector that uses a descendant combinator, except it only targets immediate child elements:

```
#container > .box {  
  float: left;  
  padding-bottom: 15px;  
}
```

This is the same code from the descendant combinator example, but instead of a space character, we are using the greater-than symbol (or right angle bracket.)

In this example, the selector will match all elements that have a class of `box` and that are immediate children of the `#container` element. That means, unlike the descendant combinator, there cannot be another element wrapping `.box`—it has to be a direct child element.

Here's an HTML example:

```
<div id="container">  
  <div class="box"></div>  
  
  <div>  
    <div class="box"></div>  
  </div>  
</div>
```

In this example, the CSS from the previous code example will apply only to the first `<div>` element that has a class of `box`. As you can see, the second `<div>` element with a class of `box` is inside another `<div>` element. As a result, the styles will not apply to that element, even though it too has a class of `box`.

Again, selectors that use this combinator can be somewhat restricting, but they can come in handy—for example, when styling nested lists.

General Sibling Combinator

A selector that uses a general sibling combinator matches elements based on sibling relationships. That is to say, the selected elements are beside each other in the HTML.

```
h2 ~ p {  
    margin-bottom: 20px;  
}
```

This type of selector is declared using the tilde character (~). In this example, all paragraph elements (<p>) will be styled with the specified rules, but only if they are siblings of <h2> elements. There could be other elements in between the <h2> and <p>, and the styles would still apply.

Let's apply the CSS from above to the following HTML:

```
<h2>Title</h2>  
<p>Paragraph example.</p>  
<p>Paragraph example.</p>  
<p>Paragraph example.</p>  
<div class="box">  
    <p>Paragraph example.</p>  
</div>
```

In this example, the styles will apply only to the first three paragraph elements. The last paragraph element is not a sibling of the <h2> element because it sits inside the <div> element.

Adjacent Sibling Combinator

A selector that uses the adjacent sibling combinator uses the plus symbol (+), and is almost the same as the general sibling selector. The difference is that the targeted element must be an immediate sibling, not just a general sibling. Let's see what the CSS code for this looks like:

```
p + p {  
    text-indent: 1.5em;  
    margin-bottom: 0;  
}
```

This example will apply the specified styles only to paragraph elements that immediately follow other paragraph elements. This means the first paragraph element on a page would not receive these styles. Also, if another element appeared between two paragraphs, the second paragraph of the two would not have the styles applied.

So, if we apply this selector to the following HTML:

```
<h2>Title</h2>  
<p>Paragraph example.</p>
```

```
<p>Paragraph example.</p>
<p>Paragraph example.</p>

<div class="box">
  <p>Paragraph example.</p>
  <p>Paragraph example.</p>
</div>
```

4.2.6 Attribute Selector

The attribute selector targets elements based on the presence and/or value of HTML attributes, and is declared using square brackets:

```
input[type="text"] {
  background-color: #444;
  width: 200px;
}
```

There should not be a space before the opening square bracket unless you intend to use it along with a descendant combinator. The above CSS would match the following element:

```
<input type="text">
```

But it wouldn't match this one:

```
<input type="submit">
```

The attribute selector can also be declared using just the attribute itself, with no value, like this:

```
input[type] {
  background-color: #444;
  width: 200px;
}
```

This will match all input elements with an attribute of type, regardless of the value.

You can also use attribute selectors without specifying anything outside the square brackets (thus targeting based on the attribute alone, irrespective of the element). It's also worth noting that, when using values, you have the option to include quotes (single or double,) or not.

4.2.7 Pseudo-class

A pseudo-class uses a colon character to identify a pseudo-state that an element might be in—for example, the state of being hovered, or the state of being activated. Let's look at a common example:

```
a:hover {  
    color: red;  
}
```

The pseudo-class portion of the selector is the: hover part. Here we have attached this pseudo-class to all anchor elements (elements). This means that when the user hovers their mouse over an element, the color property for that element will change to red. This type of pseudo-class is a dynamic pseudo-class, because it occurs only in response to user interaction—in this case, the mouse moving over the targeted element.

Pseudo-element

Finally, CSS has a selector referred to as a pseudo-element and, used appropriately, it can be very useful. The only caveat is that this selector is quite different from the other examples we have considered. Let's see a pseudo-element in context:

```
.container:before {  
    content: "";  
    display: block;  
    width: 50px;  
    height: 50px;  
    background-color: #141414;  
}
```

This example uses one kind of pseudo-element, the :before pseudo-element. As the name suggests, this selector inserts an imaginary element into the page, inside the targeted element, before its contents.

Remember

It's important to recognize that these types of selectors do not just select elements; they select elements that are in a particular state.

4.3 INHERITANCE

Inheritance is the mechanism by which certain properties are passed on from a parent element down to its children, in the

same fashion as genetics: if parents have blue eyes, their children will probably also have blue eyes. Not all CSS properties are inherited, because it does not make sense for some of them to be. For instance, margins and width are not inherited, since it is unlikely that a child element requires the same margins as its parent. Imagine if you set the content block of a site to be 70% of the browser window width, and then all of its children adopted a width of 70% of their parent elements? Designing page layouts with CSS would be a nightmare.

Why Inheritance is Useful

CSS has an inheritance mechanism because otherwise CSS rules would be redundant. Without inheritance, it would be necessary to specify styles like font family, font size, and text color individually for every single element type. The code would become bloated and repetitive. Using inheritance, you can specify the font properties for the html or body elements and the styles will be inherited by all other elements. You can specify background and text colors for a specific container element and the text color will automatically be inherited by any child elements in that container. The background color is not inherited, but the initial value for background-color is transparent, which means a parent's background will shine through. The effect is similar to the page's appearance if background colors were inherited.

Note: Consider what would happen if background *images* were inherited. Every child would display the same background image as its parent. The result would look like a jigsaw puzzle put together by someone with a serious drug problem, since the background would be redrawn inside each subsequent child element.

4.3.1 How Inheritance Works

Every element in an HTML document inherits all inheritable properties from its parent *except* the root element (<html>), which does not have a parent.

Whether or not the inherited properties will have any effect depends on other things, as described later in the section about the cascade. Just as a blue-eyed mother can have a brown-eyed child if the father has brown eyes, inherited properties in CSS can be overridden.

An example of inheritance

Copy the following HTML document into a new file in your favorite text editor and save it as inherit.html.

```
<!DOCTYPE html>
<html lang="en">
  <head>
```

```
<meta charset="UTF-8">
<title>Inheritance</title>
</head>
<body>
  <h1>Heading</h1>
<p>A paragraph of text.</p>
</body>
</html>
```

If you open the document in your web browser, you will see a rather boring document displayed according to your browser's default styling.

2. Create a new empty file in your text editor, copy the following CSS rule into it, and save the file as `style.css` in the same location as the HTML file.

```
html {
  font: 75% Verdana, sans-serif;
}
```

3. Link the style sheet to your HTML document by inserting the following line before the `</head>` tag:

```
<link rel="stylesheet" type="text/css" href="style.css">
```

4. Save the modified HTML file and reload the document in your browser.

If you do not have Verdana installed on your computer, the text displays in the default sans-serif font specified in your browser settings. The text is also smaller; only three quarters of its size in the unstyled version. The CSS rule applies only to the `<html>` element. It does not specify any rules for headings or paragraphs, yet all of the text now displays in Verdana at 75% of its default size. Why? Because of inheritance. The font property is a shorthand property that sets a whole range of font-related properties.

The CSS rule only specified two properties — the font size and the font family — but that rule is equivalent to the following:

```
html {
  font-style: normal;
  font-variant: normal;
  font-weight: normal;
  font-size: 75%;
  line-height: normal;
  font-family: Verdana,sans-serif;
}
```

All of those properties are inherited, so the `<body>` element will inherit them from the `<html>` element and then pass them on to its children — the heading and the paragraph text. But notice that there is something unusual occurring with the inheritance of font size.

The `<html>` element's font size is set to 75%, but 75% of *what*? Surely the font size of the `<body>` should be 75% of its parent's font size, and the font sizes of the heading and the paragraph be 75% of that of the `<body>` element? The value inherited is not the specified value — the value typed in the style sheet — but something else called the computed value.

The computed value is, in the case of font size, an absolute value measured in pixels. For the `<html>` element, which does not have a parent element from which to inherit, a percentage value for font size relates to the default font size set in the browser. Most contemporary browsers have a default font size setting of 16px. 75% of 16 is 12, so the computed value for the font size of the `<html>` element is around 12px.

And *that* is the value that is inherited by `<body>` and passed on to the heading and the paragraph. (The font size of the heading is larger, because the browser applies some built-in styling of its own.

Consider this example:

1. Add two more declarations to the rule in your CSS style sheet:

```
html {  
    font: 75% Verdana,sans-serif;
```

```
    /* Add these */
```

```
    background-color: blue;
```

```
    color: white
```

```
}
```

2. Save the CSS file and reload the document in your browser. Now the background of the whole document is bright blue, and all of the text is white. The white text color is inherited by the `<body>` element and passed on to all children of body — the heading and the paragraph. The heading and paragraph do not, however, inherit the background but instead default to transparent, so the net visual result is white text displayed on a blue background.

3. Add another new rule to the style sheet:

```
h1 {  
    font-size: 300%;  
}
```

4. Save and reload the document: this rule sets the font size of the heading. The percentage is applied to the inherited font size — 12px, as we discussed above — so the heading size will be 300% of 12px, or 36px.

4.3.2 Forcing Inheritance

You can force inheritance — even for properties that are not inherited by default — by using the `inherit` keyword. Use this strategy with care, however, since Microsoft Internet Explorer versions up to and including version 7 do not support this keyword.

The following rule will make all paragraphs inherit all background properties from their parents:

```
p {  
    background: inherit;  
}
```

Forcing inheritance is not a common practice. It can be useful for “undoing” a declaration in a conflicting rule, or to avoid hard coding certain values. As an example, consider a typical navigation menu:

```
<ul id="nav">  
    <li><a href="/">Home</a></li>  
    <li><a href="/news/">News</a></li>  
    <li><a href="/products/">Products</a></li>  
    <li><a href="/services/">Services</a></li>  
    <li><a href="/about/">About Us</a></li>  
</ul>
```

To display this list of links as a horizontal menu, you could use the following CSS rules:

```
#nav {  
    background: blue;  
    color: white;  
    margin: 0;  
    padding: 0;  
}
```

```
#nav li {  
    display: inline;  
    margin: 0;
```

```
padding: 0 0.5em;
border-right: 1px solid;
}

#nav li a {
  color: inherit;
  text-decoration: none;
}
```

Here the background color of the whole list is set to blue in the rule for #nav. This also sets the foreground color to white, and this property is inherited by each list item and each link. The rule for the list items sets a right border, but does not specify the border color, which means it will use the inherited foreground color, white. For the links, the `color: inherit;` forces inheritance and overrides the browser's default link color.

Of course, you can specify the border color as white and the link text color as white, but the beauty of letting inheritance do the job is that it is only necessary to update one place to change the colors if you decide to update the color scheme later.

4.3.3 The Cascade

CSS an acronym of Cascading Style Sheets, so it is not a surprise that the cascade is an important concept. It is the mechanism that controls the end result when multiple, conflicting CSS declarations apply to the same element. There are three main concepts that control the order in which CSS declarations are applied:

- Importance
- Specificity
- Source order

Importance is the most ... er ... important. If two declarations have the same importance, the specificity of the rules decide which one will apply. If the rules have the same specificity, then source order controls the outcome.

Importance

The importance of a CSS declaration depends on where it is specified. The conflicting declarations will be applied in the following order, with later declarations overriding earlier ones:

- User agent style sheets
- Normal declarations in user style sheets
- Normal declarations in author style sheets

- Important declarations in author style sheets
- Important declarations in user style sheets

User Agent Style Sheets

A user agent style sheet is the built-in style sheet of the browser. Every browser has its default rules for how to display various HTML elements if no style is specified by the user or designer of the page. For instance, unvisited links are usually blue and underlined. A user style sheet is a style sheet that the *user* has specified. Not all browsers support user style sheets, but they can be very useful, especially for users with certain types of disabilities. For instance, a dyslexic person can have a user style sheet that specifies certain fonts and colors to help them read more easily.

An author style sheet is what we normally refer to when we say “style sheet”. It is the style sheet that the author of the document (or, more likely, the website’s designer) has written and linked (or included).

Normal and Important Declarations

Normal declarations are just that: normal declarations.

The opposite is important declarations, which are declarations followed by an `!important` directive. A dyslexic user might, for instance, want all text to be displayed in Comic Sans MS if he finds that font easier to read. He could then have a user style sheet containing the following rule:

```
* {  
    font-family: “Comic Sans MS” !important;  
}
```

Important declarations in a user style sheet will trump everything else, which is logical. No matter what the designer specified, and no matter what the browser’s default font family is set to, everything will be displayed in Comic Sans MS. The default browser rendering will only apply if those declarations are not overridden by any rules in a user style sheet or an author style sheet, since the **user agent style sheet** has the lowest precedence.

Keyword

User agent style sheets are overridden by anything that you set in your own style sheet.

To be honest, most designers do not have to think too much about importance, since there's nothing we can do about it. There is no way we could know if a user has a user style sheet defined that will override our CSS. If they do, they probably have a very good reason for doing so, anyway. Still, it's good to know what importance is and how it may affect the presentation of our documents.

4.3.4 Specificity

Specificity is something every CSS author needs to understand and to think about. It can be thought of as a measure of how specific a rule's selector is. A selector with low specificity may match many elements (like `*`, which matches every element in the document), while a selector with high specificity might only match a single element on a page (like `#nav`, which only matches the element with an id of `nav`).

The specificity of a selector can easily be calculated, as we shall see below. If two or more declarations conflict for a given element, and all the declarations have the same importance, then the one in the rule with the most specific selector will “win”.

Specificity has four components; let's call them *a*, *b*, *c* and *d*. Component “*a*” is the most distinguishing, “*d*” the least.

- Component “*a*” is quite simple: it's 1 for a declaration in a style attribute (also known as inline styling), otherwise it's 0.
- Component “*b*” is the number of id selectors in the selector (those that begin with a `#`).
- Component “*c*” is the number of attribute selectors—including class selectors — and pseudo-classes.
- Component “*d*” is the number of element types and pseudo-elements in the selector.

After a bit of counting, we can thus string those four components together to get the specificity for any rule. CSS declarations in a style attribute do not have a selector, so their specificity is always 1,0,0,0.

Let's look at a few examples — after this it should be quite clear how this works.

Selector	a	b	c	d	Specificity
h1				1	0,0,0,1
.foo			1		0,0,1,0
#bar		1			0,1,0,0
html>head+body ul#nav *.home a:link		1	2	5	0,1,2,5

Let's look at the last example in some more detail. *a* = 0 since it's a selector, not a declaration in a style attribute. There is one ID selector (`#nav`), so *b* = 1. There is one

attribute selector (.home) and one pseudo-class (:link), so $c = 2$. There are five element types (<html>, <head>, <body>, and <a>), so $d = 5$.

The final specificity is thus 0,1,2,5.

Note: Combinators (like >, + and white space) do not affect a selector's specificity. The universal selector (*) has no input on specificity, either.

Note #2: There is a huge difference in specificity between an id selector and an attribute selector that happens to refer to an id attribute. Although they match the same element, they have very different specificities. The specificity of #nav is 0,1,0,0 while the specificity of [id="nav"] is only 0,0,1,0.

Let's look at how this works in practice.

1. First of all, start by adding another paragraph to your HTML document.

```
<body>
  <h1>Heading</h1>
  <p>A paragraph of text.</p>

  <!-- Add this -->
  <p>A second paragraph of text.</p>
</body>
```

2. Next, add a rule to your stylesheet to make the paragraph text have a different color:

```
p {
  color: cyan;
}
```

3. Now save both files and reload the document in your browser; there should now be two paragraphs with cyan text.

4. Set an id on the first paragraph so you can target it easily with a CSS selector.

```
<body>
  <h1>Heading</h1>
  <!-- Add the id of "special" to this paragraph -->
  <p id="special">A paragraph of text.</p>
  <p>A second paragraph of text.</p>
</body>
```

5. Carry on by adding a rule for the special paragraph in your style sheet:

```
#special {
  background-color: red;
```

```
color: yellow;  
}
```

6. Finally, save both files and reload the document in your browser to see the now rather colorful result.

Let's look at the declarations that apply to the two paragraphs.

The first rule you added sets a text color of cyan for *all* paragraphs. The second rule sets a red background color and a yellow text color for the single element that has the id of special. Your first paragraph matches both of those rules; it is a paragraph and it has an id of special.

The red background is not a problem, because it's only specified for #special. Both rules contain a declaration of the color property, though, which means there is a conflict. Both rules have the same importance — they are normal declarations in the author style sheet — so you have to look at the specificity of the two selectors.

The selector of the first rule is <p>, which has a specificity of 0,0,0,1 according to the rules above since it contains a single element type. The selector of the second rule is #special, which has a specificity of 0,1,0,0 since it consists of an id selector. 0,1,0,0 is much more specific than 0,0,0,1 so the color: yellow; declaration wins and you get yellow text on a red background.

4.3.5 Source Order

If two declarations affect the same element, have the same importance and the same specificity, the final distinguishing mark is the source order. The declaration that appears later in the style sheets will “win” over those that come before it.

If you have a single external style sheet, then the declarations at the end of the file will override those that occur earlier in the file if there's a conflict. The conflicting declarations could also occur in different style sheets.

In that case, the order in which the style sheets are linked, included or imported controls what declaration will be applied, so if you have two stylesheets linked in a document <head>, the one linked to further down will override the one linked to higher up. Let's look at a practical example of how this works.

1. Add a new rule to your style sheet at the very end of the file, like so:

```
p {  
  background-color: yellow;  
  color: black;  
}
```

2. Save and reload the web page. You will now have *two* rules that match all paragraphs. They have the same importance and the same specificity (since the selector

is the same), therefore the final mechanism for choosing which one wins will be the source order. The last rule specifies color:black and that will override color:cyan from the earlier rule.

4.4 CONCEPT OF JAVASCRIPT

JavaScript is used to extend functionality in the websites, The uses range from on screen visual effects to processing & calculating the data on the web pages with ease as well as extended functionality to the websites using third party scripts among several other handy features and it is one of the most popular programming languages.

JavaScript is a dominant web development technology, it is the full-fledged programming language, it provides many unique features, it has been around for over 20 years now, it becomes a favorite for many up-and-coming Web developers in the recent years. JavaScript is the most misunderstood language in the world , That's may be because of the name , It has a prefix Java in its name , for which few people thinks it has some relation with JAVA , but actually it is not , But the most misinterpretation is done due to the second word in its name , which is Script .



JavaScript is not a script to validate forms or making animations , It's a real language , small & sophisticated , It's not a subset of JAVA , It is a completely independent , flexible , super dynamic , object oriented language .

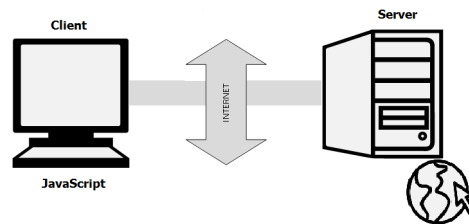
JavaScript cannot make any .exe, .dll or any abstract class files , So , it throws the errors & exceptions while executing only that particular path of the program , The frameworks such as backbone , ember , angular , jQuery and many more are developed on top of JavaScript & they are changing the old concepts of web development.

In case of languages such as JavaScript , where you do not have to use any IDE , this approach helps you in proper line by line debugging , if any part of the same program is under construction , You can run the other flows of the program .

4.4.1 Meaning of JavaScript

Javascript is a dynamic computer programming language. It is lightweight and most commonly used as a part of **web pages**, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name LiveScript. The general-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers. JavaScript is a very powerful client-side scripting language. JavaScript is used mainly for enhancing the interaction of a user with the webpage. In other words, you can make your webpage more lively and interactive, with the help of JavaScript. JavaScript is also being used widely in game development and Mobile application development.



The ECMA-262 Specification defined a standard version of the core JavaScript language.

- JavaScript is a lightweight, interpreted programming language.
- Designed for creating network-centric applications.
- Complementary to and integrated with Java.
- Complementary to and integrated with HTML.
- Open and cross-platform

4.4.2 The History of JavaScript

Beginnings at Netscape

In 1993, the National Center for Supercomputing Applications (NCSA), a unit of the University of Illinois at Urbana-Champaign, released NCSA Mosaic, the first popular graphical Web browser, which played an important part in expanding the growth of the nascent World Wide Web. In 1994, a company called Mosaic Communications was founded in Mountain View, California and employed many of the original NCSA Mosaic authors to create Mosaic Netscape. However, it intentionally shared no code with NCSA Mosaic. The internal codename for the company's browser was Mozilla, which stood for "Mosaic killer", as the company's goal was to displace NCSA Mosaic as the world's number one web browser. The first version of the Web browser, Mosaic Netscape 0.9, was released in late 1994. Within four months it had already taken three-quarters of the browser market and became the main browser for the Internet

in the 1990s. To avoid trademark ownership problems with the NCSA, the browser was subsequently renamed Netscape Navigator in the same year, and the company took the name Netscape Communications. Netscape Communications realized that the Web needed to become more dynamic. Marc Andreessen, the founder of the company believed that HTML needed a “glue language” that was easy to use by Web designers and part-time programmers to assemble components such as images and plugins, where the code could be written directly in the Web page markup.

In 1995, Netscape Communications recruited Brendan Eich with the goal of embedding the Scheme programming language into its Netscape Navigator. Before he could get started, Netscape Communications collaborated with Sun Microsystems to include in Netscape Navigator Sun’s more static programming language Java, in order to compete with Microsoft for user adoption of Web technologies and platforms. To defend the idea of JavaScript against competing proposals, the company needed a prototype. Eich wrote one in 10 days, in May 1995.

Although it was developed under the name Mocha, the language was officially called LiveScript when it first shipped in beta releases of Netscape Navigator 2.0 in September 1995, but it was renamed JavaScript when it was deployed in the Netscape Navigator 2.0 beta 3 in December. The final choice of name caused confusion, giving the impression that the language was a spin-off of the Java programming language, and the choice has been characterized as a marketing ploy by Netscape to give JavaScript the cachet of what was then the hot new Web programming language. There is a common misconception that JavaScript was influenced by an earlier Web page scripting language developed by Nombas named Cmm (not to be confused with the later C++ created in 1997). Brendan Eich, however, had never heard of Cmm before he created LiveScript. Nombas did pitch their embedded Web page scripting to Netscape, though Web page scripting was not a new concept, as shown by the ViolaWWW Web browser. Nombas later switched to offering JavaScript instead of Cmm in their ScriptEase product and was part of the TC39 group that standardized ECMAScript.

Server-side JavaScript

In December 1995, soon after releasing JavaScript for browsers, Netscape introduced an implementation of the language for server-side scripting with Netscape Enterprise Server. Since 1996, the IIS web-server has supported Microsoft’s implementation of server-side Javascript -- JScript—in ASP and .NET pages. Since the mid-2000s, additional server-side JavaScript implementations have been introduced, such as Node.js in 2009.

Adoption by Microsoft

Microsoft script technologies including VBScript and JScript were released in 1996. JScript, a reverse-engineered implementation of Netscape’s JavaScript, was part of Internet Explorer 3. JScript was also available for server-side scripting in Internet

Information Server. Internet Explorer 3 also included Microsoft's first support for CSS and various extensions to HTML, but in each case the implementation was noticeably different from that found in Netscape Navigator at the time.

These differences made it difficult for designers and programmers to make a single website work well in both browsers, leading to the use of "best viewed in Netscape" and "best viewed in Internet Explorer" logos that characterized these early years of the browser wars. JavaScript began to acquire a reputation for being one of the roadblocks to a cross-platform and standards-driven Web. Some developers took on the difficult task of trying to make their sites work in both major browsers, but many could not afford the time. With the release of Internet Explorer 4, Microsoft introduced the concept of Dynamic HTML, but the differences in language implementations and the different and proprietary Document Object Models remained and were obstacles to widespread take-up of JavaScript on the Web.

Standardization

In November 1996, Netscape submitted JavaScript to ECMA International to carve out a standard specification, which other browser vendors could then implement based on the work done at Netscape. This led to the official release of the language specification ECMAScript published in the first edition of the ECMA-262 standard in June 1997, with JavaScript being the most well-known of the implementations. ActionScript and JScript were other well-known implementations of ECMAScript.

The standards process continued in cycles, with the release of ECMAScript 2 in June 1998, which brings some modifications to conform to the ISO/IEC 16262 international standard. The release of ECMAScript 3 followed in December 1999, which is the baseline for modern day JavaScript. The original ECMAScript 4 work led by Waldemar Horwat (then at Netscape, now at Google) started in 2000 and at first, Microsoft seemed to participate and even implemented some of the proposals in their JScript .NET language.

Over time it was clear though that Microsoft had no intention of cooperating or implementing proper JavaScript in Internet Explorer, even though they had no competing proposal and they had a partial (and diverged at this point) implementation on the .NET server side. So by 2003, the original ECMAScript 4 work was mothballed.

The next major event was in 2005, with two major happenings in JavaScript's history. First, Brendan Eich and Mozilla rejoined Ecma International as a not-for-profit member and work started on ECMAScript for XML (E4X), the ECMA-357 standard, which came from ex-Microsoft employees at BEA Systems (originally acquired as Crossgain). This led to working jointly with Macromedia (later acquired by Adobe Systems), who were implementing E4X in ActionScript 3 (ActionScript 3 was a fork of original ECMAScript 4).

So, along with Macromedia, work restarted on ECMAScript 4 with the goal of standardizing what was in ActionScript 3. To this end, Adobe Systems released the

ActionScript Virtual Machine 2, code named Tamarin, as an open source project. But Tamarin and ActionScript 3 were too different from web JavaScript to converge, as was realized by the parties in 2007 and 2008.

Alas, there was still turmoil between the various players; Douglas Crockford—then at Yahoo!—joined forces with Microsoft in 2007 to oppose ECMAScript 4, which led to the ECMAScript 3.1 effort. The development of ECMAScript 4 was never completed, but that work influenced subsequent versions.

While all of this was happening, the open source and developer communities set to work to revolutionize what could be done with JavaScript. This community effort was sparked in 2005 when Jesse James Garrett released a white paper in which he coined the term Ajax, and described a set of technologies, of which JavaScript was the backbone, used to create web applications where data can be loaded in the background, avoiding the need for full page reloads and leading to more dynamic applications. This resulted in a renaissance period of JavaScript usage spearheaded by open source libraries and the communities that formed around them, with libraries such as Prototype, jQuery, Dojo Toolkit, MooTools, and others being released.

In July 2008, the disparate parties on either side came together in Oslo. This led to the eventual agreement in early 2009 to rename ECMAScript 3.1 to ECMAScript 5 and drive the language forward using an agenda that is known as Harmony. ECMAScript 5 was finally released in December 2009.

In June 2011, ECMAScript 5.1 was released to fully align with the third edition of the ISO/IEC 16262 international standard. ECMAScript 2015 was released in June 2015. ECMAScript 2016 was released in June 2016. The current version is ECMAScript 2017, released in June 2017.

Later Developments

JavaScript has become one of the most popular programming languages on the Web. Initially, however, many professional programmers denigrated the language because, among other reasons, its target audience consisted of Web authors and other such “amateurs”. The advent of Ajax returned JavaScript to the spotlight and brought more professional programming attention. The result was a proliferation of comprehensive frameworks and libraries, improved JavaScript programming practices, and increased usage of JavaScript outside Web browsers, as seen by the proliferation of Server-side JavaScript platforms.

In January 2009, the CommonJS project was founded with the goal of specifying a common standard library mainly for JavaScript development outside the browser.

With the rise of single-page applications and JavaScript-heavy sites, it is increasingly being used as a compile target for source-to-source compilers from both dynamic languages and static languages.

Did You Know?

JavaScript was designed by Brendan Eich in 1995 for Netscape to allow developers to enhance web pages with things like animated drop-down menus, and validating form entries.

4.4.3 Client-side JavaScript

Client-side JavaScript is the most common form of the language. The script should be included in or referenced by an HTML document for the code to be interpreted by the browser.

It means that a web page need not be a static HTML, but can include programs that interact with the user, control the browser, and dynamically create HTML content.

The JavaScript client-side mechanism provides many advantages over traditional CGI server-side scripts. For example, you might use JavaScript to check if the user has entered a valid e-mail address in a form field.

The JavaScript code is executed when the user submits the form, and only if all the entries are valid, they would be submitted to the Web Server. JavaScript can be used to trap user-initiated events such as button clicks, link navigation, and other actions that the user initiates explicitly or implicitly.

Limitations of JavaScript

- For security reason, JavaScript does not allow the reading or writing of files.
- This does not have any multiprocessor/multi-threading capabilities.
- As there is no support available, this cannot be used for networking applications.
- Cannot access web pages hosted on a different domain.
- Cannot access databases.
- Depends a lot on the browser.
- Inability to use local devices.
- JavaScript can be disabled.
- Not Search Engine Friendly.
- JavaScript cannot protect your page source or images.

Though the HTML and JavaScript may seem very old, there is nothing inherently problematic about making a complex application with them. The larger problems of web applications must deal with have to do with the nature of the world wide web(WWW) which is inconsistent of network communication and the statelessness of HTTP.

Advantages of CSS

The biggest advantages to a JavaScript having an ability to produce the same result on all modern browsers.

- *Speed.* Client-side JavaScript is very fast because it can be run immediately within the client-side browser. Unless outside resources are required, JavaScript is unhindered by network calls to a backend server. It also has no need to be compiled on the client side which gives it certain speed advantages (granted, adding some risk dependent on that quality of the code developed).
- *Simplicity.* JavaScript is relatively simple to learn and implement.
- *Popularity.* JavaScript is used everywhere in the web. The resources to learn JavaScript are numerous. StackOverflow and GitHub have many projects that are using Javascript and the language as a whole has gained a lot of traction in the industry in recent years especially.
- *Interoperability.* JavaScript plays nicely with other languages and can be used in a huge variety of applications. Unlike PHP or SSI scripts, JavaScript can be inserted into any web page regardless of the file extension. JavaScript can also be used inside scripts written in other languages such as Perl and PHP.
- *Server Load.* Being client-side reduces the demand on the website server.
- *Extended Functionality.* Third party add-ons like Greasemonkey enable JavaScript developers to write snippets of JavaScript which can execute on desired web pages to extend its functionality.
- *Versatility.* Nowadays, there are many ways to use JavaScript through Node.js servers. If you were to bootstrap node.js with Express, use a document database like mongodb, and use JavaScript on the front-end for clients, it is possible to develop an entire JavaScript app from front to back using only JavaScript.
- *Updates.* Since the advent of EcmaScript 5 (the scripting specification that Javascript relies on), Ecma International has dedicated to updating JavaScript annually. So far, we have received browser support for ES6 in 2017 and look forward to ES7 being supported in future months.

JavaScript Disadvantages

Biggest disadvantages to a JavaScript, code visible to everyone.

- **Code Always Visible:** The biggest disadvantages is code always visible to everyone anyone can view JavaScript code.
- **Bit of Slow execute:** No matter how much fast JavaScript interpret, JavaScript DOM (Document Object Model) is slow and will be a never fast rendering with HTML.
- **Stop Render:** JavaScript single error can stop to render with entire site. However browsers are extremely tolerant of JavaScript errors.

4.4.4 Hello World: Writing Your First JavaScript Program

“Hello World” is a staple of programming courses. The objective of this program is simple: output the text “Hello World” on a computer screen. Because of the simplicity of the message and syntax, it is usually the first program taught to beginners. Writing a “Hello World” program in JavaScript, as we will learn, is exceptionally easy and requires not more than 3 lines of code.

What You will Need

Since JavaScript is interpreted by the browser itself, we do not need any fancy compilers or additional software to write JS programs. All you need is:

- *A text editor.* Your humble Notepad will do just fine, but we highly recommend Notepad++ (free).
- *A web browser.* You can use anything you want – Google Chrome, Firefox, Internet Explorer or Safari.

Getting Started: Creating the HTML Framework

JavaScript programs are usually embedded within the web page itself. This means they are written along with the HTML, though you can include them externally as well.

To get started, we will first create a simple HTML file where we can include our JavaScript.

Open your text editor and type the following code into it:

```
<!DOCTYPE HTML>
<html>
<head>
<title>JavaScript Hello World</title>
</head>
<body>
<h1>JavaScript Hello World Example</h1>
</body>
</html>
```

Save this file as test.html (make sure to save as ‘All Files’ if using Notepad).

This is a standard HTML template, nothing special about it. It should be pretty clear to anyone with even a basic grasp of HTML.

Adding the JavaScript Code

We can now go ahead and write the JavaScript program.

Add the following code after the <h1> tag and save the file:

```
<script>
<alert("Hello World!")
</script>
```

That's it! You have now successfully created a JavaScript program.

Now use your web browser to open test.html. This is what you should see:

Easy, right?

All JavaScript code is written between <script></script> tags. We use 'alert' to create a function. The text to be displayed is written between quotes in brackets.

But what if we wanted to create a separate "Hello World!" function we can call anytime?

We can do that as well using just a few lines of code.

4.4.5 Creating a "Hello World" JavaScript Function

A function is any block of code that can be 'called' any number of times within a program. Functions are extremely useful in programming since you can create them once, use them n number of times.

We created a "Hello World!" alert box in the above example. Now we will create a function that will create the same alert box whenever we want.

Type in the following code into your text file:

```
<!DOCTYPE HTML>
<html>
<head>
<script>
function myFunction()
{
alert("Hello World!")
}
</script>
</head>
<body>
</body>
```

```
</html>
```

Save this as test2.html.

Instead of adding the script in the `<body>`, we added the script to the `<head>` and created a function called 'helloWorld'. You can turn any piece of code into a function by wrapping it in `{ }` brackets and adding "function functionName()" before it.

Now that we have created the function, we can call on it any number of times.

Add the following code anywhere between the `<body></body>` tags:

```
<p><button onclick="myFunction()">Create a Dialog Box!</button></p>
```

```
<p><button onclick="myFunction()">Create Another Dialog Box!</button></p>
```

Altogether, your code should look like this:

Now open the test2.html file in your web browser. This is what you should see:

Click on either of the two buttons and you will see the "Hello World!" dialog box pop up.

Congratulations! You have now successfully created a function in JavaScript. This is just the beginning, however. There is still a lot more to learn in this wonderful programming language. This course on JavaScript for beginners should help you get started.

ROLE MODEL

TIM BERNERS-LEE: INVENTOR OF THE WORLD WIDE WEB

Born 8 June 1955, also known as TimBL, is an English engineer and computer scientist, best known as the inventor of the World Wide Web. He is currently a professor of computer science at the University of Oxford and the Massachusetts Institute of Technology (MIT). He made a proposal for an information management system in March 1989, and he implemented the first successful communication between a Hypertext Transfer Protocol (HTTP) client and server via the internet in mid-November the same year.

Sir Tim Berners-Lee is a British computer scientist who invented what is undoubtedly one of the most revolutionary inventions of the 20th century—the World Wide Web (WWW). A qualified software engineer who was working at CERN when he came up with the idea of a global network system, Sir Tim is also credited for creating the world's first web browser and editor. He founded the World Wide Web Foundation and directs the World Wide Web Consortium (W3C). Both of his parents worked on the Ferranti Mark I, the first commercial computer, and thus it is not surprising that he too chose the field of computers. But what is surprising is the phenomenal impact his idea of a global network has had on the world of information and technology. An alumnus of the University of Oxford, he realized the need for a global communication network while working at CERN as the researchers from all over the world needed to share their data with each other. By the late 1980s he had drawn up a proposal for creating a global hypertext document system using the internet. A few more years of pioneering work in the field led to the birth of the World Wide Web making Berners-Lee one of the most significant inventors of the modern era.



Childhood and Early Life

He was born on June 8, 1955, as Timothy Berners-Lee to Mary Lee Woods and Conway Berners-Lee. He has three siblings. Both his parents worked on the first commercially-

built computer, the Ferranti Mark I and thus Tim was fascinated by computers from a young age.

He received his primary education from Sheen Mount Primary School before moving on to London's independent Emanuel School where he studied from 1969 to 1973.

He enrolled at The Queen's College of the University of Oxford in 1973 and graduated in 1976 with a first-class degree in physics.

Career

- He was appointed as an engineer at the telecommunications company, Plessey in Poole after completing his studies. He remained there for two years, working on distributed transaction systems, message relays, and bar code technology.
- He left Plessey in 1978 and joined D. G. Nash Ltd. In this job he wrote typesetting software for intelligent printers and a multitasking operating system.
- In the late 1970s he began working as an independent consultant and worked for many companies, including CERN where he worked from June to December 1980 as a consultant software engineer.
- While at CERN he wrote a program called "Enquire" for his own personal use. It was a simple hypertext program which laid the conceptual foundation for the development of the World Wide Web in future.
- He started working at John Poole's Image Computer Systems, Ltd. in 1981. For the next three years he worked on the company's technical side which enabled him to gain experience in computer networking. His work included real time control firmware, graphics and communications software, and a generic macro language.
- He returned to CERN in 1984 after receiving a fellowship there. During the 1980s thousands of people were working at CERN and they needed to share information and data with each other. Much of the work was done by email and the scientists had to keep track of different things simultaneously. Tim realized that a simpler and more efficient method of data sharing had to be devised.
- In 1989, he wrote a proposal for a more effective communication system within the organization which eventually led to the conceptualization of the World Wide Web—an information sharing system that could be implemented throughout the world.
- The world's first ever website, Info.cern.ch, was built at CERN and put online on 6th August 1991, ushering in a new era in the field of communication and technology. The site provided information of what the World Wide Web was and how it could be used for information sharing.



- He established the World Wide Web Consortium (W3C) at the Massachusetts Institute of Technology's Laboratory for Computer Science in 1994. The W3C decided that its technologies should be royalty-free so that anyone could adopt them.
- He became a professor in the Computer Science Department at the University of Southampton, UK, in December 2004. There he worked on the Semantic Web.
- In 2006, he became the Co-Director of the Web Science Trust which was launched to analyze the World Wide Web and devise solutions to optimize its usage and design. He also serves as the Director of the World Wide Web Foundation, started in 2009.
- Along with Professor Nigel Shadbolt, he is one of the key figures behind data.gov.uk, a UK Government project to make non-personal UK government data more accessible to the public.

Major Works

- His invention, the World Wide Web, is counted among the most significant inventions of the 20th century. The web revolutionized the world of information and technology and has opened up several new avenues.

Awards and Achievements

- He was presented with The Software System Award from the Association for Computing Machinery (ACM) in 1995.
- He was named as one of the 100 Most Important People of the 20th century by the Time Magazine in 1999.
- He was made the Commander of the Order of the British Empire (KBE) in the New Year Honours "for services to the global development of the Internet" in 2004.
- In 2013, he became one of five Internet and Web pioneers awarded the inaugural Queen Elizabeth Prize for Engineering.

Personal Life and Legacy

- He met Jane while studying physics at Oxford and married her soon after graduation in 1976. This marriage, however, ended in a divorce.
- While working for CERN he became acquainted with Nancy, an American software engineer. They so fell in love and tied the knot in 1990. This marriage too ended after some years.
- Currently he is married to Rosemary Leith who he wed in June 2014.

SUMMARY

- HTML is at the core of every web page, regardless the complexity of a site or number of technologies involved. It's an essential skill for any web professional.
- CSS stands for Cascading Style Sheets. This programming language dictates how the HTML elements of a website should actually appear on the frontend of the page.
- JavaScript is a logic-based programming language that can be used to modify website content and make it behave in different ways in response to a user's actions.
- HTML was developed with the intent of defining the structure of documents like headings, paragraphs, lists, and so forth to facilitate the sharing of scientific information between researchers.
- Elements are the individual components that make up the code, and include opening and closing tags and the content between them.
- HTML 4.01 was a large shake-up to the HTML standards that arrived in April 1998.
- A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.
- A CSS selector is the part of a CSS rule set that actually selects the content you want to style.
- CSS has an inheritance mechanism because otherwise CSS rules would be redundant.
- JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java.

KNOWLEDGE CHECK

1. **Language of document can be specified with tag**
 - a. lang
 - b. Ref
 - c. language
 - d. Href
2. **Style attribute of an element defines**
 - a. Size
 - b. Color
 - c. Font type
 - d. All of the Above
3. **In HTML, language of document is declared in tag**
 - a. <!DOCTYPE html>
 - b. <html>
 - c. <body>
 - d. <ref>
4. **CSS padding property is used for?**
 - a. Space
 - b. Border
 - c. Background color
 - d. Margin
5. **The selector is used to specify a style for a single, unique element**
 - a. Id
 - b. class
 - c. text
 - d. Bit
6. **Which of the following tag is used to insert a line-break in HTML?**
 - a.

 - b. <a>
 - c. <pre>
 - d.

7. How to create a hyperlink in HTML?

- a. ` javaTpoint.com `
- b. ``
- c. ` javaTpoint.com `
- d. `<a> www.javatpoint.com <javaTpoint.com /a>`

8. Which of the following is the correct syntax for referring the external style sheet?

- a. `<style src = example.css>`
- b. `<style src = "example.css" >`
- c. `<stylesheet> example.css </stylesheet>`
- d. `<link rel="stylesheet" type="text/css" href="example.css">`

9. The CSS property used to control the element's font-size is -

- a. text-style
- b. text-size
- c. font-size
- d. None of the above

10. In JavaScript, what is a block of statement?

- a. Conditional block
- b. block that combines a number of statements into a single compound statement
- c. both conditional block and a single statement
- d. block that contains a single statement

REVIEW QUESTIONS

1. What do you mean by HTML document?
2. Explain the various tags of HTML.
3. Describe the HTML page structure.
4. What do you understand by cascading style sheets (CSS)?
5. Discuss about advantages and disadvantage of CSS.
6. What is inheritance and Cascade?
7. What is JavaScript?
8. Discuss the JavaScript development tools.

Check Your Result

- | | | | | |
|--------|--------|--------|--------|---------|
| 1. (a) | 2. (d) | 3. (b) | 4. (a) | 5. (a) |
| 6. (a) | 7. (a) | 8. (d) | 9. (c) | 10. (b) |

REFERENCES

1. "HTML 4 – 4 Conformance: requirements and recommendations". Retrieved December 30, 2009.
2. "HTML 4.0 Specification". World Wide Web Consortium. April 24, 1998. Retrieved November 16, 2008.
3. "HTML 4.0 Specification". World Wide Web Consortium. December 18, 1997. Retrieved November 16, 2008.
4. "HTML 4.01 Specification". World Wide Web Consortium. December 24, 1999. Retrieved November 16, 2008.
5. Dan Cederholm (2009): Web Standards Solutions, The Markup and Style Handbook, Friends of Ed, ISBN 978-1430219200 (paperback) (Author's site)
6. Eric Meyer On CSS (2002), ISBN 0-7357-1245-X
7. Flanagan, David. JavaScript: The Definitive Guide. 7th edition. Sebastopol, California: O'Reilly, 2020.
8. Haverbeke, Marijn. Eloquent JavaScript. 3rd edition. No Starch Press, 2018. 472 pages. ISBN 978-1593279509.(download)
9. Jeffrey Zeldman (2009): Designing With Web Standards, New Riders, ISBN 978-0321616951 (paperback) (book's companion site)
10. Meyer, Eric A. (2001) Cascading Style Sheets 2.0 Programmer's Reference, McGraw-Hill Osborne Media, ISBN 0-07-213178-0
11. Meyer, Eric A. (2006). Cascading Style Sheets: The Definitive Guide, Third Edition. O'Reilly Media, Inc. ISBN 0-596-52733-0.
12. More Eric Meyer On CSS (2004) ISBN 0-7357-1425-8
13. The Zen of CSS Design (2005) (co-authored by CSS Zen Garden Owner, Dave Shea, and Molly E. Holzschlag), ISBN 0-321-30347-4
14. Zakas, Nicholas. Principles of Object-Oriented JavaScript, 1st edition. No Starch Press, 2014. 120 pages. ISBN 978-1593275402.

CHAPTER 5

MOBILE CONTENT STRATEGY

"Instead of one-way interruption, Web marketing is about delivering useful content at just the right moment that a buyer needs it."

– David Meerman Scott

LEARNING OBJECTIVES

After studying this chapter,
you will be able to:

1. Focus on mobile content marketing strategy
2. Discuss about mobile-first design



INTRODUCTION

There is such a thing as a content strategy that plans for how you'll publish and maintain your content across all these new and emerging platforms: smartphones and

tablets, sure, but also smart TVs, refrigerators, in-car audio systems—even the desktop web. But “holistic enterprise content strategy” just doesn’t have the same ring to it, right? Mobile’s the buzzword on everyone’s lips right now, so that’s the label we’ve slapped on this problem. When we talk about content strategy for mobile, we’re not talking about publishing different content to be read on smartphones. That wouldn’t be much of a strategy—who can afford to create content for only one platform? If content strategy means developing a plan for how you will create, deliver, maintain, and govern your content, then content strategy for mobile looks at the special challenges in getting your content onto a variety of devices, screen sizes, and platforms—including mobile web, native apps for iOS, Android and Windows, and, yup, even the desktop.

When we talk about content strategy for mobile, we’re also not talking about delivering content to serve the “mobile context.” “Mobile” seemingly implies motion, mobility. We imagine a hurried businesswoman, dashing through the airport, glancing at the screen out of the corner of one eye. But like the “dial” tone, the “return” key, and “cut and paste,” the word “mobile” has expanded to mean something different from its analogue in the physical world. Anyone who’s ever pecked at his mobile phone from the couch, too lazy to walk over to his desktop computer just a few feet away, knows exactly what we’re talking about. Anyone who’s ever waited for hours in that same airport, passing the time transfixed by a tiny glowing screen, knows the same thing. “Mobile” doesn’t necessarily mean you’re on the move.

5.1 IMPORTANCE OF MOBILE CONTENT MARKETING STRATEGY

It’s important to understand that a mobile content marketing strategy means more than having content that fits the typical screen on a mobile device. It means adapting your content to meet the wants and needs of mobile users. For some, a mobile content marketing strategy is synonymous with responsive design, a type of web design that allows a site to function well and look good on a variety of screen sizes.

While a responsive site can be part of a mobile content marketing strategy, it’s definitely not the entire kit and caboodle. An effective mobile content marketing strategy takes in all the ways that mobile use is different from desktop use.

With an effective mobile content marketing strategy, you are likely to rethink:

- *The format of your content:* Shorter is often better. Think bullet points, short paragraphs, and brief, attention-grabbing headlines.
- *The images you use:* Images need to load quickly and fit the screen size, otherwise people are just going to bounce.
- *The videos you embed:* Autoplay with sound is a no-no when you are putting mobile first. No one wants to be the person on the bus or in post office whose phone suddenly starts blasting music or talk.

- *Your keywords and SEO strategy:* Mobile search is a bit different from desktop search. For one thing, many mobile users use voice search. For another, many mobile users are looking for localized content or information.

5.1.1 Key to a Successful Mobile Content Strategy

Whether you are creating content for a blog, struggling to write a high-converting app description or optimizing text for App Store banners, there are several basic rules for building a successful mobile content strategy that will help you optimize each and every element on a user's journey.

Mobile has become the top way that many people to engage with content online.

As smartphones become ever more ubiquitous, your content is making its way to a whole new market. Because of this, you have to look at copywriting in a whole new way.

It's no secret that there's less screen real estate when it comes to mobile. Some marketers take that to mean they have to write less content to keep a reader's attention.

But this kind of thinking is a trap. However limited they are by technology, mobile readers are not substantially different from desktop users: quality matters more than quantity.

Remember

While more people than ever now use mobile devices when they're online, mobile users tend to have a higher bounce rate than other users. That is because, even after years of "mobile first" advice, many mobile sites still leave a lot to be desired.

Why Put Mobile First?

One of the big reasons to put mobile first is that people are using mobile now more than ever before even if there's a desktop device hanging around nearby. Way back in 2013, a study from Google and Nielsen found that 77 % of all mobile searches were conducted in an area where a desktop device was also available. The majority of people who used a mobile device even when they had a desktop available cited the ease and convenience of tapping out of a search query in a smartphone or tablet versus waiting for their computer to boot up. They load slowly, use annoying pop-ups that cover the entire screen and are difficult to close. Some websites seem to have never gotten the memo, and don't have responsive design or a mobile version at all, forcing mobile users to zoom in just

to read a bit of text. In content marketing, one of the things you should be aiming to do is set yourself apart from the competition. If your competitors seem to have buried their heads in the sand when it comes to a mobile strategy, being the one company that offers friendly, useful, mobile-optimized content will make sure you stand out.

5.1.2 Who Should Use a Mobile Content Marketing Strategy?

The average person spends about five hours a day on a mobile device. That's plenty of time for shopping, watching videos, playing games, or browsing on social media

Steps to Develop a Mobile Content Marketing Strategy

In many ways, creating a mobile content marketing strategy is similar to creating a general content marketing strategy. You want to have a goal for your strategy, and you want to have an idea of your audience. But in other ways, the process of putting together a mobile content marketing strategy is different.

There's what you can do to create your own mobile content marketing strategy:

- ***Set a goal:*** What do you want to accomplish by putting mobile first? Are you trying to drive more in-person foot traffic to a brick-and-mortar store? Do you want people to make purchases online? Are you hoping to get more people to go to your website?
- ***Find your audience:*** You might have a target audience in mind, or you might decide to tailor your content to an audience that's already checking out your site and content on a mobile device. Analytics tools can let you see what percentage of your audience is already on mobile, and give you basic information about them.
- ***Focus on the content:*** Yes, content should be responsive. But there's more to it than that. Think of how small the mobile screen is, and how unlikely a person is going to be to scroll. Then put the most important information at the top, or the information that you think will make someone want to scroll and learn more. Use bullet points and short headlines to make your content scan able. Include images, but make sure they are compressed and don't take forever to load.
- ***Optimize for mobile search and social:*** Remember to optimize for voice and social when developing a mobile content marketing strategy. Choose keywords that a person would be more likely to use in a voice search. Make your content easy to share on social by including big social share buttons.
- ***Measure and track your content's performance:*** Is your mobile strategy paying off? You have got to check to make sure. Measure the number of mobile visitors you get, how long they stick around and what actions they take afterward to get a sense of whether your mobile strategy is working or needs some tweaking.

- The great thing about using a mobile content marketing strategy is that it's often backwards compatible with a desktop strategy. While desktop users might have slightly different goals than mobile users, they will appreciate fast-loading, easy-to-scan content just as much.

5.1.3 Mobile Content Marketing

Consumers are using their smart devices for everything from watching sports to shopping online. Brands feel the need to design product experience for handheld users. Many are adopting the mobile content marketing approach. This approach places mobile at the helm and considers it more pertinent than desktop.

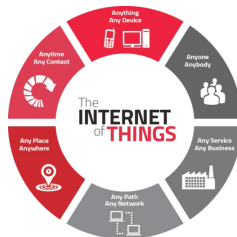
The Mobile Web

The desktop web is no longer a match for the mobile web. The latter outpaces the former in traffic and visitor retention rate. A study by Google and Nielsen shows more than 75% with access to handheld and desktop web, prefer the former.

Internet of Things

Because the crux of IoT is mobility, it adds up to the mobile web. “Things” may function as a doorway to the online world. But they can never perform multitasking. A camera can only capture stills; a car only serves the purpose of commuting.

- But mobile phones let us multitask, through apps of course.



Content Creation for “things” Should Take the Following into Consideration:

- *Data-rich:* Through built-in sensors, “things” cultivate raw data. Weather info, location, street address, share market tickers are raw data, displayable on websites. IoT-friendly content needs to be with rich with such data.
- *The format:* Files having .stl or .obj format do not run on browsers. Their front-end display is not as smooth as .avi or .mp4. Different formats are a challenge for mobile content marketing. It's a challenge because “things” can have content in such formats. For example, a 3D printer with an internet connection can contain a .stl file.

Keyword

Consumer is a person or organization that use economic services or commodities.

- *Customer experience:* “Things” create a network. This prompts brands to guide customers through every touch point. In a connected framework, the content needs to provide guidance, or else, it will fail.

Only the mobile content marketing approach can allow inbound marketers to catch up with IoT. Connected space, portable environment, small screen, and UX bottlenecks disturb mobile internet browsing experience. One faces the same set of challenges, along with a few other challenges when he moves to IoT.

Voice Search

Let’s say you are accessing Google from your desktop computer. And you want information on Kate Upton.

You type two search queries one after the other. The first query is “Kate Upton birthday.” The second query is “Kate Upton partner.” Both queries are independent. In the case of voice search, one search query can be the continuation of the other. If the first query is “When was Kate Upton born,” the second query can be “Does she have a boyfriend.”

Note “She” is a referring-back expression that connects phrases. There’s a takeaway for content marketers here. They should create content as answers to not just one search query, but a series of related queries. It’s easier said than done. Anticipating the queries requires foresight and creating content around them requires creative prowess.

How can the mobile content marketing strategy function as leverage here?

The desktop search leads to a loss-loss situation for a website and visitors. Miguel Salcido pointed it out. A visitor who lands on a page goes back to the SERP for the next search. This way, the ranking of the page drops for the query he used.

Voice-activated search creates a web of queries. Voice search signifies mobility as people can search something on the go. It is semantic search too. The voice search app for Chrome ramps up semantic searching.

Localized Content

Localized content is a prime requisite for an optimized mobile experience. Almost 50% consumers gather store location and other info through mobile search. And approximately 80% mobile searches convert into sales.

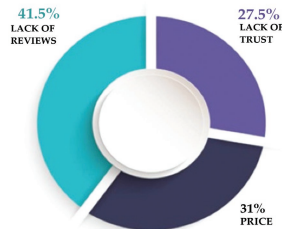
The mobile content marketing approach necessitates the creation of localized content. The key strategies include:

- **Topic update:** It's easy. Configure Google Alerts to know which topics are trending in which regions. Also, use Google Analytics to get location info of visitors. Select regional topics for regional visitors and create content around those topics.
- **Local keywords:** Local keywords are less competitive compared to global keywords. One should add such keywords to the content, to meta properties, and as ALT tags. It guarantees visibility.
- **Google map optimization:** Google Map listing is essential for a business. The Google Map API allows integration. A brand can create and integrate positive content into the map listing.

The mobile web eases localized content creation. But it demands a marketer to follow the said strategies. Hence, shift your attention from the desktop web to mobile. This way, you can harness the power of localization.

User-Generated Content

The retail segment is more sensitive to UGC than the brick-and-mortar segment. Restaurants and fashion outlets are vulnerable to user/consumer generated content.



Remember

You should focus on just one main idea in each chunk of text—whether that's a paragraph or a section). Readers are likely to scan headings and initial sentences, searching for words that they think will answer their questions. If you combine multiple ideas in a chunk with no visual separation or distinction between them, important information will likely be overlooked.

A survey on 500 consumers produced the data. Positive UGC seems to beat affordability in their priority checklist. And bear in mind they were all mobile shoppers. Positive UGC increases review conversion rate too. Shoppers used handheld devices to respond to review requests. They reviewed the brands from using those devices. The review conversion rate wouldn't have been high if shoppers responded via desktop devices. It's easy to access the web through Smart devices. Generating a review in favor of a brand is easy as there's an army of apps, custom-built for this purpose.



Remember

The mobile content marketing focus encapsulates all these consumer segments. The UGC offers consumers ease at generating content in favor of a brand.

The mobile content marketing approach taps into multi-screen marketing and get the best out of it. For dummies, multi-screen advertising is linked with an uptick in cross-device conversion. UGC can replace ads with the power of organic content. Organic content is more powerful than promotional content used in advertising. Online shopping is a multi-screen activity. Spur of the moment shopping accounts for 81% spontaneous buys. While smart devices trigger it, desktop devices function only as catalysts.

5.2 MOBILE-FIRST DESIGN

Mobile first design is a design strategy that says when you create a website or app, you should start sketching and prototyping the smallest screen first and work your way up to larger screens. Essentially, it's about delivering the right user experience to the right device. The reason that this makes sense is because with such limited real estate on small screens, UX designers must prioritize the most important aspects of their website and apps, namely content.

For many years, mobile websites were an afterthought to the design process. An addition not a necessity. The slow decline of desktop coupled with the rise of mobile phones over the last 5 years has shown that mobile first design has to take priority. The term refers to a website that allows a user the usability to do just about everything on a mobile device as they would on a desktop, with minimal effects on site loading time and functionality. While it may not always be possible for a mobile version of a website to mirror every single function of the desktop version, a mobile friendly site typically aims to meet the majority of its mobile users' needs.

While the desktop version of a website might have large hero images on the home page, the mobile version will likely scale down the size of these photos, or remove them altogether



Similarly, while a desktop version of a website might display photos horizontally across the page, the mobile version may display these photos vertically. When using this web design approach, you may lose some of the functionality a desktop provides, however they typically are functions that are not used as frequently or needed on mobile.

When mobile friendly design first became popular, many businesses were rushing to adapt their existing websites to meet the requirements for functionality. Websites that don't consider mobile users will likely see fewer and shorter visits, and lower engagement rates, as a result of the longer load times and limited functionality that mobile users will experience.

5.2.1 Benefits of Mobile First Design

Mobile conversions rates are up 64% when compared with the average desktop conversion rates. So designing mobile first can lead to more profit for your business. And since Google ranks for mobile-friendliness, it makes sense to bear this in mind when starting a new project.

This way of designing is also beneficial when it comes to download times and users accessing your content as quickly

as possible. With fewer elements, the page will load faster. When you consider a 1 second delay causing a 7% loss in conversions – it pays to design mobile first.

When you design from the smallest screen to the largest screen, this is known as progressive enhancement. It's about designing with a strong foundation and adding enhancements as you go.

With mobile first, you create your strong foundation. This foundation will help strengthen other designs for tablet and desktop. The foundation should always be content and mobile first design emphasizes content over navigation – users get the information they need quicker. This will also make your life easier since mobile first design starts off with the tackling the hardest screen size to design for. The rest inevitably falls into place.

Mobile first design forces you to really focus and maintain clarity by removing any unnecessary user interface decoration. By removing any distractions, you'll invariably improve the user experience and that makes good business sense.

If you want your users to have the best possible experience on mobile and desktop devices, designing for mobile first is the way to go. You won't have to worry about there being mobile feature constraints, or slower loading times when using a mobile device. Google shouldn't penalize you either, since your website will load fast from any viewing device.

Finally, taking a mobile first design approach allows you to think about what actually needs to be on your site. Instead of trying to fill in all the whitespace with fluff, you focus on what the user actually wants/needs to complete the action. In order to maintain optimal functionality, you'll work harder to make sure your site is full of only the most critical content and features necessary to help your business succeed.

5.2.2 Mobile first is Content First

Content takes center stage in mobile first design. With mobile first design, you have to give your users the content that they absolutely need. Designing this way, with such stringent limitations, forces UX designers to strip any extraneous elements away and focus on the essential.

When we talk of extraneous elements we don't mean that they are not necessary – they are just not necessary for your mobile users. You can still use any UI elements you remove from your mobile first design in a desktop version, for example.

This is because content is context dependent. A mobile user will have different needs than a desktop user. A desktop user may be looking for more in-depth information or additional features that would not make sense when it comes to mobile first design.

On a traditional desktop website, you are more likely to see white space whereas collapsible menus and widgets are more prominent in mobile sites. The same can be

said of photos – expect full size imagery on a desktop. You are talking advertisements and promotional material. This sort of imagery will be reduced (or even removed) on mobile websites.

Breakpoints are also important to mention. Normally you would define your breakpoints depending on the device.

For example, when you resize your browser from large to small, you will see the presentation of the content change at specific breakpoints – these breakpoints are used to change the content to fit the device. On our blog, for example, when you resize below 1223 pixels wide, the content will change to suit the newer size.

This approach becomes untenable when you consider the vast number of mobile devices on the market. Because of this, it's best to create content specific breakpoint. What does that mean? It means creating a breakpoint for when the content is no longer easy to consume. The best sort of design is one which solves future problems prematurely.



How to create mobile first design in Justinmind

Justinmind's new responsive features make creating a website that's mobile first a cakewalk. The new features include scalable UI elements. The new size attribute allows you to define an element's position by pixels as well as by percentage.

As such, UI elements will remain in place with a fixed percentage relative to the size of the canvas – so no matter how you resize or which device you view the content on, UI elements will stay in the same position. Consistent across all devices.

With the new Pin feature, you can stick UI elements on the canvas and they'll adapt depending on the screen size. Locked ratios mean your images maintain their clarity and size so you don't need to worry about creating awkward workarounds. And to save you time when crafting mobile first designs, you can create your own responsive widget libraries. If you need to use a widget time and time again with your own specific settings – save it. It'll maintain those responsive properties.

The new release comes with updated templates, including a responsive template for web which is designed with mobile first in mind. Create a new prototype and under templates you'll find a fully responsive example for you to tinker with and make your own. Explore the settings yourself and play around with the new properties to get a better feel for the mobile first design features included in the new version.

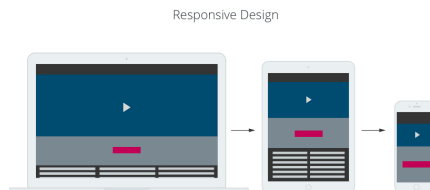
5.2.3 Difference between Mobile-first and Responsive Design

Firstly, there is often a little confusion when it comes to these two concepts. Many people mistakenly think they are one in the same when they are entirely different. Responsive design starts on the desktop; that is, at the maximum required resolution, and then scales down to the smallest screen. Even though the content and layout contract to fit smartphones, the navigation, content and download speeds are geared more for your traditional website.

Mobile-first design is similar to designing a mobile app and then adapting the layout that it can be viewed neatly on tablet and desktop devices without too many modifications. Your whole design and layout are based on providing excellent mobile user-experience: fast download speeds, rich media content to keep your target audience interested, easy touchscreen navigation and so on.

Keyword

HTML is the standard markup language for creating web pages and web applications.



Responsive web design not only encompasses the idea of a mobile friendly design, but is a broader term used to define the designing of web pages for a variety of screen sizes and orientations. When a website is built with responsive design features, the goal is to provide an optimal viewing experience for all users, no matter what type of device they are using.



The main difference between the two web design approaches is that mobile friendly design was originally seen as an afterthought or an add-on to an existing website. Responsive web design is a method used from the beginning of the web development process or in a website redesign. Although these terms are fairly similar, it's important to understand the differences between them when talking about the history of web design and the progression over time.

Why Responsive Design

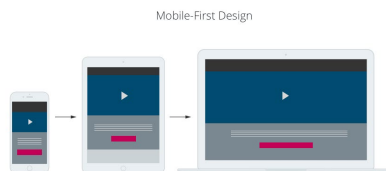
Generally, responsive design is more common among B2B companies where the website content needs to be informative and authoritative. Rich **HTML** content that is structured correctly is also great for SEO. As a digital agency, we are mainly targeting other businesses, and we know that 80% of users are on a laptop or desktop and they are accessing our site during office hours. As such, responsive design is the best approach. The Darwin Digital content, navigation, and layout are responsive to smartphones and tablets, providing an excellent UX for our mobile users

Pros

- Good for heavily stacked information websites
- Easier for large forms and complex call-to-actions
- Cost-effective development and maintenance
- Great for SEO

Cons

- The mobile experience is not 100% optimized



Keyword

Camera is an optical instrument for capturing still images or for recording moving images, which are stored in a physical medium such as in a digital system or on photographic film.

Why Mobile-First Design

Responsive design sounds good, we hear you say. Why should you risk a mobile-first design? Quite simply, the stats show that we have become addicted to surfing on mobile devices. Currently, 52.64% of the total traffic on the Internet is done via mobile phones, and by the end of the year experts from Zenith Media predict an increase of up to 75%. With this information in mind, it is essential to understand what mobile-first design is and what its benefits are. Unlike responsive design, mobile-first is about a complete mobile user-experience: adapted app-like user-interface, less text, larger fonts, fast download speed, video and audio, one call-to-action per page, short forms, and etcetera. Additionally, mobile browsers shortly will have access to more of the smartphone features like **camera**, haptic feedback, voice detection, so that a mobile-first design will be able to provide a unique experience to put your site ahead of the competition and drive traffic

Pros

- Better user-experience on mobile devices
- Majority of internet browsing in on a smartphone
- Design website to use built-in phone features
- Cheaper than building an iOS, Android or Hybrid App

Cons

- – The desktop experience is not 100% optimized
- – Not suited for content heavy websites



Mobile-First or Responsive

This is a relatively easy decision to make: use the 80-20 rule. If 80% of your target audience is on desktop use, responsive

design; if 80% are on mobile phone use mobile-first. Great you say, but what if my users are split 50-50? You need to see where the usage trends are going, what your budget is and where your business will be in 3 years. If your business is entirely digital and providing an excellent user-experience cannot be ignored, maybe you should create two distinct sites .adapted specially for both desktop and mobile users

Where can you get this info? Start by using your Google analytics. This will give you great information about how and when people are accessing your site. You should also research industry statistics for what the usage is for your market. For example, if you have a restaurant website in New York, then mobile usage should be well above 80%. Another great way to get information is to do some user testing and talk to as many of your clients as possible. At the end of the day, if someone browsing your website has a 'bad' user-experience, !there is only a 20% chance they will come back

How mobile-first Design is Different 5.2.4

With mobile-first design, there are not separate desktop and mobile versions. The design concentrates on the mobile experience and how a user would interact with content. Mobile-first means you are designing for touch, whereas starting with .a desktop is designing for the click

That doesn't mean that desktop isn't important. There are many occasions on which users do the research on mobile for convenience then make decisions or purchases on desktop. In these cases, mobile isn't the medium of conversion—but without its high-value UX, it would not have occurred. When a user later accesses your site on desktop, it will provide the same experience, just expanded.

Graceful Degradation and Progressive Enhancement

Graceful degradation describes the need to have a site function across a variety of device types and sizes. The mindset of this philosophy is to find the best design that covers the majority—accounting for the possible degradations along the way—and most importantly, to keep the site functioning. The desktop was the first design, removing certain blocks of content with each

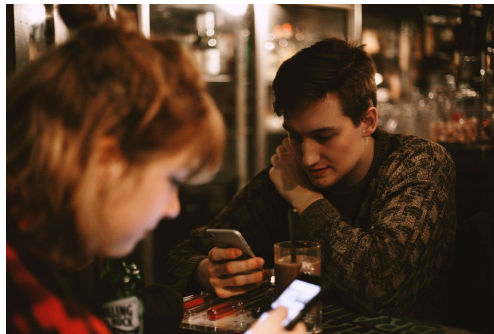
Did You Know?

The Philosophy of Data," Rachel Lovinger describes the goal of content strategy as using "words and data to create unambiguous content that supports meaningful, interactive experiences.

change in scale (as the screen size became smaller). Progressive enhancement is the approach that supports the mobile-first recommendation. Design for the mobile version first, then enhance as the size changes with new blocks of content. But progressive enhancement isn't the same as responsive design. Responsive design is the concept of media queries that target specific devices and viewport sizes. It works in tandem with progressive enhancement.

Mobile is where your audience

The overwhelming reason to start with mobile is that it's where your users are. Globally, the breakdown of online activity across devices is: 52.64% mobile, 42.75% desktop, and 4.62% tablet. Consumers use mobile to interact with a brand, read content, open emails, engage on social media, and make purchases.



You cannot screw up mobile UX

UX is complicated. It's not a cookie cutter element in design. Every page has a different objective. When you strip down to the absolute basics, as occurs in mobile-first design, you start building a UX foundation that is relevant. On mobile, the focus is on the most visited parts of your website and how to get there.

At the heart of mobile-first UX is navigation. It must be easy and intuitive. Because if your UX is not up to expectations, users will bail. Google estimates that 61 percent of users will not return to a site they had trouble accessing on mobile, and 40 percent of users will then visit a competitor's site instead. You cannot afford to not emphasize mobile UX if you want to keep users on your site.

Mobile-first is content-first, too

Not only is the UX stripped down but the content is as well. Whatever your product or service, to get mobile-first design right, it's got to be understood in a glance. This translates to several areas, including:

- Relevant, contextual imagery or video that clearly communicates visually what you do
- Direct copy that incorporates value propositions and concise language
- Options for the user to learn more or go deeper into the site (e.g., icons for products or services)
- Calls to action that provide the user an avenue to connect with you
- Help and contact information that's easy for users to find

But you have to fold all of that into a lot of limiting parameters. It helps you define the most important visuals and **messaging** that will add to the foundation you are building.

Mobile could be the only way users see your site

According to research, 25% of mobile users are mobile-only (they rarely or never use a desktop). That means a quarter of your possible audience will only see the mobile version of your site. This compelling data point supports the need to move forward with a mobile-first strategy. Think of it as just another way convenience matters to users. They can be anywhere and use a smartphone; desktop sessions require more effort. One experience across all devices matters — because many first visits begin with a mobile search. Currently, almost 60% of all searches are completed on mobile. Desktop experiences are an expanded version of mobile, not a different one.

Remember

Message is a discrete unit of communication intended by the source for consumption by some recipient or group of recipients.



App vs. mobile forward

There are differences in the UX and design for apps versus mobile-first websites. They are both focused on touch interaction, but they have a different function. Apps usually have even fewer features than mobile, meaning certain actions a user would want to take are not available in the app. Apps focus on repeatable actions or tasks, like transferring money or counting calories. So app design has an even smaller concentration that's very specific. In comparison, the U.S. Mobile App Report found that the number of mobile browser sessions is two times larger than that of app audiences.

The answer is to create an app-like experience. This type of design drives returns and facilitates organic growth. A mobile design that takes on these characteristics allows users to maneuver and convert easily. When users enjoy a great experience, they will come back. As new features or updates are made, returning users will continue to appreciate such a seamless process.

How to be mobile-first

When you make the decision to design with a mobile-first approach, there are a number of factors that must be present and strategies that must shift. If you have been in a desktop-first design mode, it's time to shift your thinking around design and UX.

Your customer is what's driving the functionality and experience of your mobile website. Users often won't tolerate "different" experiences across platforms. If the mobile site offers a promotion or allows for a way to sort or filter results, then that same experience must transfer to desktop. Otherwise, customers will be confused. This turns into an expectation gap. You will let your customers down and they'll move on to another site. Be prepared and proactive in updating and upgrading your mobile experience. Your visitors will thank you for it.

Get started with mobile-first design with Top coder

With Top-coder's community of hundreds of thousands of designers, you'll see multiple design options from top-rated designers all with different influences and perspectives. You can expect smart, UX-friendly, mobile-first design.

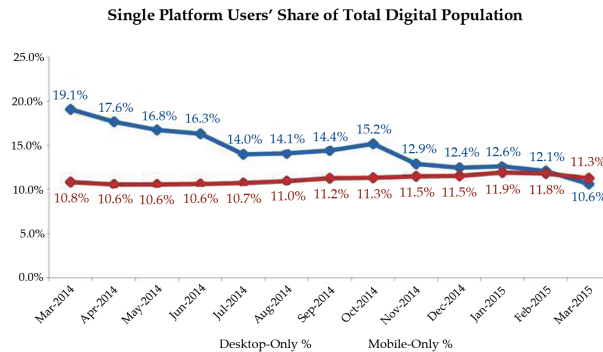
5.2.5 Mobile First Design Methodology

For a long time, website design first focused on creating the most optimal user experience for the largest screen size a user would be on, such as a desktop screen. Designing for mobile first takes the age old habit of designing for the biggest possible screen first, and designing for the smallest possible screen first.

With the rise in popularity of mobile website viewing, mobile web design provided an alternative to the main desktop viewing experience. Thus, responsive web design

was born as a solution to having to create multiple web designs – one for desktop, one for mobile, etc.

According to recent research done by comScore, “desktop only has been on a pretty steady decline since 2014, dropping from 19.1% to 10.6%. On the other hand, mobile only has been on the rise with no expectations to slow down. Considering the trend, desktop viewing may no longer be the norm. Therefore, it’s important to optimize on the devices your audience is using, and not those they aren’t.



Some aspects of a website that are designed for the desktop simply cannot be recreated on a mobile screen. So why would not you design specifically for a mobile screen? Anything that can be viewed on a mobile version can also be viewed on a desktop version. Instead of trying to fix the issues that often accompany trying to scale a large screen experience down to a small screen experience, starting with mobile works within the limitations of the smaller screen from the beginning.

SUMMARY

- If content strategy means developing a plan for how you will create, deliver, maintain, and govern your content, then content strategy for mobile looks at the special challenges in getting your content onto a variety of devices, screen sizes, and platforms—including mobile web, native apps for iOS, Android and Windows, and, yup, even the desktop.
- Mobile has become the top way that many people to engage with content online. As smartphones become ever more ubiquitous, your content is making its way to a whole new market.
- Voice-activated search creates a web of queries. Voice search signifies mobility as people can search something on the go.
- When mobile friendly design first became popular, many businesses were rushing to adapt their existing websites to meet the requirements for functionality.
- Mobile-first design is similar to designing a mobile app and then adapting the layout that it can be viewed neatly on tablet and desktop devices without too many modifications.

KNOWLEDGE CHECK

1. **Mobile marketing has innovative ways to reach the consumer. Which of the following is not one of them?**
 - a. Mobile retail payments
 - b. Barcode calls-to-action
 - c. Mobile apps
 - d. Yellow pages advertising
2. **Geotargeting allows an advertising campaign to concentrate on a fixed locale through mobile technology**
 - a. True
 - b. False
3. **One advantage of Mobile marketing is that it is not**
 - a. Inferior in its creative possibilities
 - b. Prone to security breaches
 - c. Dependent on GPS systems
 - d. Place-based media
4. **Mobile marketers are able to reach audiences**
 - a. Using a 'push' strategy
 - b. Using a 'pull' strategy
 - c. In real time
 - d. All of the above
5. **The statistics on unique visitors on a mobile device are highly reliable**
 - a. True
 - b. False
6. **`When should you start considering mobile app development for your business?**
 - a. When your customers or employees ask for apps
 - b. When your mobile Web experience lacks features or offers a poor user experience
 - c. When competitors release their own mobile apps
 - d. Any of the above
7. **Mobile apps are a good choice for every company.**
 - a. True
 - b. False

8. **If you're planning to build a mobile app, which of these issues is most important?**
 - a. Platform
 - b. Security
 - c. Usability
 - d. All three
9. **What is the most effective way for a development team to manage the constant updates necessary for business-critical mobile apps?**
 - a. Assign a team to push out updates on a regular basis
 - b. Draw straws to determine which team has to publish the next update
 - c. Automate deployment so finished updates go out on a schedule
 - d. Farm out update duties to a third party
10. **Aside from deployment, what other parts of the application lifecycle can be automated?**
 - a. Delivering failed test reports to the developer's inbox
 - b. Moving code to test environments
 - c. Coding the app
 - d. A and C

REVIEW QUESTIONS

1. Discuss the key to a successful mobile content strategy.
2. Who should use a mobile content marketing strategy?
3. What do you understand by mobile content marketing?
4. Focus benefits of mobile first design.
5. Explain the mobile first design methodology.

Check Your Result

- | | | | | |
|--------|--------|--------|--------|---------|
| 1. (d) | 2. (a) | 3. (d) | 4. (d) | 5. (a) |
| 6. (d) | 7. (b) | 8. (c) | 9. (c) | 10. (d) |

REFERENCES

1. "Content Strategy Alliance Charter". 5 July 2014. Retrieved 21 May 2019.
2. "Why You Need Two Types of Content Strategist". 2016-02-22. Retrieved 2016-09-02.
3. Domingo Karsten. "A Look at Internet History: The Most Popular Websites from the Past". tech.co.
4. Erin Scime (8 December 2009). "The Content Strategist as Digital Curator". A List Apart.
5. Halvorson, Kristina and Melissa Rach. Content Strategy for the Web (2nd Edition)
6. Halvorson, Kristina. The Discipline of Content Strategy <http://www.alistapart.com/articles/thedisciplineofcontentstrategy>
7. Jones, Coleen. Clout: The Art and Science of Influential Web Content
8. Kevin P Nichols and Anne Casson. "2013 SapientNitro Content Strategy Positioning". Sapient, Inc.
9. Kristina Halvorson. "The Discipline of Content Strategy". AListApart.com.
10. Lovering, Rachel. "Content Strategy: The Philosophy of Data". Boxes and Arrows. Retrieved 17 November 2019.
11. Redish, Ginny. Letting Go of the Words: Writing Web Content that Works
12. Sheffield, Richard (2009). The Web Content Strategist's Bible, p.35. Cluefox Publishing, Atlanta. ISBN 978-1-4414-8262-4

CHAPTER 6

STYLE TILES AND WEB DESIGN TECHNIQUES

“Graphic design will save the world right after rock and roll does.”

– David Carson

LEARNING OBJECTIVES

After studying this chapter,
you will be able to:

1. Discuss the concept of web technologies
2. Explain the client and server scripting languages



INTRODUCTION

The style tile is a design deliverable that references website interface elements through font, color, and style collections delivered alongside a site map, wireframes, and other user

experience artifacts. Style tiles are based on visual preference discussions with the client. They're sample options that spur discussion with stakeholders on a common visual language. Containing sample UI style swatches, a style tile illustrates how a designer translates a stakeholder's brand to the web. When a client uses the word "friendly" or "clean" to describe the site they want, the style tile visually represents those adjectives. Style tiles offer a catalyst for discussions to clarify and refine the client's preferences and goals. Style tiles are a flexible starting point that define a style to communicate the web in a way that clients understand. A style tile is more refined than a traditional identity mood board and less detailed than a website mockup or comp. When an interior designer redesigns a room they don't build multiple options of the designs they're proposing, they bring color swatches, paint chips, and architectural drawings. Style tiles act as paint chips and color swatches for the interface that we can execute on any device or at any dimension. It's a truly responsive solution to visual design. A mood board can provide a great jumping-off point for client discussion, but is often too vague to help clients make a clear leap from discussion to website. Mood boards are a good way to dig deep into a brand identity, but when it comes to bringing the identity to a complex web system, such a weak connection can make it hard for a client to understand and imagine the outcome. By contrast, style tiles make a great visual design artifact. They help a designer communicate how they will apply the styles across a larger web system, which includes desktop and mobile experiences.

6.1 CONCEPT OF WEB TECHNOLOGIES

Web technology is the establishment and use of mechanisms that make it possible for different computers to communicate and share resources. Web technologies are infrastructural building blocks of any effective computer network: Local Area Network (LAN), Metropolitan Area Network (MAN) or a Wide Area Network (WAN), such as the Internet. Communication on a computer could never be as effective as they are without the plethora of web technologies in existence. Web technology is the development of the mechanism that allows two or more computer devices to communicate over a network. For instance, in a typical office setting, a number of computers plus additional devices such as printers may be interconnected via a network, allowing for quick and convenient transmission of information. The processes involved in web technology are complex and diverse, which is why major businesses employ whole departments to deal with the issue. Web technology has revolutionized communication methods and has made operations far more efficient. The main advantage of web technology is that it offers convenience and a high speed of communication in the computer world. Whether in the office or the home, processes using a computer are more swift and straightforward with the use of a network. Web technology allows messages to be sent around a system, whereas before it may have been necessary to employ a runner or leave your workspace to communicate a message. It is clear to see how web technology reduces costs and makes a company more efficient, raising business potential.

Matters involving web technology can be very complicated, and it would be difficult for someone without relevant experience to sort a network problem out. This means it is necessary to employ someone with the specific skills to solve network issues, which costs money. Additionally, the existence of a network provides the opportunity for an attack on the computer system. Weaknesses in a network could be exploited; important information could be stolen or destroyed and malware could infect the various network systems. For this reason, network security is another issue that must be considered when using web technology.

6.1.1 What are Style Tiles?

Style tiles are a visual reference to the design language of a website (or other design deliverable). They help tell a story through fonts, color and style collections, and when viewed in combination with wireframes, site-maps and other UI elements, they define that story in an accessible, client-friendly manner.

Style tiles are a design deliverable consisting of fonts, colors and interface elements that communicate the essence of a visual brand for the web. They help form a common visual language between the designers and the stakeholders and provide a catalyst for discussions around the preferences and goals of the client.

Style Tiles are similar to the paint chips and fabric swatches an interior designer gets approval on before designing a room. An interior designer does not design three different rooms for a client at the first kick-off meeting, so why do Web designers design three different webpage mockups?

Website layouts are mostly based around content. A style tile is not great for explaining how content fits together but it's perfect for explaining everything else. Colors, icons, typography, textures, page elements, you name it.

SCS Mood Board

Logo



Colors



Font

abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ

Influences



The purpose of a style tile is not to plan out the full mockup. It's just a roadmap to help you reach the destination with less hassle. These tiles cannot replace wireframes or mockups. But they can be used for generating ideas and gluing the pieces together.

Adjectives

Start with the descriptive adjectives. Start creating style tiles after the student has completed research and a competitive analysis for the project. The student should have some idea for basic adjectives to describe the business. You have found it helpful to look at several websites and discuss possible adjectives to describe the brand or website so students get a feel for what descriptive adjectives are.

Fonts

What fonts represent the general tone of the site? Again, beginner web designers may not know how to choose a typeface. Consider assigning a reading about this. You have provided resources in this download for you.

Color

For many design students, choosing color is very difficult. Ask students to step beyond white, black and red (the most common “fallback” color palette) to working with color in an interesting way. Adobe Color CC is very helpful for this and can help students select a color palette. Or, you can have the student draw inspiration for a color palette from another website using Stylify Me.

Textures

You like to change this section to “imagery” and ask students to find examples of the style of images they may want to incorporate into the site. Please feel free to share the free image resources located in the download.

Logo

The logo section is the most difficult because logo and branding for the site is an entire project or class on its own! I encourage students to create the logo (for now) as a word-mark and use type only. Some design students may feel more comfortable with branding and may want to spend more time designing or redesigning a logo.

Flexibility for Mockups

By creating a style tile first you give yourself flexibility to change. It’s easier to make changes on a small UI kit rather than a full layout. These quick graphics and ideas give you points of reference for what the layout *could* become, without committing to the full layout.

Also keep in mind that most websites nowadays are fully responsive. It takes a lot of work to craft both full-width and responsive-width mockups. But style tiles do not require such considerations because they focus on smaller pieces of the puzzle. Clients also seem to respond positively to style tiles, assuming you explain what they are first. When clients expect a simple mood board they are more open to direct critique from an early-stage thought process.

Tile Design Tips and Trends

When it comes to actually designing a style tile you need to consider the overall composition along with the internal elements. Some tile designers like to create mini-headers with the website's logo, color scheme, and tile version number.

The branding should be recognizable and explain that the document is only a style guide. The version number helps clients reference different versions based on what they like.

However the website's branding should only take up a small portion of the design. Style tiles are meant to convey ideas about the interface. Naturally this would include headers, paragraphs, links, buttons, and icons.

Gathering Client Feedback

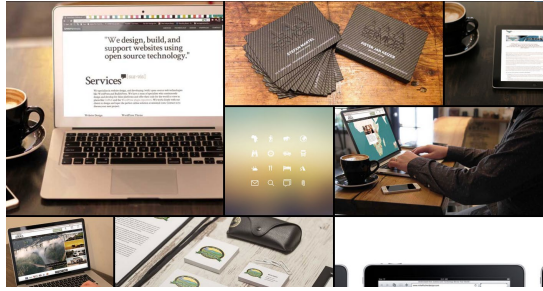
Before sending out any tiles go over your work and reaffirm the direction. Try to complete 2 or 3 different versions and see which elements draw the most attention.

Clients know what they want when they see it but do not always know how to say it in words. Offering too many choices will create choice paralysis. Instead opt for a lower number (2 is great) and gauge their reaction.

Regarding the total number of elements to use in a tile that really depends on the project. Some websites only need a small number of elements while other projects may require a full UI kit. At the very least include some basic branding, colors, typography, and vital elements for the webpage.

6.1.2 Use Style Tiles

Style tiles are for when a mood board is too vague and a comp is too literal. Style tiles establish a direct connection with actual interface elements without defining layout. They work well for clients who have established brands and need them to translate smoothly to the web. Whereas the word "mood" is often associated with brand and identity design, the word "style" was chosen to mirror "cascading style sheets" and reinforce that Style Tiles are specific to Web design.



How to use Style Tiles Successfully

Once you understand the principles of style tiles, they should fit neatly into your design process. Typically they will be generated earlier in the design lifecycle than you might have created your first mockups, and as a result they will enable you to get feedback sooner. Follow the four following steps to put Style Tiles into your workflow:

Listen to your client

Meet with your client at the start of your project, and gather as much relevant information as possible. Some designers may find it easier to conduct individual interviews with stakeholders, while others will prefer a group meeting; do what works for you. Make sure you ask all the key decision-makers and stakeholders the same core questions. If it helps, you can create a questionnaire for your clients ahead of the first project meeting. If you are struggling to know what questions you should be asking your client, the StyleTil.es website highlights some useful resources including these from Andy Rutledge and Kim Cullen.

Use a design framework to interpret the brief

Using the adjectives you recorded in the first step, create an aggregated set to identify themes in the adjective language. Apply these themes to your design framework, using design principles such as harmony and unity to make choices about the elements of design and their visual representation of the story. Don't forget to use any existing branding, hand-writing and **visual language** that's already in place to help contextualize the design brief and your response.

Define the visual language

Using the information you gathered in step 01, and the output generated through your interpretation in step 02, determine over-arching themes and start to match these up with stylistic devices. Generate a different tile for each visual interpretation you came up with

The completed style tiles will allow you to quickly establish whether you are on the same page as your client from a visual language perspective. The styletil.es website, created by Samantha Warren, provides a handy PSD template for you to download and use as a starting point for your style tiles output.

Iterate until style is agreed

Work with your client to make changes to the style tile. Because you are iterating outside the context of the final website layout, you can focus on ensuring your stylistic approach continues to reflect the business goals and design goals.

When to use style tiles

Style tiles offer a flexible starting point to help define what a website will look like, in an accessible way that clients can understand. Think of them as something in between a mood board and a mockup - they help define and communicate how a visual style will be applied across a website, including both desktop and mobile orientated user interfaces.

You can use them in place of the traditional first mockups, generating ideas and feedback from the client to help inform the final design approach.

Should style tiles replace mock ups?

It's a common approach at the start of a design project to produce page mock ups, but if you have ever found yourself having to negotiate over a hybrid comprising elements from two or more mock ups, and compromising on the design approach you will be familiar with the frustration that using mock ups early in the design process can bring.

From a client perspective, a mock up represents a complete design solution, but as it's often the first time they have seen your interpretation of the design brief, it's understandable that they will identify elements they like, and elements they do not. It's also a natural conclusion that the feedback you receive will compel you to bring together disparate elements from the various mock ups, leading to what Samantha Warren brands a 'Franken-comp' solution.

Keyword

The visual language is a system of communication using visual elements.

Remember

The website's branding should only take up a small portion of the design. Style tiles are meant to convey ideas about the interface. Naturally this would include headers, paragraphs, links, buttons, and icons. But also consider adding smaller elements like nav items and form fields.



Mock ups might seem like a simple, obvious way to define how a project can start the process of moving from the design brief to completion. But style tiles can be a smarter way of going about things, getting feedback, and define a visual language in collaboration with your client early on.

This can help avoid the feeling of frustration and disappointment in having to assemble a final mock up from individual parts of completely disconnected comps, and helps to keep the client fully engaged and bought-in to the design process.

6.1.3 Web Design Techniques

A better web means a better experience for everyone. Users should be able to reach what they need without worrying about how they are accessing the internet. Developers and designers should be creating sites that can be smoothly maintained and updated. A better web does more than just work, it works well – for those who consume it and those who build it. A better web is a more efficient web. Efficiency is at the heart of our projects, whether it takes the form of accessibility, responsiveness, content strategy or page load performance. From choosing the right CSS property to asking yourself the crucial questions that will make a whole project viable, efficiency is not just about going faster, it's about making things better. It's about, well, empathy.

We reached out to web designers and developers far and wide to hear their top tips and advice. Their suggestions span a wide variety of topics, all with a better, more efficient web at heart. Each of these tips is a door onto a new mindset that we invite you to consider and explore.

Ask questions

“Who are you building it for?” says UX designer Dylan Wilbanks. “Why do they need it? Where will they use it? What devices will they use? Have you talked to them? If not, who has? How will you know when it's done? What's the test plan? Is anyone building a prototype? Do you understand what you are being asked to build? Do you have answers to any of these questions? No? Then get them answered before you build.”

Listen to your code

“Screen readers present your web page exactly as the HTML is written,” says information architect Anne Gibson. “So do search engines. Writing semantic HTML improves the experience for both, and includes a host of other benefits.

“Try checking the semantic quality of your pages by listening to them on a screen reader. If your page sounds out of order or clogged with detritus, clean up your HTML. Screen readers can be more than ‘just’ accessibility tools. They can be used to help us write cleaner, more semantic, easier-to-maintain markup.”

Protect the web's inherent accessibility

"The web is an accessible medium," says co-founder of Perch Rachel Andrew. "An HTML document can be accessed by anyone, on any device, on the slowest of connections. Every choice you make as a designer or developer can protect that inherent accessibility, or it can damage it. That's a lot of power you have there. Use it wisely."

Take advantage of conditional loading

"Every web page consists of 'the thing' and a bunch of stuff that's not 'the thing'," says well-known web designer Brad Frost. Conditional loading is an essential tool for creating feature-rich experiences that still load fast and prioritise core content. Look for opportunities to conditionally load social buttons, comments, related content and more.

Use Ceaser to write cubic-bezier

"CSS transitions can be simpler to deal with than keyframe animations," says Jamie Kosoy, president and head of technology at Arbitrary. "But customizing them with timing functions can feel pretty daunting.

transition-timing-function: cubic-bezier(0.250, 0.250, 0.750, 0.750); /* linear easing */

"If writing out Bézier tweens by hand is not your style, have no fear. With Matthew Lein's animation tool Ceaser, you can quickly use one of Robert Penner's easing equations or build your own. Animate away!

Use height-based media queries

"Make use of height-based media queries," advises Dan Denney, a frontend developer at Code School. "These are great opportunities to improve experiences when the available height is limited. It varies according to design, but with the right adjustments you can often reduce the need for scrolling and improve readability. If the design has interactions or notifications, be sure to adjust them for limited heights as well. See an example [here](#)."

Keep the big picture in mind

"Do not make your technical choices before you understand anything about the problems you should be solving," says Sally Jenkinson, consultant and solutions architect at Records Sound the Same. "Introducing solutions too early can mean experiences end up being dictated or limited by technology, both for your visitors and administrators. Instead, make your selections with the big picture and your wider project team in mind.

"Many technologies can do everything and anything, but should they? What's the knock-on impact on people, processes, technology and experiences?"

Use namespaceing for your CSS files

“CSS file names are a glossed-over opportunity,” stresses entrepreneur and systems designer Claudina Sarahe. “Namespaceing provides helpful clues to the contents of the file. If a partial does not produce output, append .config.scss; if a file imports other partials, append .manifest.scss.”

Employ easy image fallbacks

“Responsive images are complicated, but while the RICG was getting the specification together, we found some opportunities for quick wins – brief syntaxes that could make a big difference for users,” says Mat Marquis, an open web engineer at Bocoup. “One of those was the addition of a type attribute on source, allowing you to serve alternate image formats with responsible fallbacks:

```
<picture>
  <source type="image/webp" srcset="image.webp">
  
</picture>
```

“WebP for browsers that support WebP and a plain ol’ PNG for images that do not, and just one request either way.”

Master debugging in JavaScript

“Never underestimate the power of the console.log,” says Raquel Vélez, senior software engineer at npm. “When trying to understand a specific line, tag it with a useful piece of information. Something like console.log(‘foo; a=’, a) will help you know what a is with respect to a function foo. When trying to isolate a bug, log everywhere that you think makes sense, with console.log(‘boom1’), console.log(‘boom2’) and so on, so you can see exactly where your code is breaking.”

Make your background images responsive

“For beautiful and responsive background images, regardless of a screen’s width,” says designer Julie Ann Horvath, “use the CSS property background-size:cover;. In addition to visual scalability, using background-size: cover; when specifying the size of a background element automates the pixel ratio the image will display at, without having to specify it in additional rules or markup.”

Use mixins to create complex grid systems

“My students were learning how to build grid systems, as well as the power of using Sass mixins,” says Sam Kapila, design instructor at The Iron Yard. “So as a class, we

built a mixin that would help create a flexible grid system. You can find it here on CodePen.

“At the first breakpoint, the grid system can have any number of columns, but the mixin itself will add the total number of columns with the total number of gutters (which is less one gutter than column).

Then a margin-right is added to the floated column. The mixin is variable-dependent – specifically column width and gutter width. Finally, a :last-of-type resets the margin-right for the last column in each row.”

Think about conversations, not pages

“Thinking mobile-first is a useful way to approach the prioritization of page content, and to decide which parts are critical and which can be loaded conditionally later,” says graphic designer and frontend developer Paul Robert Lloyd. “However, this exercise still assumes the presence of an interface. To build truly universal and accessible websites, we need to look beyond visual appearance. An alternative approach might be to think about pages as being part of a phone conversation; what information might a user request, and how might the web page respond?”

Provide keyboard support

“When building interactive interfaces, remember to provide keyboard support,” says Monika Piotrowicz, frontend development lead at Shopify. “Mouse events can also bind to keyboard events, keyboard focus should update as new content is shown or hidden, and the escape key can help cancel interactions. By taking these steps, keyboard users can get the same great experience as anyone using a mouse! Check out this demo.”

Design quickly in the browser

“A nice way to design new features in-browser is to sketch the new thing directly in the code, to show to others,” says designer and frontend developer Frances Berriman. “Mark the parent element of the new feature with a class like sketchy, letting you apply styles to it to show it’s a work in progress, or quickly comment all the sketchy bits out at once for a clean product demo.”

Simplify your code

“Have you ever needed to return a value, but only if another test returns true?” asks coder and designer Sacha Greif. “For example, you might write something like:

```
if (user.isAdmin()) {  
    return password;
```

```
} else {  
    return false;  
}
```

“You can actually simplify this into:

```
return user.isAdmin() && password;
```

“If `user.isAdmin()` returns false, the whole expression will return false. However, if `user.isAdmin()` returns true, the expression will return the value of `password`.”

Convert fixed widths to percentages

“A simple way to start transitioning a site from fixed-width to a responsive format is by converting all its widths to percentages and removing floats,” says Inayaili de León Persson, lead web designer at Canonical. “If your main container is 900px wide, change that to a max-width instead – change columns and other elements to the percentage they take up on the page rather than pixel units, and so on.

Embrace the title field

“Nailing the title field can be an easy way to classify on the backend and clarify on the frontend,” advises Corey Vilhauer, UX strategist at Blend Interactive. “Make things easier for editors by being clear what each title field means. One should be for navigation (‘Classes’). One should represent the page’s title, which will populate the `<title>` (‘Our Classes’). And one should be used in the CMS to help keep things in order (‘Class Listing’). Mix and match as your organisation requires.”

Configure Git

“Save time by configuring Git,” advises Stacey Mulcahy, senior technical evangelist for Microsoft. “Configure a global ignore file for files you will never want to commit. Place a `gitignore` file in `~/.` and then you can `$ git config--global core.excludesfile ~/.gitignore`.”

“As new technology appears to streamline our workflow, we begin to adopt more and more dependencies,” says Brian Suda, owner of optional.is. “How many people are regularly using Less or Sass to manage their styles? Or tools like JSLint for JavaScript? Tools to concatenate, compress and cache your assets?

“Inevitably we forget to run some commands and outdated files get saved into repos or pushed to production. We use Git commit hooks so we never forget. By setting up some pre-commit hooks, all of these commands are run each time we submit our changes.”

Learn from issues and errors

No matter how many times you have tested your application, and how many devices you have used, there will be unexpected behaviours,” says developer Alex Duloz. “You will never, ever launch an application that does not contain bugs, so make it easy for your users to let you know about any issues they encounter while using your website. Leverage the GitHub API from your own website using a simple form. Log server errors and analyse them. And use something as trivial as `window.onerror` to keep track of JavaScript errors (or go for a `try {} catch(e) {}` at the entry point of your code). Take a deep breath, go live, and perfect your work based on that wisdom.”

Feel how fast your site is

“Waterfalls are not the only way to visualise your site’s performance,” says Lara Hogan, senior engineering manager of performance at Etsy. “On Webpagetest.org, you can create a video of how your site loads over time – simply check the ‘Capture Video’ box in advanced settings. This will help you feel how fast your site is from different locations and on different devices, and you can see what your users are truly experiencing.”

Avoid long select boxes

Stop using select elements for date of birth,” says Alice Bartlett, frontend developer for GOV.UK. “Providing a date of birth is part of many basic transactions on the web. It’s common to see select boxes used for dates of birth, but research shows that long selects are very difficult for those who are not confident with technology to use. Text inputs with server-side validation allow users with low technical confidence to transfer their understanding of paper forms onto web forms

;Scale icons with background-image: cover

If you are using the `background-image` CSS property to add and style an icon image,” says freelance designer Julie Ann Horvath, “another option for scalability is `background-image: contain`. While in theory you could use `background-image: cover`; if the element containing the icon has rounded corners, `cover` may cut off parts of the icon image, `contain` will size it appropriately, and to scale

Keep it short and sweet

Be succinct in your emails and robust in your code comments,” says information architect Anne Gibson. “If you need practice in writing ‘short and sweet’, join the development community on Twitter

6.1.4 Effective Web Designing Techniques and Their Importance for Content Gurus

It is of critical importance to design an exceptional website that would help you stand apart from the rest and stay way ahead of your competitors. A wonderfully designed website however, is not just enough; a good website should assist you in building your brand as well as business strategy.

An efficient team would be delivering personalized design concepts for literally every project. Your website should be suited to your business model. Whether your company is looking for a communication platform or sales focused website, competent designers today use suitable tools to create a site that certainly enhances your online presence and definitely takes your business forward.

Responsive Web Design Strategy

Thanks to the incredible rise in the number of mobile devices, responsive web designing is not anymore a UX-only discipline. All marketers need to understand and consider responsive web designing as a component of their strategic planning. It may not be necessary for you as a content marketer to know all the tricks of the designing and development of a responsive web. However, you need to know what exactly RWD is. Also, you need to know your exact content marketing responsibilities.



The web world is said to have undergone some technological changes and responsive web design is at the forefront as far as content marketers are concerned. Responsive web design implies you can write content and publish it once. The layout actually changes according to the capabilities and size of the device.

There has been an astronomical rise in web surfing through mobile devices. You are confronted with a multitude of diverse screen sizes across tablets, smartphones, phablets, desktops, TVs, consoles and even wearable devices like smart watches. As the screen size is constantly changing, you need to optimize your website for all these mobile devices. Your website should be designed in such a manner that it should be able to adapt to whatever screen size.

Maintaining Consistency

It is a good idea to keep elements such as color, layout or font fairly consistent in your site. Your site should have a smooth flow from one page to another. This means that your font, color and layout structure should be basically the same throughout your site to maintain consistency. It is important to keep the elements across all pages constant so that the viewers do not feel lost.

No Stock Photography

People are no longer interested in the glamour shots. They are looking for definitely more realistic view of exactly what a product is portraying or representing. Dry stock photos with dull white background are definitely not in vogue. People are looking for story and personality. Use only meaningful images in your website. Your images generate subconscious messages that are transmitted to your audience.

**You don't have to use
cheesy stock photos.**



Using Responsive Images

Designers can now create responsive layouts which are known to serve various image sizes of different resolutions. Designers are now able to develop mobile-optimized images that are ideal for smaller screens and they can then come up with higher-resolution versions for larger screens. You may use JavaScript and htaccess files for different sized images according to screen width. You could also, use tools such as TinySrc that lets you only prefix all large pictures with a TinySrc URL and the rest is left to the tool to do.

CSS Media Queries

You could follow a tutorial from CSS-Tricks, which discusses ways to bring about subtle modifications with media queries and ways of using media queries on a single style-sheet. *For example*, in case of a fluid-width design with a sidebar that is 35 % of the actual width of the page, then based on the actual width of the browser window,

one could tell what to do if the browser is very narrow, or what to do if it is wider, and also, what exactly to do if it is really quite wide.



Using Responsive Data Tables

Data tables are often pretty wide. You may see the whole table by zooming out, but in that case, you would need to scroll both horizontally and vertically to be able to browse the table. One best solution seems to be reformatting the table for much better readability. Yet another way out is to display a pie-graph according to the data. You may even consider adapting the table into an effective mini-graphic which is just right for narrow screens, (instead of interfering with the content, while the full table is on display.)

Design a Great Navigation System

The best way to engage visitors in your site is by developing a solid and impressive navigation system that would effectively support all search preferences. The most vital factor in web design is the ease with which one can find information. If visitors are not able to find the information they are looking for, they would lose interest and leave. The navigation system should be intuitive and self-explanatory.

- Your primary navigation structure should be simple.
- Always integrate navigation controls into the footer of your site.
- Breadcrumbs should be kept on every page barring the homepage so people are well aware of their exact navigation trail.
- Provide visitors with an option to search your site using keywords, by adding a visible search box towards the top.
- Minimize the provision of unnecessary navigation options as far as possible.
- It is a good idea to keep three levels of navigation, and not too many.
- Should you include links in your webpage, clearly mention and explain their destinations, and this is also very effective for SEO.

- Prepare your site keeping your target audience in mind.
- Ensure that the overall navigation of the site is simple, so that users can find their way to what they want without running through a complicated procedure. Ensure that deliberation on the part of the user is minimized by integrating a simple interface.

Minimize Flash and Animation

Minimize the use of JavaScript and Flash as far as possible, as a number of mobile devices and tablets do not support Flash, and this would sorely inhibit the navigability of your website. Several web browsers run outdated versions of Flash plugins, and some may not even have Flash installed, so you must keep the masses in mind. Switch to HTML5 if applicable, in case you require animations. HTML5 is a fantastic browser-compliant substitute for Flash

Remember

You must ensure that your website can be accessed by everyone visiting it no matter what application or browser they are using.

Make Your Site Accessible

If you are aspiring for an overwhelming traffic then you need to make your site compatible and attuned to multiple devices and browsers

Usability

It is important to keep in mind that the success of a site depends on its usability and not on its visual design. It is best to use a user-centric design if you have a profit-oriented website in mind

The success of a website depends on its usability because its existence is meaningless if users cannot use it. There is absolutely nothing wrong in using eye-catching 1-2-3-done steps or large buttons with attractive visual effects.

It is necessary to let the user view all the available functions clearly. Feature exposure contributes to a good user interface design. The visitors should be able to interact with the system comfortably.

Pay Attention to the Writing Style

Web content writing is much different from print. You need to write as per users' browsing habits and preferences. Use objective language. No one is interested in reading promotional content. Everyone would avoid long text blocks without keywords and images. You need to talk business and provide .crisp and concise content

Minimize User Workload

First of all it is best to minimize cognitive load so that it becomes easier for visitors to understand the idea behind your system. Also, ensure that less action is needed from users to try a service. Only then would a random visitor try it out. First time visitors are not interested in filling web forms for some account which they may never use again. It is a wise idea to design your website in such a way that visitors have the ease and freedom to browse the site without being compelled to share private data. They would keep coming back to your .site once they discover your services

Remember

You need to categorize your content, remember to use subheadings, use bulleted lists and visual elements to break the monotony of blocks of uniform.



Use Negative Space

Using negative space in your web design is an excellent idea. Nobody wants fluff. Visitors should be able to figure out precisely what they have in front of them without too much searching. Designers should not worry about negative space while creating a site. You just need to create normally and then go through it and remove stuff wherever required to .get the desired look

Conventions Are Fruitful

Incorporating conventional site elements need not necessarily make your website boring. In fact, conventions are good for reducing the learning curve or the need to understand how a particular thing works. Don't you think it would be a usability crisis if different websites had completely different RSS-feeds' visual presentation? Conventions help you to achieve users' trust, confidence and reliability. It is a good idea to keep track of users' expectations and understand precisely what expectations they have from search placement, site navigation, text structure etc

Test Early, Test Often

TETO-principle is vital for all web design projects. Usability tests are great for providing critical insights into the major issues and problems associated to a given layout. You must test early and never too late. You should go for comprehensive tests

6.1.5 Web Technologies: Importance and Carrier

A technology which acts as an interface between web servers and clients. It includes markup languages, programming interfaces, standards to define document identity and display. Basically web technology is the process which allows two more computing devices over a network. In order to face the new market environment which is in constant change, the company must place the customer in the center of its attention. As a result, the company will not follow, first of all, the benefit brought by a certain successful business, but to develop long-term business relationships with the same customers. The integration of Web technologies has an important place into the process of accomplishing companies' objectives to increase the competitiveness degree on the market by generating customers' loyalty. Developing a web-site makes it possible a very good communication with the clients, and this leads, finally, to a constant adaptation of the company's offer to the continuously changing customers' requests.



Web technology is interpreted with more scripting languages, i.e. if the web developer writes a code which is easily understandable by the web browsers. On the other-hand, compiled scripting languages needs translator to interpret the code into

machine language code. Different web technologies have different server dependencies, so before choosing the particular web technology, determine the services hosted by that technology.

List of Different Web Technologies

List of web technologies used in web development are:

- **ASP:** ASP (Active Server Pages), today intuitively offers many web application development services to the clients across the world. It is one of the popular web technology created by Microsoft to develop dynamic web pages, applications and web services. This web hosting technology creates codes using VB Script and Java Script languages that can be developed and processed on all computers through windows operating system. The .net development company's services have brought innovation in the field of web development.
- **PHP:** PHP (Hypertext Preprocessor) is a server side scripting language designed for web development to produce dynamic web pages. It is a web technology that offers choices other than ASP UNIX based servers. The potential of PHP web application is that the PHP code is embedded into the body of an HTML page document which makes it a great tool for web development. In the recent times, for most of the web application PHP development India becomes the obvious choice for various ecommerce and web portal designers.
- **JSP/JavaScript:** JSP (Java Server Pages) web technology enables web developers and designers to create informative dynamic web pages. With the intro of JSP, website owners find revolutionary changes in the development of websites. It is a prototype based scripting language that supports dynamic, weakly typed and first-class functions. JSP promotes rapid development of platform independent web applications. The main feature of JavaScript scripting language is that it can be used to create dynamic HTML codes and almost every popular browsers support this web technology.
- **Perl:** Perl is a script based programming language that offers huge number of uses to web application development technology. It includes major features like multiple programming paradigms (Procedural, object-oriented, and functional styles), reference built-in support for text processing, and a large collection of third-party modules.

Perl scripts are very simple & easy-to-use and are used by Unix/Windows administrators widely. It is similar to C programming language and used in UNIX-based servers. Using Perl, the programmers can automate tasks, send automatically generated emails, monitor logs, and transfer files, logs and data.
- **CGI:** CGI (Common Gateway Interface) similar to ASP and PHP, is one of the most widely user server applications on the internet. CGI applications can

be written in any programming languages like C, C++, Java, and Perl etc. to design & develop dynamic and interactive web pages. Some of CGI scripts you can find are web site management, password protection, shopping carts and many more. Perl is the most popular scripting language used for CGI programming.

- **VBScript:** VBScript (Visual Basic Scripting Edition) is an Active Scripting language that is easy to learn and commonly used for server side scripting.
- **Visual Basic .NET:** VB.NET (Visual Basic .NET) is an object oriented computer programming language & advanced form of visual basic programming language that hosts number of features to help web developers to create bug free custom web applications in a simple manner.

6.1.6 Careers in Web Technologies

Literally any one can! The job of a web developer is that of a programmer who can write efficient codes in various languages used for the web. “Becoming a good programmer” does not actually need any formal education. You just need to decide on which language you want to learn, buy a couple of books for that language, read tutorials on internet & then try your hands on coding. You need to write lots of code yourself, learn by developing simple & small applications initially & then move up yourself to big & challenging applications. That is the only way you can become a good programmer. However having a formal education in the following streams will definitely make your job easier graduation in mathematics, engineering in computer sciences & electronics, graduation in computer sciences. The reason is these courses teach basics of programming as part of their curriculum. But at the same time anyone can become a good programmer by learning themselves. All you need is a taste for logic & programs

Web technology is a blanket term that can be used to cover all the people who plan, build, and manage a given website-which can include everyone from product and project managers to writers, designers, information architects, programmers, and database administrators. Web professionals are charged with nothing less than conceiving, designing, building, programming, populating with content, branding, marketing, supporting, and managing websites.

Web development combines a large dose of creativity with a large background in technical knowledge. Although much of what one needs to know as a Web developer can be gathered in college classes, preparation in high school can help. Some high schools might be better equipped to handle the creative end than the technical end.

Web developers, like most computer specialists, are scattered throughout every segment of the U.S. economic scene. However, the greatest concentration is in retail business, especially market segments that rely on a strong online presence. These

businesses require a great expertise in designing pages that are easily navigable for the customer.

Also, as in other computer-based careers, the job market is growing faster than the national average and is expected to continue for the next several years. Computer skills remain among the better-paying positions, especially for recent graduates. Top Web developers, who combine creativity with technical know-how, can command salaries high above the average computer professional.

6.1.7 Job Role of Web Developer

The job of the Web development team is to create a compelling website. Such websites, generally speaking, are designed to support a business, be it selling things or enabling other kinds of transactions (Charles Schwab & Co., eBay), providing financial advice (The Motley Fool) or other online content (The Onion), or helping people search the Web (Google). Web developers conceive of the website strategy, working for or in consultation with the decision-makers at a company. They figure out the hardware that the site should use, the software necessary to make it function properly, the design and navigation that will get the public to use the site in a way that will support its business, and the information that will keep users coming back. Web developers also program the site so that it functions effectively, adding tools like community discussions and newsletter sign-up capability. They also set up reporting tools and databases to record traffic to the site and what visitors are doing there. (Buying things? Chatting with others? Clicking through on banner or popup ads? Reading?)

These days, because Web surfers are increasingly accessing the Internet via wireless devices, be they Wi-Fi- or Bluetooth-enabled computers, cell phones, or personal digital assistants, Web dev professionals are increasingly facing the need to optimize the websites they run for wireless devices.

Some Web development positions require technical skills. Creating the back end of a website (the database and hardware infrastructure on which it sits and through which orders are fulfilled, for instance) and the front end (the design and navigation and tools used by site visitors, such as stock quotes, relocation calculators, polls) requires programming skills.

Other positions may require a familiarity with technology, but not technical skills. Producing a site-coordinating between front-end and back-end developers, making sure the site supports the company's brand, working with users and designers to come up with a navigable site, ensuring that the content supports the site's business objectives-does not necessarily require high-level technical skills, though a familiarity with technology is generally a prerequisite.

Product management and content development jobs can also require an understanding of a specific industry or business niche; for instance, a potential content developer at

an investment-advice website will typically need to display an understanding of stocks and the markets to get the job. Content developers write, edit, shape, and aggregate information. Project managers work across functions to make sure projects get finished on time.

A decade ago, a person with a little computer expertise and some HTML skills might be considered a Web developer simply because few people had better backgrounds. Fortunately (or unfortunately), Web development has come a long way since then. Web development has become a profession with standards. More expertise than ever is needed to succeed.

Web developers produce sites and systems on the World Wide Web for organizations. The sites might be purely for information sites, but they are based on Web information systems that combine analysis, design, and authoring with multimedia development and traditional computing skills, particularly programming and using databases.

A Web developer, or Internet developer, is often responsible for site design, creation and day-to-day operation. Again, the job title might differ from company to company. Many companies contract with consultants or freelancers rather than train staff to create an entire site. They might use an on-site staff to keep the site operating and to update it periodically. Other companies might hire a Web developer to create the site and maintain it once it is in operation.

Work in entry-level Web development can include:

- Gathering relevant content and information from key company personnel;
- Developing the Web design concept and page organization;
- Presenting the site for approval or refinement from company personnel;
- Designing, building, and testing Web pages and links; and
- Updating contents and maintaining the site.

Developers who design sites from the initial concept must gather data to identify client needs and turn the data into functional and technical specifications for a Web site. This can include selecting programming languages and writing the code. Developers can recommend Web hardware and software, evaluate Web technologies, and develop standards for the site.

The responsibilities or required skills for a Web developer can be divided into three categories. Design skills involve content development, graphics and layout, and security. Technical skills are needed to write code, develop databases, and implement testing. Management skills are used to evaluate the hardware and software systems, train other designers and developers, coordinate with other company personnel, and provide customer service.

Web developers have to evaluate the needs of the individual or company for whom they are creating a site. They assist in planning an Internet strategy that allows those

goals to be achieved. A Web developer first must talk with Web site owners and users to understand what the Web site owners want their Web site to look like and to do.

Web sites can consist of intranets, which are confined to internal employees, and extranets, which include groups of external users. Developers must design a site that allows specific users to have appropriate access. They have to develop, assess, and communicate security policies and standards among these constituencies.

Once the needs and desires have been assessed, the developer might create a storyboard to help the Web site owner understand what the site will look like. These **storyboards** are collections of sketches, much like the tools that movie and television directors use to plot one scene's transition to the next. The Web storyboard simulates transitions between pages showing how the user will move through the site and to make sure the pages flow logically. If the storyboard is approved, then the building of the site can begin.

The creative side is followed by the technical side. Web developers establish the tools and software required to produce and operate the Web environment. A variety of software is needed to construct the site. A simple site might incorporate easy-to-use software such as FrontPage or Dreamweaver. More complicated sites are likely to be built from scratch using HTML in conjunction with other computer programming languages, such as Java or Visual Basic.

The variety of technology programs and applications seems endless, but the Web developer needs to remain up-to-date on the latest versions of common publishing tools. They must understand which applications work best with a particular system, or which is more likely to produce the desired expectations of the Web site owner. They test applications and sites to make sure that users can navigate easily while getting their transactions completed securely. Good Web developers encourage feedback to make adjustments and improvements as needed.

Management skills are a third necessity for the Web developer. Although he or she might spend considerable time in front of a computer, the Web developer must interact with others in the company. This can include training people within that company to work with the system, or even selling

Keyword

Storyboard is a graphic organizer in the form of illustrations or images displayed in sequence for the purpose of pre-visualizing a motion picture, animation, motion graphic or interactive media sequence.

company officials on the idea once it has been created. It can involve market research and consultations with management and various experts.

Sometimes the job requirements might extend to customer service. In these circumstances, developers must take feedback from Web site users, provide information, and update the site as needed. It is helpful to have some understanding of e-commerce to be better prepared for the business goals of a Web site and to understand the importance of solid customer relations. Entry-level Web developers might be called upon to perform most or all the above duties. However, professionals that work for larger companies are more likely to have additional personnel who either assist with the duties or are assigned specific duties outright. In any case, each of these skill sets will be useful in succeeding as a Web developer.

6.1.8 How the Website Works?

Before trying to make your own web page and launch it on the Internet, first you need to know how web pages work. Here are the basic terms:

- The server receives the request for a page sent by your browser.
- The browser connects to the server through an IP Address; the IP address is obtained by translating the domain name.
- In return, the server sends back the requested page.

Web pages are written in HTML, Hypertext Markup Language. A markup language is a computer language that describes the layout, format and content of a page. The Web browser renders the page according to the HTML code.

Web servers are computers whose job is to respond to a browser's request for a web page and deliver it through the internet. Pages hosted on a web server can be displayed to anyone all over the world. It is like a hard drive that stores your website files and images. In order to host your page on a web server you need to pay a hosting charge.

In order to display your website on the internet, you need a web hosting provider. You could set up your own hosting server at home, but it would take a huge amount of knowledge and require a lot of time to set up. Paying a hosting service offers you a lot more freedom for your site, gives you opportunity to do professional work and is a reliable solution. To create your own website, you first need a domain name.

You can get the domain name from your web hosting provider or from a separate provider specialized in domains names. Sometimes web hosting providers offer you a domain name along with your hosting account. It may appear to be the easiest option but could be more costly. There are a lot of web hosting providers available in a wide range of prices depending on the services such as number of domains, hosting space or bandwidth provided. Choose carefully, or you could face slow servers, periods of malfunction or no support.

When you enter something like Google.com the request goes to one of many special computers on the Internet known as **Domain Name Servers** (DNS). All these requests are routed through various routers and switches. The domain name servers keep tables of machine names and their IP addresses, so when you type in Google.com it gets translated into a number, which identifies the computers that serve the Google Website to you.

When you want to view any page on the Web, you must initiate the activity by requesting a page using your browser. The browser asks a domain name server to translate the domain name you requested into an IP address. The browser then sends a request to that server for the page you want, using a standard called Hypertext Transfer Protocol or HTTP.

The server should constantly be connected to the Internet, ready to serve pages to visitors. When it receives a request, it looks for the requested document and returns it to the Web browser. When a request is made, the server usually logs the client's IP address, the document requested, and the date and time it was requested. This information varies server to server.

An average Web page actually requires the Web browser to request more than one file from the Web server and not just the HTML / XHTML page, but also any images, style sheets, and other resources used in the web page. Each of these files including the main page needs a URL to identify each item. Then each item is sent by the Web server to the Web browser and Web browser collects all this information and displays them in the form of Web page.

Keyword

Domain Name Server is a computer hardware or software server that implements a network service for providing responses to queries against a directory service.

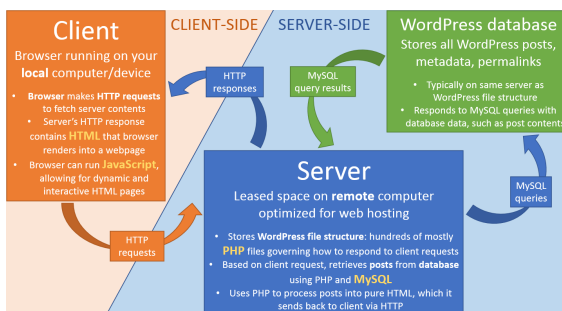
6.2 CLIENT AND SERVER SCRIPTING LANGUAGES

Client-side scripting is a process by which a web-based computer program runs on the user's computer rather than on the computer hosting the website. Specifically, it runs through a type of program known as a script, which is handled by the user's web browser. The main reason for client-side scripting is to allow a web page to be created specifically around the user's own data and options, rather than being a fixed page that always appears the same to every user.

This made websites very limited in dealing with large sets of data, such as a railway schedule. With static pages, the only solution was to print the schedule in full and let the user hunt down the relevant detail.

This problem was solved by the development of dynamic web pages. These can adapt to meet a specific situation, such as if a user is searching for a journey between two stations within a specific time period. The result of the search is displayed through a dynamic web page, which is automatically created for that query.

Server-side scripting is a term primarily used with regards to serving custom content via the hypertext transfer protocol (HTTP) on a web server by having the server execute small programs. These programs, usually written in a scripting language, are executed by the server when a client request arrives. Depending on the parameters given by the client at the time of the request, the script then generates a web page for the client. Web pages created in this way are often referred to as dynamic pages. This is in contrast to a client-side script that is sent from the server to the requesting client and then executed.



Did You Know?

In 1989, whilst working at CERN Tim Berners-Lee proposed to create a global hypertext project, which later became known as the World Wide Web. During 1991 to 1993 the World Wide Web was born. Text-only pages could be viewed using a simple line-mode browser.

The server-side scripts were often times small, executable files containing a series of commands to pass to the operating system. The web daemon, the software running the web server, would use these shell scripts to further execute an additional program residing on the host computer. This general technique was then defined in 1993 in the common gateway interface (CGI) standard developed by the Internet Engineering Task Force (IETF). In most cases, these early CGI scripts were used to send database responses from the server back to the requesting client. As interpreted scripting languages were developed,

such as Perl, and PHP: Hypertext Preprocessor (PHP), server-side scripting methods also evolved. HTTP daemons were updated to include use extensions that allowed for these various scripting languages to be called from the web daemon itself, instead of being passed to the operating system on the host. With this addition, bits of scripting language could be included within the HTML document. As the web server reads through a document before sending it out across the web, it checks for and executes any of the script within the document. Since the content called up by server-side scripting resides in a database, it can be virtually anything. Content such as product descriptions, price variations, weblog entries, images, and even formatting is stored in the database. It is also possible to nest one server-side script snippet within another, where the first script grabs certain data from the database, but also calls a second script which accesses secondary data.



When delivering some fairly custom standard content with additional nested content that may be time sensitive or as yet unknown, such as comments on a weblog entry. Web site maintenance, then, involves updating data in the database, which .will then affect every page on the website with the included script to call that data

Keyword

Common Gateway Interface (CGI)

is a standard way for web servers to interface with executable programs installed on a server that generate web pages dynamically.

Many of the server-side scripting techniques have been further developed into what are known as content management systems (CMS). PHP is probably the most frequently used language for this purpose, running at the core of many CMS implementations. Here, a user operating a website running on a CMS will edit her HTML documents to include what are called tags. The tags are essentially cues to the CMS to tell it what script to include in the document. The CMS can then be customized by creating additional scripts and their .associated tags

6.2.1 Domains and Hosting

The domain name is much like an entry in a phone book. Computers communicate by using numbers, called IP addresses, to contact each other, much like you use a phone number to dial a specific person's phone. If you want people to find your business's phone number, you want to be listed in a phone book. The phone book tells people looking for your

phone number “Company A’s Phone number is xxx-xxx-xxxx” just as a domain tells people (i.e. their computers) “domainA.com is hosted on the server xxx.xxx.xxx.xxx” Without the domain, you would have to tell your customers “Hey my site is located at 123.456.789.123/~mysite/” instead of “mysite.com” You can see how, without a domain, having a site or hosting is impractical.

The web-hosting or server portion is much like the space that you rent out to have your business in. It is merely the space itself. It does not include furnishings like shelves for your products, just as the web-hosting account does not include a site for you to sell your products. Luckily, in the web-hosting world, it is very easy to furnish the space provided by your host, because you can install many framework applications through the Fantastico icon within your cPanel. Without the hosting services, you would not have a place for your files to reside, so your domain would then become like a disconnected phone number in the phone directory, and your site files would have nowhere to stay.

Domain Name and Web Hosting Explained

To simplify: A domain name, is like the address of your home; web hosting on the other hand, is the space of your house where you place your furniture. Instead of street name and area code, set of words or/and numbers are used for the website’s naming’. The same goes with hosting, computer hard disk and computer memory are used instead of wood and steel for storing and processing data files. The idea is presented clearer with the diagram below.

When you register a domain, it gives you sole ownership and rights to the name of your site. No one else in the market has the access to the actual name of that particular domain besides you. A domain registration is a “right to use and control” a domain name. It is not related to any content or services associated with that domain. Domain registrations are quite simple. They are valid for a period of time, they generally have some contact information associated with them, and may have some DNS servers too. (DNS servers are often called name servers.

A web hosting normally refers to the web server (big computer) that stores lots of data files. A web hosting providers normally rent out web servers and network connection to the end-users or the resellers. For most cases, the hosting providers will be the parties handling most server maintenance work such as backup, root configuration, maintenance, disaster recoveries, etc.; but for certain cases, the end users will need to get everything cover by themselves. Web-Hosting is a set of services. We configure our web-servers to hand out web pages and handle e-mail for your domain name. That sounds a lot simpler than it is. Web pages can be complex. They are often generated by programs which store and retrieve information from databases. The world of e-mail is not simple either. It can be as simple as forwarding messages to a mailbox at an existing ISP, or it can include multiple layers of filtering (virus,

content analysis for spam detection, challenge/response) before being stored for later retrieval. The message might be retrieved by our Webmail system, or an e-mail client on a home or business computer like Outlook or Eudora.

6.2.2 Websites Creation

Technology has really advanced over the past few years. The internet has grown faster than ever before. Facebook has taken over, Google is king, and normal folks like yourself can build your own website for free. More and more people are creating websites online as a way to express themselves and get in touch with others. Businesses cannot afford not to have websites as a way of reaching more potential customers.

Building a website is a great way to share your ideas and thoughts with the world. But if you have never done one, it can seem daunting. There's all that http-dot-whatever and `<tag this="">` and `<tag that="">` and how do you get pictures and text in there? Well fear not, this will help you to grasp the intricacies very quickly!

Designing Your Website

1. Get inspired. Look at websites with great designs and think about why they are great designs. It usually comes down to the information, resources, links, and pages being laid out in a way that is easy to see and use. To get ideas about how to design your own site, look at sites which do similar things to get ideas about where you should put different types of content.
 - Stay realistic to your skills.
 - Ease of access is the most important thing. If you do not have a certain piece of information easily visible, make sure that getting to that information is very logical.
 - Generally the simpler the design, the fewer the pages, the better.
2. Choose a topic and purpose. If you already have a fairly good idea about what your website will focus on, skip this step. If not, here are some things to help you figure that out. First, understand that there are billions of people on the Internet, and a large percentage have websites. If you limit yourself to something that has not been done, you will never get started.
 - When you think, "Internet," what is the first thing that comes to your mind? E-commerce? Music? News? Socializing? Blogging? Those are all good places to start.
 - You could create a website that's dedicated to your favorite band, and have a chat area where people can talk about it.
 - You can build a page for your family, but be careful about things like this. The internet is full of unsavory characters and information you put up about your family could end up being used against you. Consider adding password protection to your personal family website.

- If you are a news junkie, or want something less filtered than traditional media, build a website and get publicly available feeds from news providers such as Reuters, BBC, AP, and others. Build your own customized news aggregator (what used to go by the quaint name of “newspaper”), then see and show all the news that’s fit to digitize.
- If you are creative at writing then you can start a blog where you can write about anything you want and attract monthly readers!
- 3. Make a plan. Building your website is going to take a commitment of time and possibly money, so set a limit on both, and then dig in. The plan does not have to be a big, complicated spreadsheet, or a fancy graphic presentation, but at the very least, you will want to consider what it will do for you and the visitors, what you will put on the website, what goes where on the webpages.
- 4. Gather the content. There are lots of different types of content and many have their own considerations. You will need to figure out what’s best for your website and your needs. Some things to consider including:
 - A store. If you want to sell things, you will need to figure out how you want the items to be available. If you have relatively few things to sell, you might want to consider having a store with a hosting service. Society, Amazon, and Cafepress are all well-established store hosts which let you sell a variety of items and set your own prices.
 - Media. Do you want to display videos? Music? Do you want to host your own files or do you want them hosted somewhere else? YouTube and SoundCloud are great examples of hosting options, but you will need to be sure that the way you design your website allows these media types to display correctly.
 - Images. Are you a photographer? An artist? If you plan on putting original images on your website, you might want to use a format that can help keep them from getting stolen. Make sure the images are relatively small or that they are hidden behind some Flash code, so as to keep them from being easily saveable.
 - Widgets. These are mini-programs which run on your website, usually to help you keep track of who visits, what they are looking for, and where they are from. You can also find widgets for booking appointments, displaying a calendar, etc. Look into what might be useful for you (just make sure the widget comes from a reputable source).
 - Contact information. Do you want to have contact information on your webpage? For your own safety, you should be careful about what kind of information you have available. You should never display things like your home address or home phone number, as information like this can be used to steal your identity. You may want to set up a PO Box or a special email address for people to contact you at, if you do not have a business address.

5. Draw a flow chart. For most people, the website starts on the home page. This is the page that everybody sees when they first go to `www.yourSite.com`. But where do they go from there? If you spend some time thinking about how people might interact with your site, you will have a much easier time down the line when you are making navigation buttons and links.
6. Plan for user devices and situations. In recent years, smartphones and tablets have become incredibly popular platforms for browsing the internet, and they require websites to be designed for them. If you really want to make a website that will stand the test of time and be accessible to the highest number of viewers, plan on making different versions of your site for different devices, or plan to use a responsive design that adjusts as necessary.

Building Your Website

1. Decide what method or tool you will use to build it. When you have the basic idea down and have a plan for how it will be laid out, the next to think about is how you are going to build it. The options seem endless, and people will try to sell you this or that ‘fantastic’ application, and every other thing that you “absolutely must have” on your site, however the reality is that there are a few great tools for building websites, and one of them will be best-suited to your situation and needs.
2. Build it yourself. This is the first option. If you have a website-building application like Adobe Dreamweaver, it is not very difficult to create a website from scratch. You might need to do some coding but do not panic! HTML looks complicated, but it is like listening to Shakespeare—it is hard at first, but once you get the feel of it, it is not that difficult.
 - Pros: website design software simplifies the process of building sites by letting you drag-and-drop images, text, buttons, movies, and anything else you can think of, usually without ever having to dig into HTML. Many web design applications will even let you create sites specifically for your smart phone or pad. If you are building a basic, personal website, this is really a great way to go.
 - Cons: there is a learning curve, and though you do not have to dig into HTML, it is not totally geek-free. If you are in a hurry, this might not be the best solution. Perhaps the biggest con, though, is that if you are not a graphic designer, you could end up with a page that hurts the eyes. To mollify this somewhat, there are a number of free templates in the applications, and on the internet, but be aware of your limitations—if you have any!
3. Use a **content management system** (CMS). This is the second option. WordPress is an example of a great option for building websites. It helps you create web pages and blog posts quickly and easily, set up the menus, allow and

manage user comments, and has thousands of themes and plugins that you can choose from and use for free. Drupal and Joomla are other great CMS options. Once the CMS is hosted, you can manage your site from anywhere (in the world) that has an Internet connection.

- Pros: Very easy to use, quick to get started with one click install, and lots of options for the beginner (with enough depth for more experienced users).
- Cons: Some themes are limiting, and not all are free.

4. Build the website from scratch. This is the third option. If you decide to build your website from scratch, you will need to start using HTML and CSS. There are ways to extend your HTML skills and add more features and more depth to your website. If you are developing a professional website, these tools will help you get that edge that is needed in any business venture.

- CSS, which stands for “Cascading Style Sheets”. CSS gives more flexibility for styling the HTML, and makes it much easier to make basic changes—fonts, headers, color schemes—in one place, and have those changes ripple through the site.
 - XHTML is a web language set by W3C’s standards. Almost identical to HTML, it follows a stricter set of rules for marking up information. What this means, for the most part, is minor changes to the way you write code.
 - Look into HTML5. It is the fifth revision of the core HTML standard, and will eventually subsume the current version of HTML (HTML4), and XHTML as well.
 - Learn a client-side scripting language, such as JavaScript. This will increase your ability to add interactive elements to your site, such as charts, maps, etc.
 - Learn a server-side scripting language. PHP, ASP with JavaScript or VB Script or Python can be used to change the way web pages appear to different people, and lets you edit or create forums. They can also help store information about people who visit your site, like their username, settings, and even temporary “shopping carts” for commercial sites.
 - AJAX (Asynchronous JavaScript and XML) is a technique of using a browser sided language and a server sided language to make the page get new information from the server without refreshing the page, often greatly reducing user wait time and vastly improving the user’s experience but increasing bandwidth usage. For a website that will see a lot of traffic, or an e-Commerce site, this is an excellent solution.
5. Hire a professional. This is the fourth and final option. If you are not up to designing your own website, or learning new coding languages—especially for more advanced sites—hiring a professional may be your best option. Before you hire, ask to see a portfolio of their work, and check their references carefully.

6.2.3 Types of Websites

A Web site is a related collection of World Wide Web (WWW) files that includes a beginning file called a home page. A company or an individual tells you how to get to their Web site by giving you the address of their home page. From the home page, you can get to all the other pages on their site. *For example*, the Web site for IBM has the home page address of <http://www.ibm.com>. The home page address actually includes a specific file name like `index.html` but, as in IBM's case, when a standard default name is set up, users do not have to enter the file name. IBM's home page address leads to thousands of pages.

Since site implies a geographic place, a Web site can be confused with a Web server. A server is a computer that holds the files for one or more sites. A very large Web site may be spread over a number of servers in different geographic locations. IBM is a good example; its Web site consists of thousands of files spread out over many servers in world-wide locations. But a more typical example is probably the site you are looking at, whatis.com. We reside on a commercial space provider's server with a number of other sites that have nothing to do with Internet glossaries.

A synonym and less frequently used term for Web site is "Web presence." That term seems to better express the idea that a site is not tied to specific geographic location, but is "somewhere in cyberspace." However, "Web site" seems to be used much more frequently.



There are many static websites on the Internet, you may not be able to tell immediately if it is a static or dynamic website, but the chances are, if the site looks basic and is for a smaller company, and simply delivers information without any bells and whistles, it could be a static website.

Static Websites

There are many static websites on the Internet, you may not be able to tell immediately if it is a static or dynamic website, but the chances are, if the site looks basic and is for a smaller company, and simply delivers information without any bells and whistles, it could be a static website.

Advantages of static websites

- Quick to develop
- Cheap to develop
- Cheap to host

Disadvantages of static websites

- Requires web development expertise to update site
- Site not as useful for the user
- Content can get stagnant

Dynamic Websites

Dynamic sites on the other hand can be more expensive to develop initially, but the advantages are numerous. At a basic level, a dynamic website can give the website owner the ability to simply update and add new content to the site.

News and events could be posted to the site through a simple browser interface. Dynamic features of a site are only limited by .imagination



Some examples of dynamic website features could be: content management system, e-commerce system, bulletin / discussion boards, intranet or extranet facilities, ability for clients or users to upload documents, ability for administrators or users to create content or add information to a site (dynamic publishing). Advantages of dynamic websites

- Much more functional website
- Much easier to update
- New content brings people back to the site and helps in the search engines

- Can work as a system to allow staff or users to collaborate

Disadvantages of dynamic websites

- Slower / more expensive to develop
- Hosting costs a little more

6.2.4 Web Standards

Web standards are rules and guidelines established by the World Wide Web Consortium (W3C) developed to promote consistency in the design code which makes up a web page. Without getting technical, simply it is the guideline for the mark-up language which determines how a web page displays in a visitor's browser window.

The advantages in adhering to these standards are many:

- Web pages will display in a wide variety of browsers and computers, including new technology like iPhones, Droids, iPads, PDA devices, mobile phones, which greatly increases the viewing audience.
- W3C Standards promote the use of "Cascading Style Sheets" (CSS) or design code which is attached to the web page rather than embedded in the page. The use of style sheets significantly reduces the page file size which means not only a faster page loading time but lower hosting costs for frequently visited sites due to using less bandwidth.
- Design features such as colors and fonts can be easily changed by just modifying one style sheet instead of editing every individual page in a web site, reducing the costs to modify your site.
- Search Engines are able to access and index pages designed to web standards with greater efficiency.

W3C Standard

W3C standards define an open web platform for application development that has the unprecedented potential to enable developers to build rich interactive experiences, powered by vast data stores that are available on any device. Although the boundaries of the platform continue to evolve, industry leaders speak nearly in unison about how HTML5 will be the cornerstone for this platform. But the full strength of the platform relies on many more technologies that W3C and its partners are creating, including CSS, SVG, WOFF, the Semantic Web stack, XML, and a variety of APIs.

W3C develops these technical specifications and guidelines through a process designed to maximize consensus about the content of a technical report, to ensure high technical and editorial quality, and to earn endorsement by W3C and the broader community.

If you are learning about Web technology, you may wish to start with the introduction below, and follow links for greater detail.

Web Design and Applications Header link

Web Design and Applications involve the standards for building and Rendering Web pages, including HTML, CSS, SVG, Ajax, and other technologies for Web Applications (“Web-Apps”). This section also includes information on how to make pages accessible to people with disabilities (WCAG), to internationalize them, and make them work on mobile devices.

Web of Devices Header link

W3C is focusing on technologies to enable Web access anywhere, anytime, using any device. This includes Web access from mobile phones and other mobile devices as well as use of Web technology in consumer electronics, printers, interactive television, and even automobiles.

Web Architecture Header link

Web Architecture focuses on the foundation technologies and principles which sustain the Web, including URIs and HTTP.

Semantic Web Header link

In addition to the classic “Web of documents” W3C is helping to build a technology stack to support a “Web of data,” the sort of data you find in databases. The ultimate goal of the Web of data is to enable computers to do more useful work and to develop systems that can support trusted interactions over the network. The term “Semantic Web” refers to W3C’s vision of the Web of linked data. Semantic Web technologies enable people to create data stores on the Web, build vocabularies, and write rules for handling data. Linked data are empowered by technologies such as RDF, SPARQL, OWL, and SKOS.

XML Technology Header link

XML Technologies including XML, XML Namespaces, XML Schema, XSLT, Efficient XML Interchange (EXI), and other related standards.

Web of Services Header link

Web of Services refers to message-based design frequently found on the Web and in enterprise software. The Web of Services is based on technologies such as HTTP, XML, SOAP, WSDL, SPARQL, and others.

Browsers and Authoring Tools Header link

The web's usefulness and growth depends on its universality. We should be able to publish regardless of the software we use, the computer we have, the language we speak, whether we are wired or wireless, regardless of our sensory or interaction modes. We should be able to access the web from any kind of hardware that can connect to the internet stationary or mobile, small or large. W3C facilitates this listening and blending via international web standards. These standards ensure that all the crazy brilliance continues to improve a web that is open to us all.

SUMMARY

- The style tile is a design deliverable that references website interface elements through font, color, and style collections delivered alongside a site map, wireframes, and other user experience artifacts.
- Web technologies are infrastructural building blocks of any effective computer network: Local Area Network (LAN), Metropolitan Area Network (MAN) or a Wide Area Network (WAN), such as the Internet.
- Website layouts are mostly based around content. A style tile is not great for explaining how content fits together but it's perfect for explaining everything else.
- The branding should be recognizable and explain that the document is only a style guide.
- Style tiles offer a flexible starting point to help define what a website will look like, in an accessible way that clients can understand.
- The web world is said to have undergone some technological changes and responsive web design is at the forefront as far as content marketers are concerned.
- Designers are now able to develop mobile-optimized images that are ideal for smaller screens and they can then come up with higher-resolution versions for larger screens.
- Server-side scripting is a term primarily used with regards to serving custom content via the hypertext transfer protocol (HTTP) on a web server by having the server execute small programs.

KNOWLEDGE CHECK

1. **Common gateway interface is used to**
 - a. Generate executable files from web content by web server
 - b. Generate web pages
 - c. Stream videos
 - d. None of the mentioned
2. **Dynamic web page**
 - a. Is same every time whenever it displays
 - b. Generates on demand by a program or a request from browser
 - c. Both (a) and (b)
 - d. None of the mentioned
3. **What is a web browser?**
 - a. A program that can display a web page
 - b. A program used to view html documents
 - c. It enables user to access the resources of internet
 - d. All of the mentioned
4. **URL stands for**
 - a. Unique reference label
 - b. Uniform reference label
 - c. Uniform resource locator
 - d. Unique resource locator
5. **Style tiles are the wonderful visual concept created by Samantha Warren that help form a common visual language between the designers and the stakeholders.**
 - a. True
 - b. False
6. **To make your website mobile friendly, you can make your website**
 - a. Responsive
 - b. Reactive
 - c. Fast Loading
 - d. Light
7. **Which program is used by web clients to view the web pages?**
 - a. Web browser
 - b. Protocol

Web server

Search Engine

8. **How many colour names are used by the browsers?**
 - a. 8
 - b. 10
 - c. 12
 - d. 16
9. **Which of the following softwares could be used to build a website**
 - a. Power Point
 - b. Excel
 - c. Dream Weaver
 - d. ERP
10. **Which of the following statements is true**
 - a. The web designer shouldn't just be concerned about the looks but also about user interface
 - b. Usability is very important in web design
 - c. a and b
 - d. None of the above

REVIEW QUESTIONS

1. How to use style tiles?
2. Focus on web design techniques.
3. What are the job roles of web developer?
4. Discuss how the website works.
5. Differentiate between static and dynamic website.

Check Your Result

- | | | | | |
|--------|--------|--------|--------|---------|
| 1. (a) | 2. (b) | 3. (d) | 5. (c) | 5. (a) |
| 6. (a) | 7. (a) | 8. (d) | 9. (c) | 10. (c) |

REFERENCES

1. Chapman, Cameron, The Evolution of Web Design, Six Revisions, archived from the original on 30 October 2013
2. Design and Prototyping for Drupal: Drupal for Designers - Page 25books.google.co.in › books, Dani Nordin · 2011
3. Niederst, Jennifer (2006). Web Design In a Nutshell. United States of America: O'Reilly Media. pp. 12–14. ISBN 0-596-00987-9.
4. Redesign the Web: Smashing Book #3 - Page 323books.google.co.in › books, Smashing Magazine · 2010
5. The Principles of Beautiful Web Designbooks.google.co.in › books, Jason Beaird · 2010
6. W3C QA. "My Web site is standard! And yours?". Retrieved 2012-03-21.
7. Web Style Guide: Foundations of User Experience Design - Page 282books.google.co.in › books Patrick J. Lynch, ©Sarah Horton · 2016
8. World Wide Web Consortium: Understanding Web Content Accessibility Guidelines 2.2.2: Pause, Stop, Hide

CHAPTER 7

BOOTSTRAP

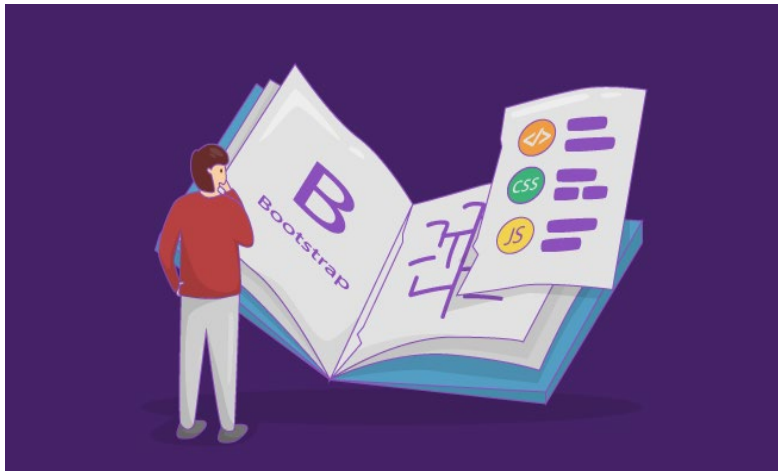
“The Kadence Importer allows you to easily import all including images, from any of our Kadence themes demos. When you install a Kadence theme, the importer will automatically see what theme you are using and give you options to import anyone of those themes”

– BBBootstrap Team

LEARNING OBJECTIVES

After studying this chapter,
you will be able to:

1. What is bootstrap?
2. Explain about responsive design for bootstrap
3. Define bootstrap CSS
4. Elaborate blockquotes
5. Explain forms
6. Discuss about buttons



INTRODUCTION

Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for

developing responsive, mobile-first websites. It solves many problems which we had once, one of which is the cross-browser compatibility issue. Nowadays, the websites are perfect for all the browsers (IE, Firefox, and Chrome) and for all sizes of screens (Desktop, Tablets, Phablets, and Phones). All thanks to Bootstrap developers -Mark Otto and Jacob Thornton of Twitter, though it was later declared to be an open-source project.



As the web evolves more and more toward responsive design, it can be a real challenge for web developers to keep up. Bootstrap can make things a whole lot easier. Bootstrap enables you to create responsive websites without you needing to do the “responsive” bit. Bootstrap takes care of that.

Bootstrap by Twitter is one such framework that was first developed by Twitter and is so versatile that it can be used to develop anything from web apps to themes on WordPress.

Almost every web developer today is familiar with Bootstrap – the world’s most popular framework to develop highly responsive mobile-first sites. A free framework, Bootstrap allows web developers the freedom to customize and create highly responsive designs real fast and with minimal efforts.

The framework offers users a collection of templates for every element of a web page – forms, buttons, typography, navigation, tables, modals, image carousels and more. It also offers JavaScript and jQuery plug-ins for additional functionality on the web-pages and supports all major browsers and their versions. Bootstrap offers developers many shortcuts to develop web pages in the least time and with minimum efforts.

7.1 WHAT IS BOOTSTRAP?

Bootstrap is an open source product from Mark Otto and Jacob Thornton who, when it was initially released, were both employees at Twitter. There was a need to standardize the frontend toolsets of engineers across the company. In the launch blog post, Mark Otto introduced the project like this:

In the earlier days of Twitter, engineers used almost any library they were familiar with to meet front-end requirements. Inconsistencies among the individual applications made it difficult to scale and maintain them. Bootstrap began as an answer to these challenges and quickly accelerated during Twitter’s first Hackweek. By the end of Hackweek, we had reached a stable version that engineers could use across the company. Mark Otto

Since Bootstrap launched in August 2011, it has taken off in popularity. It has evolved from being an entirely CSS-driven project to include a host of JavaScript plugins and icons that go hand in hand with forms and buttons. At its base, it allows for responsive web design and features a robust 12-column, 940px-wide grid. One of the highlights is the build tool on Bootstrap's website, where you can customize the build to suit your needs, choosing which CSS and JavaScript features you want to include on your site. All of this allows frontend **web development** to be catapulted forward, building on a stable foundation of forward-looking design and development. Getting started with Bootstrap is as simple as dropping some CSS and JavaScript into the root of your site.

Keyword

Web development is the process of building websites and applications for the internet, or for a private network known as an intranet.

7.1.1 Bootstrap File Structure

The Bootstrap download includes three folders: css, js, and img. For simplicity, add these to the root of your project. Minified versions of the CSS and JavaScript are also included.

It is not necessary to include both the uncompressed and the minified versions. For the sake of brevity, I use the uncompressed version during development and then switch to the compressed version in production.

```
bootstrap/
├── css/
│   ├── bootstrap.css
│   └── bootstrap.min.css
├── js/
│   ├── bootstrap.js
│   └── bootstrap.min.js
├── img/
│   ├── glyphs-halflings.png
│   └── glyphs-halflings-white.png
└── README.md
```

7.1.2 Basic HTML Template

Normally, a web project looks something like this:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Bootstrap 101 Template</title>
  </head>
  <body>
    <h1>Hello, world!</h1>
  </body>
</html>
```

With Bootstrap, we include the link to the CSS stylesheet and the JavaScript:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Bootstrap 101 Template</title>
    <link href="css/bootstrap.min.css" rel="stylesheet">
  </head>
  <body>
    <h1>Hello, world!</h1>
    <script src="js/bootstrap.min.js"></script>
  </body>
</html>
```

7.1.3 Global Styles

With Bootstrap, a number of items come prebuilt. Instead of using the old reset block that was part of the Bootstrap 1.0 tree, Bootstrap 2.0 uses Normalize.css, a project from Nicolas Gallagher that is part of the HTML5 Boilerplate. This is included in the bootstrap.css file.

In particular, the following default styles give special treatment to typography and links:

- margin has been removed from the body, and content will snug up to the edges of the browser window.
- background-color: white; is applied to the body.
- Bootstrap is using the @baseFontFamily, @baseFontSize, and @baseLineHeight attributes as our typographic base. This allows the height of headings and other content around the site to maintain a similar line height.
- Bootstrap sets the global link color via @linkColor and applies link underlines only on :hover.

Remember

If you don't like the colors or want to change a default, this can be done by changing the globals in any of the .less files. To do this, update the scaffolding.less file or overwrite colors in your own style-sheet.

7.1.4 Default Grid System

The default Bootstrap grid (see Figure 1) system utilizes 12 columns, making for a 940px-wide container without responsive features enabled. With the responsive CSS file added, the grid adapts to be 724px or 1170px wide, depending on your viewport. Below 767px viewports, such as the ones on tablets and smaller devices, the columns become fluid and stack vertically. At the default width, each column is 60 pixels wide and offset 20 pixels to the left. An example of the 12 possible columns is in Figure 1.

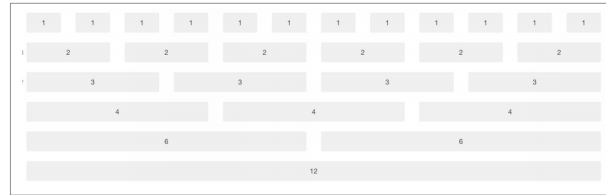


Figure 1. Default grid.

7.1.5 Basic Grid HTML

To create a simple layout, create a container with a that has a class of `.row` and add the appropriate amount of `.span*` columns. Since we have a 12-column grid, we just need the amount of `.span*` columns to equal 12. We could use a 3-6-3 layout, 4-8, 3-5-4, 2-8-2... we could go on and on, but I think you get the gist.

The following code shows `.span8` and `.span4`, which adds up to 12:

```
<div class="row">
  <div class="span8">...</div>
  <div class="span4">...</div>
</div>
```

7.1.6 Offsetting Columns

You can move columns to the right using the `.offset*` class. Each class moves the span over that width. So an `.offset2` would move a `.span7` over two columns (see Figure 2):

```
<div class="row">
  <div class="span2">...</div>
  <div class="span7 offset2">...</div>
</div>
```



Figure 2. Offset grid.

7.1.7 Nesting Columns

To nest your content with the default grid, inside of a `.span*`, simply add a new `.row` with enough `.span*` that it equals the number of spans of the parent container (see Figure 3):

```
<div class="row">
  <div class="span9">
    Level 1 of column
    <div class="row">
      <div class="span6">Level 2</div>
      <div class="span3">Level 2</div>
    </div>
  </div>
</div>
```

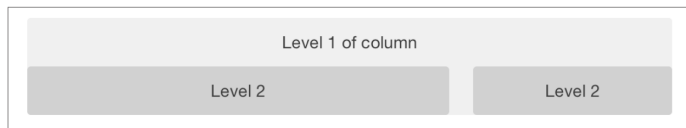


Figure 3. Nesting grid.

7.1.8 Fluid Grid System

The **fluid grid system** uses percentages instead of pixels for column widths. It has the same responsive capabilities as our fixed grid system, ensuring proper proportions for key screen resolutions and devices. You can make any row “fluid” by changing `.row` to `.row-fluid`. The column classes stay exactly the same, making it easy to flip between fixed and fluid grids. To offset, you operate in the same way as the fixed grid system— add `.offset*` to any column to shift by your desired number of columns:

```
<div class="row-fluid">
  <div class="span4">...</div>
  <div class="span8">...</div>
</div>

<div class="row-fluid">
  <div class="span4">...</div>
  <div class="span4 offset2">...</div>
</div>
```

Nesting a fluid grid is a little different. Since we are using percentages, each `.row` resets the column count to 12. This is the case for responsive content, as we want the content to fill 100% of the container:

Keyword

Fluid grid layout

provides a visual way to create different layouts corresponding to devices on which the website is displayed.

If you were inside a `.span8`, instead of two `.span4` elements to divide the content in half, you would use two `.span6` divs.



```
<div class="row-fluid">
  <div class="span8">
    <div class="row">
      <div class="span6">...</div>
      <div class="span6">...</div>
    </div>
  </div>
</div>
```

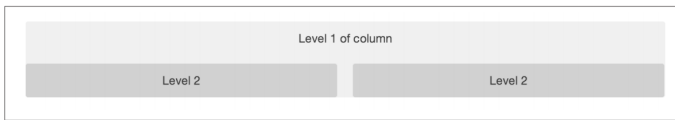


Figure 4. Nesting fluid grid.

7.1.9 Container Layouts

To add a fixed-width, centered layout to your page, simply wrap the content in `<div class="container">...</div>`. If you would like to use a fluid layout but want to wrap everything in a container, use the following: `<div class="container-fluid">....</div>` Using a fluid layout is great when you are building applications, administration screens, and other related projects.

7.1.10 Responsive Design

To turn on the responsive features of Bootstrap, you need to add a `<meta>` tag to the `<head>` of your web page. If you haven't downloaded the compiled source, you will also need to add the responsive CSS file. An example of required files looks like this:

```
<!DOCTYPE html>
<html>
  <head>
    <title>My amazing Bootstrap site!</title>
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <link href="/css/bootstrap.css" rel="stylesheet">
    <link href="/css/bootstrap-responsive.css" rel="stylesheet">
  </head>
```

7.2 RESPONSIVE DESIGN FOR BOOTSTRAP

Responsive design is a method for taking all of the existing content that is on the page and optimizing it for the device that is viewing it. For example, the desktop not only gets the normal version of the website, but it might also get a widescreen layout, optimized for the larger displays that many people have attached to their computers. Tablets get an optimized layout, taking advantage of their portrait or landscape layouts. And then with phones, you can target their much narrower width. To target these different widths, Bootstrap uses CSS media queries to measure the width of the browser viewport and then, using conditionals, changes which parts of the stylesheets are loaded. Using the width of the browser viewport, Bootstrap can then optimize the content using a combination of ratios or widths, but it mostly relies on min-width and max-width properties.

At the core, Bootstrap supports five different layouts, each relying on CSS media queries. The largest layout has columns that are 70 pixels wide, contrasting with the 60 pixels of the normal layout. The tablet layout brings the columns to 42 pixels wide, and when narrower than that, each column goes fluid, meaning the columns are stacked vertically and each column is the full width of the device (see Table 1).

Table 1. Responsive media queries

Label	Layout width	Column width	Gutter width
Large display	1200px and up	70px	30px
Default	980px and up	60px	20px
Portrait tablets	768px and up	42px	20px
Phones to tablets	767px and below	Fluid columns, no fixed widths	
Phones	480px and below	Fluid columns, no fixed widths	

To add custom CSS based on the media query, you can either include all rules in one CSS file via the media queries below, or use entirely different CSS files:

```

/* Large desktop */
@media (min-width: 1200px) { ... }

/* Portrait tablet to landscape and desktop */
@media (min-width: 768px) and (max-width: 979px) { ... }

/* Landscape phone to portrait tablet */
@media (max-width: 767px) { ... }

```



```
/* Landscape phones and down */
@media (max-width: 480px) { ... }
```

For a larger site, you might want to divide each media query into a separate CSS file. In the HTML file, you can call them with the `<link>` tag in the head of your document. This is useful for keeping file sizes smaller, but it does potentially increase the HTTP requests if the site is responsive. If you are using LESS to compile the CSS, you can have them all processed into one file:

```
<link rel="stylesheet" href="base.css" />
<link rel="stylesheet" media="(min-width: 1200px)" href="large.css" />
<link rel="stylesheet" media="(min-width: 768px) and (max-width: 979px)"
      href="tablet.css" />
<link rel="stylesheet" media="(max-width: 767px)" href="tablet.css" />
<link rel="stylesheet" media="(max-width: 480px)" href="phone.css" />
```

7.2.1 Helper classes

Bootstrap also includes a handful of helper classes for doing responsive development (see Table 2). Use these sparingly. A couple of use cases that have seen involve loading custom elements based on certain layouts.

Perhaps you have a really nice header on the main layout, but on mobile you want to pare it down, leaving only a few of the elements. In this scenario, you could use the `.hidden-phone` class to hide either parts or entire dom elements from the header.

Table 2. Media queries helper classes

Class	Phones	Tablets	Desktops
<code>.visible-phone</code>	Visible	Hidden	Hidden
<code>.visible-tablet</code>	Hidden	Visible	Hidden
<code>.visible-desktop</code>	Hidden	Hidden	Visible
<code>.hidden-phone</code>	Hidden	Visible	Visible
<code>.hidden-tablet</code>	Visible	Hidden	Visible
<code>.hidden-desktop</code>	Visible	Visible	Hidden

There are two major ways that you could look at doing development. The mantra that a lot of people are shouting now is that you should start with mobile, build to that platform, and let the desktop follow. Bootstrap almost forces the opposite, where you would create a full-featured desktop site that “just works.”

If you are looking for a strictly mobile framework, **Bootstrap** is still a great resource.

Keyword

Bootstrap is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.

7.3 BOOTSTRAP CSS

At the core of Bootstrap is a set of basic HTML elements that have been styled to allow for easy enhancement via classes and user styles.

7.3.1 Typography

Starting with typography, Bootstrap uses Helvetica Neue, Helvetica, Arial, and sansserif in its default font stack. These are all standard fonts and are included as defaults on all major computers. If by chance these fonts don't exist, they fall back to sansserif (the catchall) to tell the browser to use the default font for the browser. All body copy has the font-size set at 14 pixels, with the line-height set at 20 pixels. The `<p>` tag has a margin-bottom of 10 pixels, or half the line-height.

7.3.2 Headings

All six standard heading levels have been styled in Bootstrap, with the `<h1>` at 36 pixels tall, and the `<h6>` down to 12 pixels (for reference, default body text is 14 pixels tall). In addition, to add an inline subheading to any of the headings, simply add `<small>` around any of the elements and you will get smaller text in a lighter color. In the case of the `<h1>`, the small text is 24 pixels tall, normal font weight (i.e., not bold), and gray instead of black:

```
h1 small {
  font-size: 24px;
  font-weight: normal;
  line-height: 1;
  color: #999;
}
```

Heading 1

Heading 2

Heading 3

Heading 4

Heading 5

Heading 6

Figure 5. Headings.

7.3.3 Lead Body Copy

To add some emphasis to a paragraph, add `class="lead"` (see Figure 6). This will give you larger font size, lighter weight, and a taller line height. This is generally used for the first few paragraphs in a section, but it can really be used anywhere:

```
<p class="lead">Bacon ipsum dolor sit amet tri-tip pork loin ball tip frankfurter  
swine boudin meatloaf shoulder short ribs cow drumstick beef jowl.  
Meatball chicken sausage tail, kielbasa strip steak turducken venison prosciutto.  
Chuck filet mignon tri-tip ribeye, flank brisket leberkas. Swine  
turducken turkey shank, hamburger beef ribs bresaola pastrami venison rump.</p>
```

Lead Example

Bacon ipsum dolor sit amet tri-tip pork loin ball tip frankfurter swine boudin meatloaf shoulder short ribs cow drumstick beef jowl. Meatball chicken sausage tail, kielbasa strip steak turducken venison prosciutto. Chuck filet mignon tri-tip ribeye, flank brisket leberkas. Swine turducken turkey shank, hamburger beef ribs bresaola pastrami venison rump.

Figure 6. Lead body copy classes.

7.3.4 Emphasis

In addition to using the `<small>` tag within headings, as discussed above, you can also use it with body copy. When `<small>` is applied to body text, the font shrinks to 85% of its original size.

7.3.5 Bold

To add emphasis to text, simply wrap it in a `` tag. This will add `font-weight:bold;` to the selected text.

7.3.6 Italics

For italics, wrap your content in the `` tag. The term “em” derives from the word “emphasis” and is meant to add stress to your text.

7.3.7 Emphasis Classes

Along with Bootstrap offers a few other classes that can be used to provide emphasis (see Figure 7). These could be applied to paragraphs or spans:

```

<p class="muted">This content is muted</p>
<p class="text-warning">This content carries a warning class</p>
<p class="text-error">This content carries an error class</p>
<p class="text-info">This content carries an info class</p>
<p class="text-success">This content carries a success class</p>
<p>This content has <em>emphasis</em>, and can be <strong>bold</strong></p>

```



Figure 7. Emphasis classes.

7.3.8 Abbreviations

The HTML `<abbr>` element provides markup for abbreviations or acronyms, like WWW or HTTP (see Figure 8). By marking up abbreviations, you can give useful information to browsers, spell checkers, translation systems, or search engines. Bootstrap styles `<abbr>` elements with a light dotted border along the bottom and reveals the full text on hover (as long as you add that text to the `<abbr>` title attribute):

```

<abbr title="Real Simple Syndication">RSS</abbr>

```

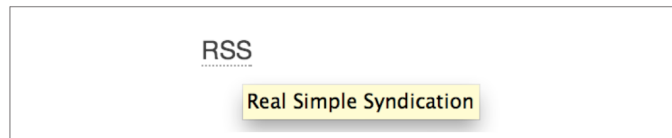


Figure 8. Abbreviation example.

Add `.initialism` to an `<abbr>` for a slightly smaller font size (see Figure 9):

```

<abbr title="rolling on the floor, laughing out loud">That joke had me ROTFLOL</abbr>

```



Figure 9. Another abbreviation example.

7.3.9 Addresses

Adding `<address>` elements to your page can help screen readers and search engines locate any physical addresses and phone numbers in the text (see Figure 10). It can also be used to mark up email addresses. Since the `<address>` defaults to `display: block`; you'll need to use `
` tags to add line breaks to the enclosed address text (e.g., to split the street address and city onto separate lines):

```
<address>
  <strong>O'Reilly Media, Inc.</strong><br>
  1005 Gravenstein HWY North<br>
  Sebastopol, CA 95472<br>
  <abbr title="Phone">P:</abbr> <a href="tel:+17078277000">(707) 827-7000</a>
</address>

<address>
  <strong>Jake Spurlock</strong><br>
  <a href="mailto:#">flast@oreilly.com</a>
</address>
```

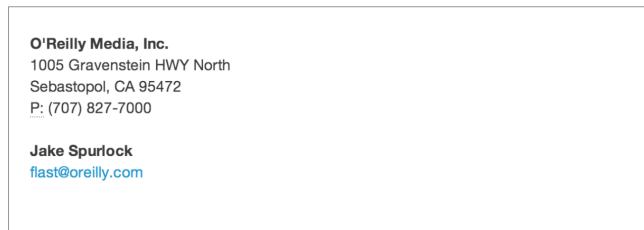


Figure 10. Address tag.

7.4 BLOCKQUOTES

To add blocks of quoted text to your document—or for any quotation that you want to set apart from the main text flow—add the `<blockquote>` tag around the text. For best results, and for line breaks, wrap each subsection in a `<p>` tag. Bootstrap's default styling indents the text and adds a thick gray border along the left side. To identify the source of the quote, add the `<small>` tag, then add the source's name wrapped in a `<cite>` tag before closing the `<small>` tag:

```

<blockquote>
  <p>That this is needed, desperately needed, is indicated by the
  incredible uptake of Bootstrap. I use it in all the server software
  I'm working on. And it shows through in the templating language I'm
  developing, so everyone who uses it will find it's "just there" and
  works, any time you want to do a Bootstrap technique. Nothing to do,
  no libraries to include. It's as if it were part of the hardware.
  Same approach that Apple took with the Mac OS in 1984.</p>
  <small>Developer of RSS, <cite title="Source Title">Dave Winer</cite>
</small>
</blockquote>

```

When you put it all together, you get something that looks like Figure 11.

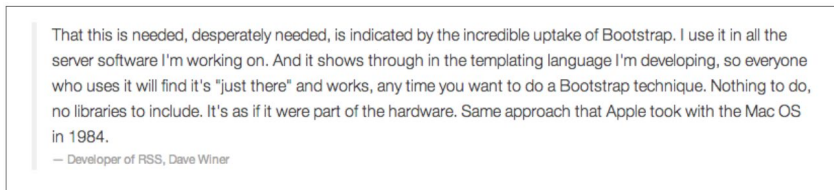


Figure 11. Basic blockquote.

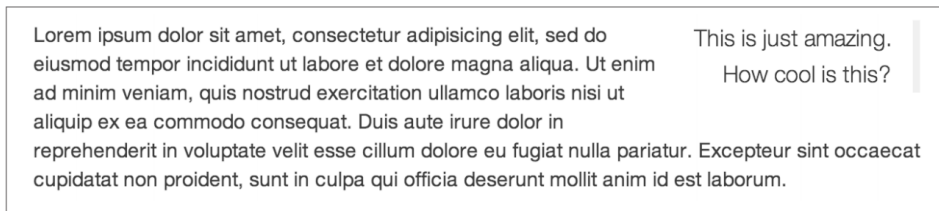


Figure 12. Pull right blockquote.

7.4.1 Lists

Bootstrap offers support and styling for the three main list types that HTML offers: ordered, unordered, and definition lists. An unordered list is a list that doesn't have any particular order and is traditionally styled with bullets.

Unordered list

If you have an ordered list that you would like to remove the bullets from, add `class="unstyled"` to the opening `` tag (see Figure 2-9):

```
<h3>Favorite Outdoor Activities</h3>
<ul>
  <li>Backpacking in Yosemite</li>
  <li>Hiking in Arches
    <ul>
      <li>Delicate Arch</li>
      <li>Park Avenue</li>
    </ul>
  </li>
  <li>Biking the Flintstones Trail</li>
</ul>
```

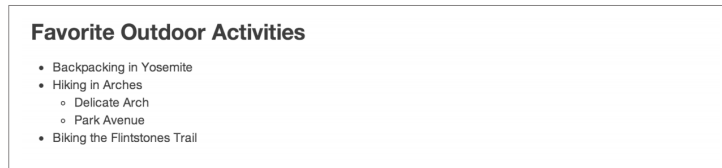


Figure 13. Unordered list.

Ordered list

An ordered list is a list that falls in some sort of sequential order and is prefaced by numbers rather than bullets (see Figure 14). This is handy when you want to build a list of numbered items like a task list, guide items, or even a list of comments on a blog post:

```
<h3>Self-Referential Task List</h3>
<ol>
  <li>Turn off the internet.</li>
  <li>Write the book.</li>
  <li>... Profit?</li>
</ol>
```

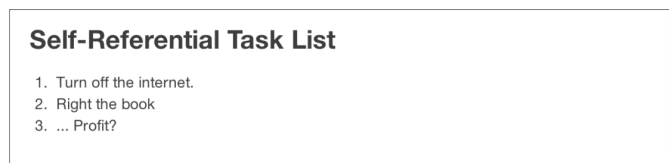


Figure 14. Ordered list

Definition list

The third type of list you get with Bootstrap is the definition list. The definition list differs from the ordered and unordered list in that instead of just having a block-level `` element, each list item can consist of both the `<dt>` and the `<dd>` elements. `<dt>` stands for “definition term,” and like a dictionary, this is the term (or phrase) that is being defined. Subsequently, the `<dd>` is the definition of the `<dt>`.

A lot of times in markup, you will see people using headings inside an unordered list. This works, but may not be the most semantic way to mark up the text. A better method would be creating a `<dl>` and then styling the `<dt>` and `<dd>` as you would the heading and the text (see Figure 15). That being said, Bootstrap offers some clean default styles and an option for a side-by-side layout of each definition:

```
<h3>Common Electronics Parts</h3>
<dl>
  <dt>LED</dt>
  <dd>A light-emitting diode (LED) is a semiconductor light source.</dd>
  <dt>Servo</dt>
  <dd>Servos are small, cheap, mass-produced actuators used for radio
    control and small robotics.</dd>
</dl>
```

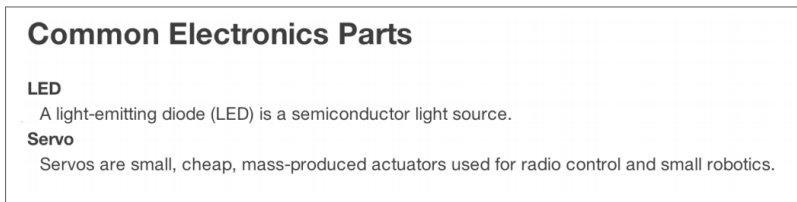


Figure 15. Definition list.

To change the `<dl>` to a horizontal layout, with the `<dt>` on the left side and the `<dd>` on the right, simply add `class="dl-horizontal"` to the opening tag (see Figure 16).

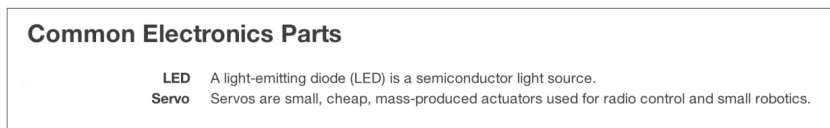


Figure 16. Horizontal definition list.

7.4.2 Code

There are two different key ways to display code with Bootstrap. The first is the `<code>` tag and the second is the `<pre>` tag. Generally, if you are going to be displaying code inline, you should use the `<code>` tag. But if the code needs to be displayed as a standalone block element or if it has multiple lines, then you should use the `<pre>` tag:

```
<p>Instead of always using divs, in HTML5, you can use new elements like
<code>&lt;section&gt;</code>, <code>&lt;header&gt;</code>, and
<code>&lt;footer&gt;</code>. The html should look something like this:</p>
<pre>
  &lt;article&gt;
    &lt;h1&gt;Article Heading&lt;/h1&gt;
    &lt;/article&gt;
</pre>
```

7.4.3 Tables

One of my favorite parts of Bootstrap is the nice way that tables are handled. We do a lot of work looking at and building tables, and the clean layout is a great feature that's included in Bootstrap right off the bat. Table 3 lists the various elements supported by Bootstrap.

Table 3. Table elements supported by Bootstrap

Tag	Description
<code><table></code>	Wrapping element for displaying data in a tabular format
<code><thead></code>	Container element for table header rows (<code><tr></code>) to label table columns
<code><tbody></code>	Container element for table rows (<code><tr></code>) in the body of the table
<code><tr></code>	Container element for a set of table cells (<code><td></code> or <code><th></code>) that appears on a single row
<code><td></code>	Default table cell
<code><th></code>	Special table cell for column (or row, depending on scope and placement) labels. Must be used within a <code><thead></code>
<code><caption></code>	Description or summary of what the table holds, especially useful for screen readers

If you want a nice, basic table style with just some light padding and horizontal dividers, add the base class of `.table` to any table (see Figure 17). The basic layout has a top border on all of the `<td>` elements:

```
<table class="table">
  <caption>...</caption>
  <thead>
    <tr>
      <th>...</th>
      <th>...</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>...</td>
      <td>...</td>
    </tr>
  </tbody>
</table>
```

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 17. Basic table class.

7.4.4 Optional Table Classes

Along with the base table markup and the `.table` class, there are a few additional classes that you can use to style the markup. These four classes are: `.table-striped`, `.table-bordered`, `.table-hover`, and `.table-condensed`.

Striped table

By adding the `.table-striped` class, you will get stripes on rows within the `<tbody>` (see Figure 18). This is done via the CSS `:nth-child` selector, which is not available on Internet Explorer 7–8.

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 18. Striped table class.

Bordered table

If you add the `.table-bordered` class, you will get borders surrounding every element and rounded corners around the entire table, as shown in Figure 19.

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 19. Bordered table class.

Hover table

Figure 20 shows the `.table-hover` class. A light gray background will be added to rows while the cursor hovers over them.

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 20. Hover table class.

Condensed table

If you add the `.table-condensed` class, as shown in Figure 21, row padding is cut in half to condense the table. This is useful if you want denser information.

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 21. Condensed table class.

Table Row Classes

The classes shown in Table 4 will allow you to change the background color of your rows (see Figure 22).

Table 4. Optional table row classes

Class	Description	Background color
<code>.success</code>	Indicates a successful or positive action.	Green
<code>.error</code>	Indicates a dangerous or potentially negative action.	Red
<code>.warning</code>	Indicates a warning that might need attention.	Yellow
<code>.info</code>	Used as an alternative to the default styles.	Blue

#	Product	Payment Taken	Status
1	TB - Monthly	01/04/2012	Approved
2	TB - Monthly	02/04/2012	Declined
3	TB - Monthly	03/04/2012	Pending
4	TB - Monthly	04/04/2012	Call in to confirm

Figure 22. Table row classes.

Did You Know?

On January 31, 2012, Bootstrap 2 was released, which added built-in support for Glyphicons, several new components, as well as changes to many of the existing components. This version supports responsive web design, meaning the layout of web pages adjusts dynamically, taking into account the characteristics of the device used (whether desktop, tablet, or mobile phone).

7.5 FORMS

Another one of the highlights of using Bootstrap is the ability to create forms with ease. As a web developer, styling forms is one of my least favorite tasks. Bootstrap makes it easy with the simple HTML markup and extended classes for different styles of forms.

The basic form structure comes with Bootstrap; there is no need to add any extra helper classes (see Figure 23). If you use the placeholder, keep in mind that it is only supported in newer browsers. In older browsers, no placeholder text will be displayed:

```
<form>
  <fieldset>
    <legend>Legend</legend>
    <label for="name">Label name</label>
    <input type="text" id="name"
      placeholder="Type something...">
    <span class="help-block">Example block-level help
      text here.</span>
    <label class="checkbox" for="checkbox">
      <input type="checkbox" id="checkbox">
      Check me out
    </label>
    <button type="submit" class="btn">Submit</button>
  </fieldset>
</form>
```

Figure 23. Basic form.

7.5.1 Optional Form Layouts

With a few helper classes, you can dynamically update the layout of your form. Bootstrap comes with a few preset styles to choose from.

Search form

Add `.form-search` to the `<form>` tag, and then add `.search-query` to the `<input>` for an input box with rounded corners and an inline submit button (see Figure 24):

```
<form class="form-search">
  <input type="text" class="input-medium search-query">
  <button type="submit" class="btn">Search</button>
</form>
```

A screenshot of a Bootstrap search form. It consists of a single horizontal line containing a rounded rectangular input field on the left and a button labeled "Search" on the right. The entire form is enclosed in a thin gray border.

Figure 24. Search form.

Inline form

To create a form where all of the elements are inline and labels are alongside, add the class `.form-inline` to the `<form>` tag (see Figure 25). To have the label and the input on the same line, use this inline form code:

```
<form class="form-inline">
  <input type="text" class="input-small" placeholder="Email">
  <input type="password" class="input-small" placeholder="Password">

  <label class="checkbox">
    <input type="checkbox" /> Remember me
  </label>
  <button type="submit" class="btn">Sign in</button>
</form>
```

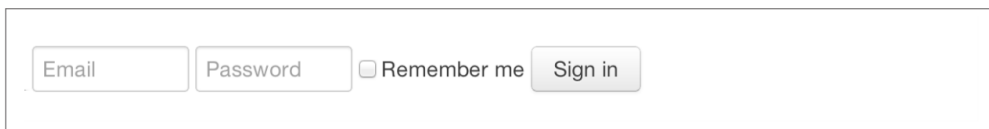
A screenshot of a Bootstrap inline form. The form is a single horizontal line containing four elements: an "Email" input field, a "Password" input field, a checkbox followed by the text "Remember me", and a "Sign in" button. The entire form is enclosed in a thin gray border.

Figure 25. Inline form.

Horizontal form

Bootstrap also comes with a prebaked horizontal form; this one stands apart from the others not only in the amount of markup, but also in the presentation of the form. Traditionally you'd use a table to get a form layout like the one shown in Figure 26, but Bootstrap manages to do it without using tables. Even better, if you're using the responsive CSS, the **horizontal form** will automatically adapt to smaller layouts by stacking the controls vertically.

To create a form that uses the horizontal layout, do the following:

- Add a class of `.form-horizontal` to the parent `<form>` element.
- Wrap labels and controls in a `<div>` with class `.control-group`.
- Add a class of `.control-label` to the labels.
- Wrap any associated controls in a `<div>` with class `.controls` for proper alignment.

Keyword

Horizontal form means that the labels are aligned next to the input field (horizontal) on large and medium screens.

Figure 26. Horizontal form.

```
<form class="form-horizontal">
  <div class="control-group">
    <label class="control-label" for="inputEmail">Email</label>

    <div class="controls">
      <input type="text" id="inputEmail" placeholder="Email">
    </div>
  </div>
  <div class="control-group">
    <label class="control-label" for="inputPassword">Password</label>
    <div class="controls">
      <input type="password" id="inputPassword" placeholder="Password">
    </div>
  </div>
  <div class="checkbox">
    <input type="checkbox" value="" checked="" /> Remember me
  </div>
  <button type="button" class="btn">Sign in</button>
</form>
```

```
<div class="control-group">
  <div class="controls">
    <label class="checkbox">
      <input type="checkbox"> Remember me
    </label>
    <button type="submit" class="btn">Sign in</button>
  </div>
</div>
</form>
```

7.5.2 Supported Form Controls

Bootstrap natively supports the most common form controls. Chief among them are input, textarea, checkbox, radio, and select.

Inputs

The most common form text field is the input—this is where users will enter most of the essential form data (see Figure 27). Bootstrap offers support for all native HTML5 input types: text, password, datetime, datetime-local, date, month, time, week, number, email, URL, search, tel, and color:

```
<input type="text" placeholder="Text input">
```

A screenshot of a single Bootstrap text input field. It is a rectangular box with a light gray border and a placeholder text "Text input" inside.

Figure 27. Input.

Textarea

The textarea is used when you need multiple lines of input (see Figure 2-24). You'll find you mainly modify the rows attribute, changing it to the number of rows that you need to support (fewer rows = smaller box, more rows = bigger box):

```
<textarea rows="3"></textarea>
```

A screenshot of a Bootstrap textarea field. It is a rectangular box with a light gray border, divided into three horizontal sections by thin gray lines, representing three rows of text input.

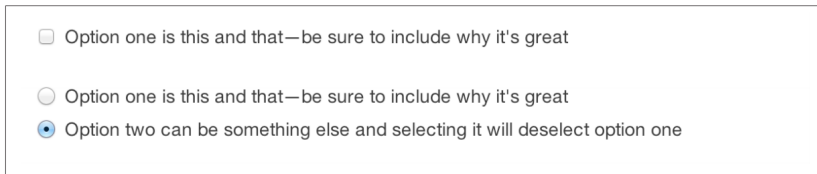
Figure 28. Both the :active default and the textarea.

Checkboxes and radio buttons

Checkboxes and radio buttons are great when you want users to choose from a list of preset options (see Figure 29). When building a form, use checkbox if you want the user to select any number of options from a list. Use radio if you want to limit him to just one selection:

```
<label class="checkbox">
  <input type="checkbox" value="">
  Option one is this and that—be sure to include why it's great.
</label>

<label class="radio">
  <input type="radio" name="optionsRadios" id="optionsRadios1" value="option1"
  checked>
  Option one is this and that—be sure to include why it's great.
</label>
<label class="radio">
  <input type="radio" name="optionsRadios" id="optionsRadios2" value="option2">
  Option two can be something else, and selecting it will deselect option one
</label>
```



☒ Option one is this and that—be sure to include why it's great

☐ Option one is this and that—be sure to include why it's great

☐ Option two can be something else and selecting it will deselect option one

Figure 29. Checkbox and radio buttons.

If you want multiple checkboxes to appear on the same line together, add the `inline` class to a series of checkboxes or radio buttons (see Figure 30):

```
<label for="option1" class="checkbox inline">
  <input id="option1" type="checkbox" id="inlineCheckbox1" value="option1"> 1
</label>
<label for="option2" class="checkbox inline">
  <input id="option2" type="checkbox" id="inlineCheckbox2" value="option2"> 2
</label>
<label for="option3" class="checkbox inline">
  <input id="option3" type="checkbox" id="inlineCheckbox3" value="option3"> 3
</label>
```



☒ 1 ☐ 2 ☐ 3

Figure 30. Inline checkboxes.

Selects

A select is used when you want to allow the user to pick from multiple options, but by default it only allows one (see Figure 31). It's best to use `<select>` for list options with which the user is familiar, such as states or numbers. Use `multiple="multiple"` to allow the user to select more than one option. If you only want the user to choose one option, use `type="radio"`:

```
<select>
  <option>1</option>
  <option>2</option>
  <option>3</option>
  <option>4</option>
  <option>5</option>
</select>

<select multiple="multiple">
  <option>1</option>
  <option>2</option>
  <option>3</option>
  <option>4</option>
  <option>5</option>
</select>
```

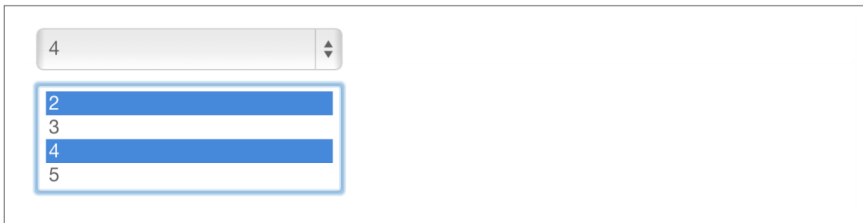


Figure 31. Select.

7.5.3 Extended Form Controls

In addition to the basic form controls listed in the previous section, Bootstrap offers a few other form components to complement the standard HTML form elements; for example, it lets you easily prepend and append content to inputs.

7.5.4 Prepended and Appended Inputs

By adding prepended and appended content to an input field, you can add common elements to the user's input (see Figure 32). For example, you can add the dollar symbol, the @ for a Twitter username, or anything else that might be common for

Keyword

Application interface is the set of features an application provides so that a user may supply input to, and receive output from, the program.

your **application interface**. To add extra content before the user input, wrap the prepended input in a `<div>` with class `.input-prepend`. To append input, use the class `.input-append`. Then, within that same `<div>`, place your extra content inside a `` with an `.add-on` class, and place the `` either before or after the `<input>` element:

```
<div class="input-prepend">
  <span class="add-on">@</span>
  <input class="span2" id="prependedInput" type="text" placeholder="Username">
</div>
<div class="input-append">
  <input class="span2" id="appendedInput" type="text">
  <span class="add-on">.00</span>
</div>
```

Figure 32. Prepend and append.

If you combine both of them, you simply need to add both the `.input-prepend` and `.input-append` classes to the parent `<div>` (see Figure 33):

```
<div class="input-prepend input-append">
  <span class="add-on">$</span>
  <input class="span2" id="appendedPrependedInput" type="text">
  <span class="add-on">.00</span>
</div>
```

Figure 33. Using both the append and prepend.

Rather than using a ``, you can instead use `<button>` with a class of `.btn` to attach (surprise!) a button or two to the input (see Figure 2-30):

```
<div class="input-append">
  <input class="span2" id="appendedInputButtons" type="text">
  <button class="btn" type="button">Search</button>
  <button class="btn" type="button">Options</button>
</div>
```

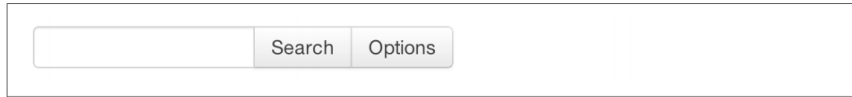


Figure 34. Attach multiple buttons to an input.

If you are appending a button to a search form, you will get the same nice rounded corners that you would expect (see Figure 35):

```
<form class="form-search">
  <div class="input-append">
    <input type="text" class="span2 search-query">
    <button type="submit" class="btn">Search</button>
  </div>
  <div class="input-prepend">
    <button type="submit" class="btn">Search</button>
    <input type="text" class="span2 search-query">
  </div>
</form>
```

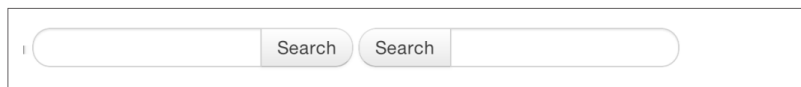


Figure 35. Append button to search form.

7.5.5 Form Control Sizing

With the default grid system that is inherent in Bootstrap, you can use the `.span*` system for sizing form controls. In addition to the span column-sizing method, you can also use a handful of classes that take a relative approach to sizing. If you want the input to act as a block-level element, you can add `.input-block-level` and it will be the full width of the container element, as shown in Figure 36:

```
<input class="input-block-level" type="text" placeholder=".input-block-level">
```

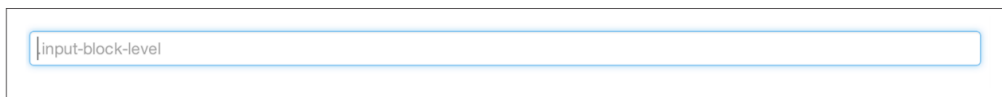


Figure 36. Block-level input.

Relative input controls

In addition to using `.span*` for input sizing, you can also use a few different class names (see Figure 37):

```
<input class="input-mini" type="text" placeholder=".input-mini">
<input class="input-small" type="text" placeholder=".input-small">
<input class="input-medium" type="text" placeholder=".input-medium">
<input class="input-large" type="text" placeholder=".input-large">
<input class="input-xlarge" type="text" placeholder=".input-xlarge">
<input class="input-xxlarge" type="text" placeholder=".input-xxlarge">
```



Figure 37. Relative input controls.

Grid sizing

You can use any `.span` from `.span1` to `.span12` for form control sizing (see Figure 38):

```
<input class="span1" type="text" placeholder=".span1">
<input class="span2" type="text" placeholder=".span2">
<input class="span3" type="text" placeholder=".span3">
<select class="span1">
  ...
</select>
<select class="span2">
  ...
</select>
<select class="span3">
  ...
</select>
```

Figure 38. Span-sized inputs.

If you want to use multiple inputs on a line, simply use the `.controls-row` modifier class to apply the proper spacing (see Figure 39). It floats the inputs to collapse the white space; sets the correct margins; and, like the `.row` class, clears the float:

```
<div class="controls">
  <input class="span5" type="text" placeholder=".span5">
</div>
<div class="controls controls-row">
  <input class="span4" type="text" placeholder=".span4">
  <input class="span1" type="text" placeholder=".span1">

</div>
...
```

Figure 39. Control row.

Uneditable text

If you want to present a form control without allowing the user to edit the input, simply add the class `.uneditable-input` (see Figure 40):

```
<span class="input-xlarge uneditable-input">Some value here</span>
```



Figure 40. Uneditable input.

7.5.6 Form Actions

When you place the form actions at the bottom of a `.horizontal-form`, the inputs will correctly line up with the floated form controls (see Figure 41):

```
<div class="form-actions">
  <button type="submit" class="btn btn-primary">Save changes</button>
  <button type="button" class="btn">Cancel</button>
</div>
```



Figure 41. Form controls.

Help text

Bootstrap form controls can have either block or inline text that flows with the inputs (see Figure 42):

```
<input type="text"><span class="help-inline">Inline help text</span>
```

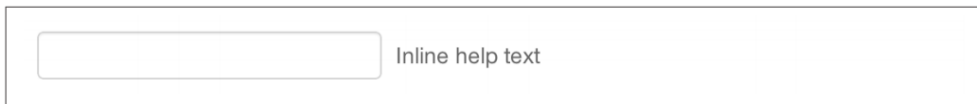


Figure 42. Inline help.

To add a full width block of content, use the `.help-block` after the `<input>` (see Figure 43):

```
<input type="text"><span class="help-block">A longer block of help text that
breaks onto a new line and may extend beyond one line.</span>
```

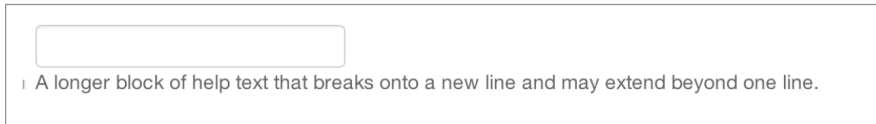


Figure 43. Block help.

7.5.7 Form Control States

In addition to the `:focus` state, Bootstrap offers styling for disabled inputs and classes for form validation.

Input focus

When an input receives `:focus` (i.e., a user clicks into the input or tabs onto it), the outline of the input is removed and a box-shadow is applied. I remember the first time that I saw this on Twitter's site; it blew me away, and I had to dig into the code to see how they did it. In WebKit, this is accomplished in the following manner:

```
input {
  -webkit-box-shadow: inset 0 1px 1px rgba(0, 0, 0, 0.075);
  -webkit-transition: box-shadow linear 0.2s;
}

input:focus {
  -webkit-box-shadow: inset 0 1px 1px rgba(0, 0, 0, 0.075), 0 0 8px
    rgba(82, 168, 236, 0.6);
}
```

The `<input>` has a small inset box-shadow, which gives the appearance that the input sits lower than the page (see Figure 44). When `:focus` is applied, an 8px light blue border appears. The `webkit-transition` tells the browser to apply the effect in a linear manner over 0.2 seconds:

```
<input class="input-xlarge" id="focusedInput" type="text"
  value="This is focused...">
```



Figure 44. Focused input.

Nice and subtle; a great effect.

Disabled input

If you need to disable an input, simply adding the disabled attribute will not only disable it; it will also change the styling and the mouse cursor when the cursor hovers over the element (see Figure 45):

```
<input class="input-xlarge" id="disabledInput" type="text"
      placeholder="Disabled input here..." disabled>
```


 A screenshot of a single disabled text input field. The input is light gray with rounded corners and contains the placeholder text "Disabled input here...". It is centered within a white rectangular container.

Figure 45. Disabled input.

Validation states

Bootstrap includes validation styles for error, warning, info, and success messages (see Figure 46). To use, simply add the appropriate class to the surrounding `.controlgroup`:

```
<div class="control-group warning">
  <label class="control-label" for="inputWarning">Input with warning</label>
  <div class="controls">
    <input type="text" id="inputWarning">
    <span class="help-inline">Something may have gone wrong</span>
  </div>
</div>
<div class="control-group error">
  <label class="control-label" for="inputError">Input with error</label>
  <div class="controls">
    <input type="text" id="inputError">
    <span class="help-inline">Please correct the error</span>
  </div>
</div>
<div class="control-group success">
  <label class="control-label" for="inputSuccess">Input with success</label>
  <div class="controls">
    <input type="text" id="inputSuccess">
    <span class="help-inline">Woohoo!</span>
  </div>
</div>
```



 A screenshot showing four rows of form elements, each with a label, an input field, and a message.
 1. "Input with warning" with an orange-bordered input and the message "Something may have gone wrong".
 2. "Input with error" with a red-bordered input and the message "Please correct the error".
 3. "Input with info" with a blue-bordered input and the message "Username is taken".
 4. "Input with success" with a green-bordered input and the message "Woohoo!".

Figure 46. Validation states.

7.6 BUTTONS









One of my favorite features of Bootstrap is the way that buttons are styled. Dave Winer, inventor of RSS and big fan of Bootstrap, has this to say about it:

That this is needed, desperately needed, is indicated by the incredible uptake of Bootstrap. I use it in all the server software I'm working on. And it shows through in the templating language I'm developing, so everyone who uses it will find it's "just there" and works, any time you want to do a Bootstrap technique. Nothing to do, no libraries to include. It's as if it were part of the hardware. Same approach that Apple took with the Mac OS in 1984. — Dave Winer

Bootstrap is unifying the Web and allowing a unified experience of what an interface can look like across the Web. With the advent of Bootstrap, you can usually spot the sites that have adopted it by the buttons that they use. A grid layout and many of the other features fade into the background, but buttons, forms, and other unifying elements are a key part of Bootstrap.

Now, buttons and links can all look alike with Bootstrap. Anything that is given a class of `.btn` will inherit the default look of a gray button with rounded corners. However, you can add color to the buttons by adding extra classes (see Table 5).

Table 5. Button color examples

Buttons	Class	Description
	<code>btn</code>	Standard gray button with gradient
	<code>btn btn-primary</code>	Provides extra visual weight and identifies the primary action in a set of buttons (blue)
	<code>btn btn-info</code>	Used as an alternative to the default styles (light blue)
	<code>btn btn-success</code>	Indicates a successful or positive action (green)
	<code>btn btn-warning</code>	Indicates caution should be taken with this action (orange)
	<code>btn btn-danger</code>	Indicates a dangerous or potentially negative action (red)
	<code>btn btn-inverse</code>	Alternate dark-gray button, not tied to a semantic action or use
	<code>btn btn-link</code>	De-emphasizes a button by making it look like a link while maintaining button behavior

7.6.1 Button Sizes

If you need larger or smaller buttons, simply add `.btn-large`, `.btn-small`, or `.btn-mini` to links or buttons (see Figure 2-43):

```

<p>
  <button class="btn btn-large btn-primary" type="button">Large button</button>
  <button class="btn btn-large" type="button">Large button</button>
</p>
<p>
  <button class="btn btn-primary" type="button">Default button</button>
  <button class="btn" type="button">Default button</button>
</p>
<p>
  <button class="btn btn-small btn-primary" type="button">Small button</button>

  <button class="btn btn-small" type="button">Small button</button>
</p>
<p>
  <button class="btn btn-mini btn-primary" type="button">Mini button</button>
  <button class="btn btn-mini" type="button">Mini button</button>
</p>

```

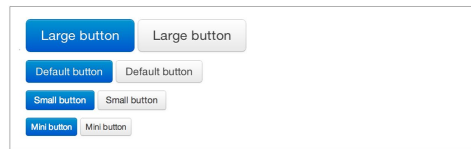


Figure 47. Different button sizes.

If you want to create buttons that display like a block-level element, simply add the `.btnblock` class (see Figure 48). These buttons will display at 100% width:

```

<button class="btn btn-large btn-block btn-primary" type="button">Block-
level button</button>
<button class="btn btn-large btn-block" type="button">Block-level button</button>

```

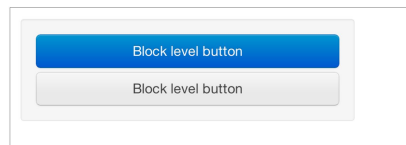


Figure 48. Block-level button.

7.6.2 Disabled Button Styling

For anchor elements, simply add the class of `.disabled` to the tag and the link will fade in color, and lose the gradient (see Figure 49):

```

<a href="#" class="btn btn-large btn-primary disabled">Primary link</a>
<a href="#" class="btn btn-large disabled">Link</a>

```



Figure 49. Disabled link.

For a button, simply add the `disabled` attribute to the button (see Figure 50). This will actually disable the button, so JavaScript is not directly needed:

```
<button type="button" class="btn btn-large btn-primary disabled"
disabled="disabled">Primary button</button>
<button type="button" class="btn btn-large" disabled>Button</button>
```



Figure 50. Disabled button.

Remember

The `.disabled` class is being used much like the `.active` class. So, there's no `.btn` prefix, and remember, this is only for looks. You will need to use some JavaScript to actually disable the link.

7.6.3 Images

Images have three classes (see Figure 51) that can be used to apply some simple styles: `.img-rounded` adds `border-radius:6px` to give the image rounded corners, `.img-circle` makes the entire image round by adding `border-radius: 500px`, and `.img-polaroid` adds a bit of padding and a gray border:

```



```



Figure 51. Images.

7.6.4 Icons

Bootstrap bundles 140 icons into one sprite that can be used with buttons, links, navigation, and form fields. The icons are provided by GLYPHICONS; see Figure 52.

 icon-glass	 icon-music	 icon-search	 icon-envelope
 icon-heart	 icon-star	 icon-star-empty	 icon-user
 icon-film	 icon-th-large	 icon-th	 icon-th-list
 icon-ok	 icon-remove	 icon-zoom-in	 icon-zoom-out
 icon-off	 icon-signal	 icon-cog	 icon-trash
 icon-home	 icon-file	 icon-time	 icon-road
 icon-download-alt	 icon-download	 icon-upload	 icon-inbox
 icon-play-circle	 icon-repeat	 icon-refresh	 icon-list-alt
 icon-lock	 icon-flag	 icon-headphones	 icon-volume-off
 icon-volume-down	 icon-volume-up	 icon-qrcode	 icon-barcode
 icon-tag	 icon-tags	 icon-book	 icon-bookmark
 icon-print	 icon-camera	 icon-font	 icon-bold
 icon-italic	 icon-text-height	 icon-text-width	 icon-align-left
 icon-align-center	 icon-align-right	 icon-align-justify	 icon-list
 icon-indent-left	 icon-indent-right	 icon-facetime-video	 icon-picture
 icon-pencil	 icon-map-marker	 icon-adjust	 icon-tint
 icon-edit	 icon-share	 icon-check	 icon-move
 icon-step-backward	 icon-fast-backward	 icon-backward	 icon-play
 icon-pause	 icon-stop	 icon-forward	 icon-fast-forward
 icon-step-forward	 icon-eject	 icon-chevron-left	 icon-chevron-right
 icon-plus-sign	 icon-minus-sign	 icon-remove-sign	 icon-ok-sign
 icon-question-sign	 icon-info-sign	 icon-screenshot	 icon-remove-circle
 icon-ok-circle	 icon-ban-circle	 icon-arrow-left	 icon-arrow-right
 icon-arrow-up	 icon-arrow-down	 icon-share-alt	 icon-resize-full
 icon-resize-small	 icon-plus	 icon-minus	 icon-asterisk
 icon-exclamation-sign	 icon-gift	 icon-leaf	 icon-fire
 icon-eye-open	 icon-eye-close	 icon-warning-sign	 icon-plane
 icon-calendar	 icon-random	 icon-comment	 icon-magnet
 icon-chevron-up	 icon-chevron-down	 icon-retweet	 icon-shopping-cart
 icon-folder-close	 icon-folder-open	 icon-resize-vertical	 icon-resize-horizontal
 icon-hdd	 icon-bullhorn	 icon-bell	 icon-certificate
 icon-thumbs-up	 icon-thumbs-down	 icon-hand-right	 icon-hand-left
 icon-hand-up	 icon-hand-down	 icon-circle-arrow-right	 icon-circle-arrow-left
 icon-circle-arrow-up	 icon-circle-arrow-down	 icon-globe	 icon-wrench
 icon-tasks	 icon-filter	 icon-briefcase	 icon-fullscreen

Figure 52. Icons by GLYPHICONS.

SUMMARY

- Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.
- The Bootstrap download includes three folders: css, js, and img. For simplicity, add these to the root of your project
- To create a simple layout, create a container with a that has a class of .row and add the appropriate amount of .span* columns.
- To nest your content with the default grid, inside of a .span*, simply add a new .row with enough .span* that it equals the number of spans of the parent container.
- The fluid grid system uses percentages instead of pixels for column widths. It has the same responsive capabilities as our fixed grid system, ensuring proper proportions for key screen resolutions and devices.
- To add blocks of quoted text to your document—or for any quotation that you want to set apart from the main text flow—add the <blockquote> tag around the text.

KNOWLEDGE CHECK

1. **Who developed the bootstrap?**
 - a. James Gosling
 - b. Mark Jukervich
 - c. Dennis Ritchie
 - d. Mark Otto and Jacob Thornton
2. **Which of the following class in Bootstrap is used to provide a responsive fixed width container?**
 - a. container-fixed
 - d. container-fluid
 - c. container
 - d. All of the above
3. **How many columns are allowed in a bootstrap grid system?**
 - a. 2
 - b. 12
 - c. 3
 - d. 5
4. **Which of the following class in bootstrap is used to create a big box for calling extra attention?**
 - a. box
 - b. container
 - c. container-fluid
 - d. jumbotron
5. **Which of the following class is used to create a black navigation bar?**
 - a. navbar-default
 - b. navbar-inverse
 - c. navbar-black
 - d. navbar-dark

REVIEW QUESTIONS

1. Define blockquotes.
2. Illustrate the form layouts.
3. Describe the supported form controls.
4. What are the prepended and appended inputs?
5. How to create buttons in bootstrap?

Check Your Result

1. (d) 2. (c) 3. (b) 4. (d) 5. (b)

REFERENCES

1. Jan Freeman, Bootstraps and Baron Munchausen, Boston.com, January 27, 2009
2. Otto, Mark (October 29, 2014). "Bootstrap 3.3.0 released". Archived from the original on July 24, 2016. Retrieved February 23, 2017.
3. Otto, Mark (August 19, 2011). "Bootstrap from Twitter". Developer Blog. Twitter. Archived from the original on February 23, 2017. Retrieved February 23, 2017.
4. Otto, Mark (January 17, 2012). "Bootstrap in A List Apart No. 342". Mark Otto's blog. Archived from the original on October 28, 2016. Retrieved February 23, 2017.
5. Otto, Mark (January 31, 2012). "Say hello to Bootstrap 2.0". Developer Blog. Twitter. Archived from the original on February 23, 2017. Retrieved February 23, 2017.
6. Otto, Mark (August 19, 2013). "Bootstrap 3 released". Archived from the original on October 21, 2016. Retrieved February 23, 2017.
7. Otto, Mark (August 19, 2015). "Bootstrap 4 alpha". Archived from the original on January 23, 2017. Retrieved February 23, 2017.
8. "About". Bootstrap. Retrieved February 23, 2017.

Index

A

Actual interface element 209
Attribute selector 152, 160, 161

B

Beginner web designer 208
Bootstrap 249, 250, 251, 252, 255, 256, 257, 258, 259, 261, 262, 264, 265, 268, 269, 270, 271, 273, 275, 278, 279, 280, 281, 283, 285, 286, 288
Business potential 206
Business strategy 33

C

Cascading Style Sheets (CSS) 105
Clothing design software 86
Common gateway interface (CGI) 231
Common visual language 207, 244
Communication method 206
Competitive analysis 208
Computer aided design (CAD) 48, 57, 59
Computer based design software 77
Computer browser 13
Computer numeric control (CNC) 66
Computer sketching 77, 101
Content development 226, 227
Content management systems (CMS) 232

Convincing sketches 78
CSS stylesheet 252

D

Descendant selector 149
Design framework 210
Design language 207
Design project 211
Design suggestion 21
Direct metal laser sintering (DMLS) 57, 59
Document type declaration (DTD) 115
Document type definition (DTD) 121
Domain Name Servers (DNS) 230
DOM (Document Object Model) 169
Dynamic game balancing (DGB) 36

E

Editorial Review Board (ERB) 142
Electron beam melting (EBM) 59
Extensible HyperText Markup Language (XHTML) 105
Extensible Markup Language (XML) 105
External style sheet 143

F

Fashion sketches 84, 85
Firefox 170

First Person Perspective (FPP) 36

Fixed grid system 254, 285

Flexibility 208, 237

Fluid grid 254, 255, 285

Fused Deposition Modeling (FDM) 55

G

Gameplay duration 36

H

Hand sketching 84

HTML elements 146, 148, 159

HyperText Markup Language (HTML) 118, 132

Hypertext Preprocessor (PHP) 232

Hypertext transfer protocol (HTTP) 231

I

ID selector 147, 148, 160

Inheritance 153, 154

Inline style 143

Interior designer 207

Internal style sheet 143

Internet Engineering Task Force (IETF) 120, 139

Internet Explorer 164, 165, 166, 170

Internet stationary 242

Internet traffic 2, 3

J

JavaScript 163, 164, 165, 166, 167, 168, 169, 170, 171, 172

JavaScript development 167

L

Layout transform 25

Local Area Network (LAN) 206

M

Machine language code 224

Manufacture prototype castings 66

Markup language 106, 115, 117, 131, 132, 133, 135, 139, 223

Message-based design 241

Metropolitan Area Network (MAN) 206

Mobile content marketing 181, 182, 184, 185, 186, 187, 188, 203

N

National Center for Supercomputing Applications (NCSA) 164

P

Parent container 254, 285

Pen-based user interface 79

Portable Document Format (PDF) 46

Post-production machining 65, 66

Proposed Recommendation (PR) 139

R

Rapid prototype model 66, 67, 71

Rapid Prototyping (RP) 52

Rapid Tooling 47

Relevant information 210

Responsive CSS file 252, 255

Responsive design 192, 194, 196, 256

Responsive web design (RWD) 2, 3

S

Scalar Vector Graphics (SVG) 139

Scientific application 76

Significant aspect 33

Standard Generalized Markup Language (SGML) 118

Stereo-lithography process 47, 54

Stylistic approach 211

T

Traditional desktop website 190
Typography 207, 209, 250, 252, 258

U

Unix Circuit Design System (UCDS) 53

V

Visual language 210, 211, 212
Visual language perspective 211

W

Web browsers 164
Web Hypertext Application Technology
Working Group (WHATWG) 139
Web pages 140, 141, 142, 143, 145, 163, 168,
169
Web technology 206, 223, 225, 241
Wide Area Network (WAN) 206
World Wide Web Consortium (W3C) 105,
107, 124, 131, 133, 139, 142, 173, 175, 240
World-Wide Web (WWW) 118

Level: Beginner to Advanced
Subject: Computer and Information Science

Basic Computer Coding: Responsive Design

2nd Edition

In the field of Web design and development, we're quickly getting to the point of being unable to keep up with the endless new resolutions and devices. For many websites, creating a website version for each resolution and new device would be impossible, or at least impractical. Responsive Web design is the approach that suggests that design and development should respond to the user's behavior and environment based on screen size, platform and orientation. This book introduces students to responsive web design using HTML (Hypertext Markup Language) and CSS (Cascading Style Sheets). Throughout the book students are introduced to planning and designing effective web pages; implementing web pages by writing HTML, CSS code, and JavaScript; enhancing web pages with the use of page layout techniques, text formatting, graphics, images, and multimedia; and producing a functional, multi-page website.

This edition is organized into seven chapters. In this book you'll learn the fundamentals of responsive web design using the viewport tag and CSS media queries. Additionally, you will learn how to apply concepts from interaction design and human computer interaction in order to design and build an interactive, professional looking website.