Basic Markup

Unit: Programming Languages

Problem Area: HTML

Lesson: Basic Markup

Student Learning Objectives.

Instruction in this lesson should result in students achieving the following objectives:

1. **Review scripting language.**
2. **Use text editors.**
3. **Summarize basic website structures.**
4. **Create valid, well-indented code.**
5. **Use open-source software.**

Resources.

The following resources may be useful in teaching this lesson:

E-unit(s) corresponding to this lesson plan. CAERT, Inc. http://www.mycaert.com.

"A Brief SGML Tutorial," *W3C*. Accessed June 24, 2016. https://www.w3.org/TR/WD-html40-970708/intro/sgmltut.html.

"Google HTML/CSS Style Guide," *Google and GitHub*. Accessed June 24, 2016. https://google.github.io/styleguide/htmlcssguide.xml.

Madden, Jennifer. "Add-ons," *Mozilla*. Accessed June 24, 2016. https://addons.mozilla.org/en-US/firefox/addon/view-source-chart/.

"Markup Language," *Wikipedia*. Accessed June 24, 2016. https://en.wikipedia.org/wiki/Markup_language.

Shannon, Ross. "The History of HTML," *HTML Source*. Accessed June 24, 2016. http://www.yourhtmlsource.com/starthere/historyofhtml.html/.

Equipment, Tools, Supplies, and Facilities

- ✓ Overhead or PowerPoint projector
- ✓ Visual(s) from accompanying master(s)
- ✓ Copies of sample test, lab sheet(s), and/or other items designed for duplication
- ✓ Materials listed on duplicated items
- ✓ Computers with printers and Internet access
- ✓ Classroom resource and reference materials

Key Terms.

The following terms are presented in this lesson (shown in bold italics):

- binary code
- body element
- CLI
- comment statements
- compiler
- element
- file extension
- Git
- GML
- GUI
- head element
- heading element
- HTML
- index.html
- interpreter
- list element
- machine code file
- markup
- markup languages
- object code
- open source software
- paragraph element
- programs
- SCM
- scripting languages
- scripts
- SGML
- source code
- SVN
- syntax
- table element
- tag
- VCS
- W3C

Interest Approach.

Use an interest approach that will prepare the students for the lesson. Teachers often develop approaches for their unique class and student situations. A possible approach is included here.

> *Today, the main source of information is the World Wide Web. Understanding the basics of data representation on the web makes it easier to retrieve and utilize its information. It also allows people to create content for the web.*

CONTENT SUMMARY AND TEACHING STRATEGIES

Objective 1: Review scripting language.

Anticipated Problem: What is scripting language?

I. Scripting language

A. Markup and code files

1. ***Markup*** is the sequence of characters or other symbols inserted at certain places in a text or word-processing file to indicate how the file should look when it is printed or displayed or to describe the file's logical structure. Markup indicators are often called "tags."
2. Computers are composed of electrical circuits. Program logic is implemented at the machine level (on the motherboard) by pushing electricity through a series of paths in a circuit, much like traversing a maze. In essence, a program is a set of instructions about which circuits should be electrified and which ones should be left un-electrified. These two states (electrified vs. un-electrified) are depicted as 0s and 1s. Therefore, a computer machine is referred to as a binary machine.
3. An executable file is referred to as a machine code file. A ***machine code file*** is a file composed of binary code converted into instructions that designate which circuit paths are to be used on a computer's motherboard.
4. ***Binary code*** is the simplest form of computer code that uses only two digits: 0 and 1. At the machine level, instructions are composed of patterns of zeros and ones. The instruction set for a specific PC chip contains the sequences for each computer operation that can be performed in that machine. Humans do not think in zeros and ones (i.e., in binary code), which makes it difficult to write a program in machine code format. Instead, programs are written in a computer language using English-like instructions. Then they are saved in a file. (See VM–A.) The instruction to display "Hello" on the screen is shown in various computer languages. Files with English-like instructions instead of machine code are referred to as ***source code***. Source code files cannot be executed since they are not written in machine code. They need to be translated into machine code using a compiler or an interpreter.
5. ***Syntax*** is the spelling and grammar of a programming language. Each language has its own rules for what is considered "correct" and "incorrect" or the language's syntax. To become a proficient programmer in a specific language, it is essential to learn the syntax rules. Syntax rules vary from one computer language to another.

B. Compiler and interpreter

1. A ***compiler*** is a special program that reads a source code file and translates the instructions coded inside it into machine code. (See VM–B.) A file with output produced by a compiler is called ***object code***. The output produced by the compiler is saved. (See VM–C.) It lists commonly used compiled languages. Once a program has been compiled, the machine code file can be used to execute the program. If a change is required to be made to the program, the source code file is changed and recompiled to generate a new machine code file.
2. An ***interpreter*** is a program similar to a compiler as it translates source code into machine code. Once the translation is completed, the machine code file is executed and then deleted. (See VM–D.) At the end of the process, only the source code file remains. The next time the program is to be executed, the source code file is interpreted. Then the machine code is generated, executed, and deleted. (See VM–E.) It lists commonly used scripting languages, along with their usage.

C. Programs versus scripts

1. ***Programs*** are source code files compiled to create machine code files that can be saved. ***Scripts*** are source code files interpreted into machine code, which is discarded after execution. ***Scripting languages*** are programming languages that make use of interpreters.
2. Scripts are easier to manage and deploy than programs, as there is just one file (the source code file) that needs to be maintained. Programs are harder to maintain because each program is composed of two files—a source code file and a machine code file. On the other hand, compiled code executes faster than interpreted code because no time is needed to translate source code into machine code. (See VM–F.) It shows the advantages and disadvantages of compiled and interpreted code.

D. Markup languages

1. Prior to the days of personal computers (PCs), programmers worked with "dumb" terminals directly connected to a mainframe computer. Screen displays for "dumb" terminals were line based and lacked the ability to display graphics or formatted text. To format text-based files, special text commands were inserted into the text file to specify formatting information. Commands were used when printing the file. Scientists at IBM developed a markup language called Generalized Markup Language (GML).
   a. ***GML*** is a markup language that describes a file in terms of its organization structure, content parts, and their relationship. GML was used to create Standard Generalized Markup Language or ***SGML*** standards that specify how a markup language should be defined. The most common implementation of SGML is HTML.
   b. Hypertext Markup Language or ***HTML*** is a markup language used to create webpages. HTML files contain content to be displayed and information about how content should be rendered. A HTML file is saved with an

extension of "html," where all the alphabets are lowercase. The htm extension may also be used.

2. A markup language differs from a computer programming language (compiled or interpreted) in many respects. (See VM–G.) Computer programming languages focus on implementing logic structures and performing calculations. ***Markup languages*** are languages that focus on content presentation. In general, programming languages generate data, which is presented by markup languages.

E. CLI and GUI programs

1. Before PCs, programs were written requesting information from users processed to display output. These applications worked with Command Line Interface (CLI), which was line oriented. ***CLI*** is a user interface to a computer's operating system or an application in which the user responds to a visual prompt by typing in a command on a specified line. Then the user receives a response back from the system, and so on.
2. With the advent of PCs, applications were able to provide richer interfaces to end-users called Graphical User Interface (GUI; pronounced GOO-ee). ***GUI*** is a graphical user interface rather than a text-only interface (as with CLI).
3. See VM–H for a comparison of CLI and GUI interfaces. Both displays show the files maintained in a folder. The CLI interface lacks the images and graphical content that appear in the GUI version.

Teaching Strategy: *Many techniques can be used to help students master this objective. Use VM–A through VM–H to review.*

Objective 2: Use text editors.

Anticipated Problem: What is a text editor?

II. Using a text editor

A. File types and extensions

1. A file in a computer is required to have a filename. Additionally, it has a file extension. A ***file extension*** is the suffix at the end of a filename that indicates the type of file (e.g., "mylabreport.txt" indicates the file is a text file). The filename and extension are separated by a dot (".") Depending on the computer settings, file extensions may or may not be visible. (See VM–I.) It shows files displayed with and without file extensions.
2. Windows displays an icon across each file. The icon displayed is based on the file extension that determines the software application that opens a file. For example, a Microsoft (MS) Word 2013 file has an extension of "docx." When double-clicked, Word 2013 opens the file because each file extension is registered to one—and only one—application. When a file is double-clicked, it is opened by the application to which the file extension is registered. (VM–J

shows some of the common file extensions and the applications to which each is registered. It also shows the two different icons that may be displayed across text files depending upon the application that "owns" the txt extension.)

B. HTML file extensions

1. HTML files must be saved with an html or an htm extension. HTML files are ASCII files (or text files). A text file contains alphabets, numbers, punctuation, and the data generated by hitting the "Enter" key. It does not contain any formatting information about font, color, font size, or highlighting. Text files are created using simple text editors. Notepad is a text editor that is part of the Windows operating system. It can be found under the "Windows Accessories" group. Text files are typically saved with the "txt" extension. Notepad, the default application that opens text files, is a no-frills interface.
2. Microsoft Word should not be used to create html files. MS Word files are always stored with formatting information (e.g., font name, font size and color, word count, page count, and more).

C. Creating HTML files in NotePad++

1. NotePad++ is a popular, free alternative to Notepad and can be downloaded. It provides a nicer interface than Notepad and provides color-coding based on the file extension. (See VM–K.) It displays the same file in Notepad and Notepad++. NotePad++ displays line numbers across each line in the file. This feature can be helpful when debugging errors in a file. NotePad++ can open multiple files and display each one under a separate tab.
2. NotePad++, like Notepad, saves a file with a txt extension. When creating an HTML file, steps must be taken to save the file with "html" as the extension instead of the default "txt" extension. (See VM–L.) It displays the steps to save a file created in NotePad++ with "html" as the file extension. For example, once the file content is keyed in, the user must:
   a. Click on the "Save As" option from the File menu option.
   b. Key in the name of the file followed by ".html".
   c. Click on the "Save as Type" button and select the "Hyper Text Markup Language file" option. This saves the file with the html extension, marking it as a text HTML file.

Teaching Strategy: *Many techniques can be used to help students master this objective. Use VM–I through VM–L to review.*

Objective 3: Summarize basic website structures.

Anticipated Problem: How are files organized on a website?

III. Develop a website structure

A. Storyboard a website

1. Creating a website involves the development of many HTML files, along with audio, video, and image files to build the site. These files need to be organized in a logical fashion so website maintenance tasks are easy to perform. Most websites contain a file called ***index.html***, which is a default page that automatically displays when the Website URL is entered in a browser address bar. The index page is the "root" of the website, and all other pages are linked to it. For example, the URL "http://www.mysite.com" will look for a webpage within the website called "index.html." The name of the file must be "index," but the extension may be .php, .jsp, or .aspx, depending upon the website and the file extension of the index file.
2. Storyboarding (a graphic organizer of the project) can be performed using a variety of software. (See VM–M.) It lists various storyboarding software applications.
   a. MS Word PowerPoint provides features that make creation of a website layout/storyboard easy. (See VM–N.) It shows the site layout of a simple website, along with the names of the files in the website.
   b. The names of all the webpages in the website follow the format of filename-as-one-word.html. It is important to create filenames that do not contain an embedded blank. For example, "List of products.html" is not an appropriate webpage name because it does not translate correctly into a URL. For example, a space embedded in an HTML filename is replaced in the URL by the characters "%20" (without the quotes).

B. Websites and how they work

1. A website is a collection of HTML files, images, and programs that execute on the web server and other required resources. These components are maintained on a web server machine. A web server can be envisioned as a machine connected to the web, with a vast array of hard disks housing thousands of webpages.
   a. See VM–O. When a user working on a PC or a mobile device keys in a URL in a browser window, a request is sent to the web server that corresponds to that URL. If the file extension of the page is "html," the file is sent back to the device that made the request. If the request is for a web server program (which means its extension is php, aspx, or jsp), the program is executed and the stream of HTML produced by the program is sent back to the client PC.
   b. VM–P lists extensions used by some of the popular and common web server programs in use today.

2. The client PC that made the initial URL request receives HTML from a file with the html extension or from a stream of HTML produced by a web server program. The HTML code is interpreted by the browser software on the PC and is rendered on screen.
    a. Multiple web browser software applications are on the market (e.g., Microsoft Edge, Google Chrome, and Firefox). While they all render HTML code, the presentation may be different.
    b. Additionally, it is possible to customize browser software applications. The same page may render differently on two different PCs using the same browser software due to differences in browser software settings.

C. HTML versions

1. Several versions of HTML have existed since the beginning of the web. The World Wide Web Consortium (***W3C***) governs the version standards. VM–Q lists the various HTML versions and the years that the specifications were standardized. HTML 4.0 dropped some of the elements used in earlier versions and declared them as deprecated, which meant that they were denounced by W3C and were no longer recommended for use.
2. Extensible Hypertext Markup Language (XHTML) represented the next stage in HTML development. It attempted to enforce the rigor of XML specifications into the HTML language. Attempts to extend XHTML to XHTML 2 were not pursued. Instead, a group called Web Hypertext Application Technology Working Group (WHATWG) started work on a new standard called HTML5. (NOTE: There is no space between HTML and the number 5.) HTML5 maintains most of the work done in earlier versions of HTML while allowing for future enhancements.

D. Structuring a webpage

1. Webpages are created as text files with an HTML extension. Since an HTML file is essentially a text file, the first line in an HTML file identifies it as an HTML file and not a simple text file. The identification line in an HTML file looks different in various versions of HTML. The statement shown below is the declaration statement for HTML5 files.

<!DOCTYPE html>

2. ***Comment statements*** are programmer-readable annotations (footnotes or explanations) in the source code placed in an HTML file to provide information about the file, such as the author of the file and the reason for its creation. The browser software ignores comments. However, placing comments is considered a good practice, and most web developers place comment code in HTML files to create self-documenting pages. Comments are placed as follows:

<!— comment placed here —>

3. Elements and tags
    a. The term “tag” and the term “element” refer to the markup used to write HTML. They are similar but not interchangeable. When a person writes

HTML, he or she is writing an HTML ***tag***. HTML files contain text content and specialized elements that specify formatting information.

b. An ***element*** is the basic building block of HTML and is typically made up of two tags: a start tag and an end tag accompanied by the "<" and ">" characters. The name of the element is placed between the <"and"> characters. The end tag contains the "/" symbol before the ">" character. Therefore, if there is a requirement to actually display the ">" and "<" characters, specialized techniques must be used. Most HTML elements are used with start and end tags. After the DOCTYPE declaration statement, HTML files are required to contain an html element's start tag. The html element is a two-sided tag. The last line of text in an HTML file is the </html> tag.

4. (See VM–R.) An HTML file is composed of two sections—a head and a body—identified by the head element and the body element. Both of these are two-sided elements and need start and end tags, which are <head>, </head>, <body>, and </body> tags.

   a. The ***head element*** is a two-sided element that contains child elements inside it, including the title element. The contents of the title element are displayed in the title bar of the browser window. The head element also contains a meta-tag that tells the browser about the language that the page is using. "UTF-8" indicates that the file is using plain English text.

   b. The ***body element*** is an element that follows the head tag and contains a description of the body of the webpage. VM–S shows the code in a simple HTML file and its rendering in a web browser. The body element contains a paragraph element inside it, which is displayed in the main area of the browser window.

E. Essential HTML elements

1. A ***heading element*** is a two-sided mechanism that has a start and an end tag. Headings are created in an HTML file using heading elements. VM–T shows the six heading elements used to set up headings in the webpage. The elements do not specify font names or size. Heading element 1 will render in larger font than heading element 2, which, in turn, is larger than heading 3, and so on. By default, text is rendered in black color. Each heading is placed in a separate line.

2. The ***paragraph element*** is an element coded using the and tags. Each paragraph begins on a new line. Adding extra spaces and hitting the Enter key does not create paragraphs because HTML processors do not honor spaces. Only the paragraph element can create paragraphs. A paragraph auto sizes itself when the browser window is resized. See VM–U. [NOTE: Even though the first paragraph was coded in multiple lines, it is shown as a single paragraph.]

3. A ***list element*** is a mechanism to present information. Two types of lists can be created in HTML: ordered and unordered. The ordered list records numbered elements, and the unordered list displays information with a round dot.
   a. Ordered lists are created using the "ol" element and its tags: <ol> and .
   b. Unordered lists are created with the "ul" element and its tags: <ul> and .
   c. The "li" element is used to create list items for both lists. It is a two-sided tag, using the <li> and tags. The tags must be nested inside the list tags.
   d. VM–V shows an unordered list and an ordered list. It also includes comments to indicate the purpose of the code inside the file. Line breaks inserted in the HTML file are not honored when rendering the list. However, if the list element contents are too long to fit inside the browser window, it is auto wrapped, keeping the indent level of the list item.
4. A ***table element*** is a mechanism to arrange data—text, preformatted text, images, links, forms, form fields, and other tables—into rows and columns of cells. Tables are created in HTML using three elements: <table>, <tr>, and <td> elements. They set up a table, a table row, and a table data (or table "cell") element. All of these elements are two-sided and need the start and end tag. Nesting is important when creating tables.
   a. VM–W shows the structure of the table element. The tr elements are nested inside the table element, and each tr element represents a row in the table. Within the tr element are td elements. Each td element represents a single cell in a single row of the table. A td element must be nested completely inside a tr element.
   b. See VM–X. Code for the table element with the tr and td elements inside it are shown, along with the rendering of the table in the browser. Browsers render table elements without a border.

Teaching Strategy: *Many techniques can be used to help students master this objective. Use VM–M through VM–X to review. Assign LS–A and LS–B.*

Objective 4: Create valid, well-indented code.

Anticipated Problem: What coding standards are followed when creating webpages?

IV. Create valid, well-indented code

A. Validate HTML documents

1. HTML is a free-format markup language. Elements in an HTML file do not have to be placed at any special level of indenting, and spaces are not needed between an element's end tag and the next element's start tag. As long as elements are correctly named, the start and end tags match up, and the proper nesting principles are followed, a page will be rendered correctly in a

browser window. [NOTE: Even when proper nesting principles are not followed, browsers will render pages to the extent possible.]

2. For validating HTML files, refer to VM–Y. It shows an HTML file with some missing tags that are commented out and highlighted. Also shown is the rendering of the page in a browser.
   a. Even though the and </body> tags were missing, the page was rendered correctly in the browser. In fact, Google's programming standards indicate that the <html>, <head>, <body>, and <li> tags may be omitted and will be inferred. However, this may not be true for all browsers, and not all tag-related errors can be handled correctly. Therefore, code must be validated before being deployed.
   b. The W3C validator web service validates code. VM–Z shows a file with improper table tags rendered correctly in a browser window. When sent to W3C's validator web service, errors were detected and displayed. The W3C validator web service is provided at the following URL: https://validator.w3.org/.

B. Create code with proper indents
   1. An HTML file is a free-format file, so it can be written without any particular style. However, it does make it difficult to debug such files. Implementing good coding practices makes an HTML file more readable and easier to maintain, even if it does not add to its functionality.
      a. A commonly followed coding standard is to use indents (by pressing the "tab" key) to show the nesting of elements within a file.
      b. In VM–AA, an HTML file is shown with and without proper indentation of elements. Indenting makes it easier to see the hierarchy of elements in the file.
   2. Using Firefox to format HTML files
      a. In VM–BB, the file is valid, and elements are placed with appropriate start and end tags. However, the file is not properly indented.
      b. Firefox has an add-on called "View Source Chart" that can be downloaded and added to Firefox at https://addons.mozilla.org/en-US/firefox/addon/view-source-chart/. After this add-on has been installed, a person can right-click the page display in Firefox and select "View Source Chart." The layout of the file elements is displayed, with nesting shown in nested boxes.
   3. NotePad++ can be used to format HTML files.
      a. NotePad++ can be used to indent code.
      b. To indent elements in an HTML file, a person can install the XML plug-in and select the "Pretty-print" option.

Teaching Strategy: *Many techniques can be used to help students master this objective. Use VM–Y through VM–BB to review. Assign LS–C.*

Objective 5: Use open-source software.

Anticipated Problem: What is open-source software, and how is it used?

V. Open-source software

A. Open-source vs. free software

1. Many free software applications exist in the market today (e.g., NotePad++, Google Chrome, and Firefox). These may be used for perpetuity and are not disabled after a certain amount of time.
   a. Many free applications downloaded from the web are provided in machine code format. The source code for the application, however, is not available. This means that a person cannot make changes to the software provided. Only the company that owns the source code for the product can make changes. Periodically, new versions (of the free software) are released, where changes are incorporated into the source code and the program is recompiled to create new machine code. These applications are not deemed (considered) to be open-source software.
   b. ***Open source software*** is software whose source code is available for modification by anyone. Open source software provides users with source code and machine code. Users may make changes to the source code and recompile it to create new machine code. The Linux operating system is distributed in source code and machine code format and is an example of open source software.

B. Versioning software

1. In the "real" world, program development is completed in phases, with versions of the program maintained in various stages of completion. Keeping track of versions of software is more than a programming issue. It is a software management issue. Additionally, when multiple developers are working on different parts of an application, each team member needs to be kept abreast of the versions of individual portions of the application.
2. Version Control Software (***VCS***) or Source Code Management (***SCM***) applications are used to implement version control procedures in an application development process. Currently, two popular software applications help with version control: Subversion (SVN) and Git.

C. SVN and Git

1. Subversion or ***SVN*** is a free/open-source version control system. SVN manages files and directories and manages changes a person makes to those files and directories. It also operates across networks and is able to foster collaboration.
2. ***Git*** is open-source software that performs version control. It is the most widely used SCM software and was originally developed for tracking changes made to

the Linux Operating system. When using Git, files that need to be versioned are maintained in a repository, which is a special type of folder controlled by Git.

a. Within the Git-controlled folder/repository, each finalized change is referred to as a commit action. Git maintains a log of each commit action made to each file in the repository.
b. Various programs provide Git functionality on a PC. These include GitHub for Windows, Git GUI, and TortoiseGit. Each one of these has different techniques for maintaining a repository and tracking changes made to individual files.

3. GitHub
   a. Git is a distributed VCS that maintains a local repository and a remote repository on a web server. Multiple users are allowed to make changes to the remote repository. Each person who makes a change is required to have a local repository as well. Even though Git supports distributed repositories, an individual user can maintain versions of applications on the local PC.
   b. GitHub is a company that hosts remote repositories. A user running Git on a PC can connect to http://www.GitHub.com and push local repositories to http://www.GitHub.com so other users in the project can utilize the latest version in their application. Also, http://www.BitBucket.com is a company that provides remote hosting of Git repositories.

Teaching Strategy: *Many techniques can be used to help students master this objective. Have students create a GitHub account and improve an online code file. Students may work in pairs to improve a posted file.*

Review/Summary.

Use the student learning objectives to summarize the lesson. Have students explain the content associated with each objective. Student responses can be used in determining which objectives need to be reviewed or taught from a different angle. If a textbook is being used, questions at the ends of chapters may be included in the Review/Summary.

Application.

Use the included visual master(s) and lab sheet(s) to apply the information presented in the lesson.

Evaluation.

Evaluation should focus on student achievement of the objectives for the lesson. Various techniques can be used, such as student performance on the application activities. A sample written test is provided.

Answers to Sample Test:

Part One: Completion

1. binary code files
2. compiler

3. markup
4. index
5. p AND/OR , AND/OR
6. tr AND/OR <tr> AND/OR </tr>

Part Two: True/False

1. F
2. F
3. T
4. T
5. T
6. F

Part Three: Short Answer

1. The main purpose of a compiler is that it translates/converts source code into machine/binary code and saves the machine/binary code file.
2. The first statement in a HTML5 file is: <!DOCTYPE html>
3. Git is used to maintain different versions of files.

Sample Test

Name _____

Basic Markup

Part One: Completion

Instructions: **Provide the word or words to complete the following statements.**

1. Machine code file are also referred to as _____.
2. A special program that reads a source code file and translates the instructions coded inside it into machine code is a/an _____.
3. A language that combines content and formatting information in a single file is _____.
4. A file considered the "root" of a website is a/an _____.
5. The element used to set up a paragraph is _____.
6. The element used to create a single row in a table is _____.

Part Two: True/False

Instructions: **Write *T* for true or *F* for false.**

_____1. Source code can execute on a computer.

_____2. Interpreters save machine code files.

_____3. Scripts are interpreted and not compiled.

_____4. HTML files need to be saved with a specific extension.

_____5. Google standards indicate that some elements may be coded without end tags.

_____6. Lack of proper indentation in an HTML file cause problems when displayed in a browser.

Part Three: Short Answer

Instructions: **Answer the following.**

1. What is the main purpose of a compiler?

2. Write the first statement in a HTML5 file.

3. Why is Git used?

COMPUTER LANGUAGE COMPARISON

COMPUTER LANGUAGE	INSTRUCTION TO DISPLAY "HELLO"
C++	`cout << "Hello" << endl;`
JAVA	`System.out.println("Hello");`
PYTHON	`print("Hello")`
JAVASCRIPT	`alert("Hello")`
PHP	`echo "Hello\n";`

COMPILATION PROCESS

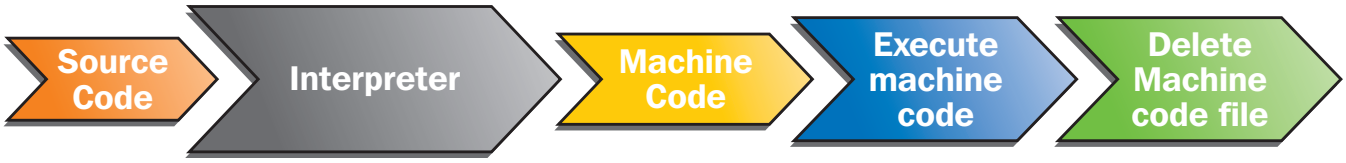
Save Source Code File

Save Machine Code File

COMPILED LANGUAGES AND USES

Compiled Language	Uses
C++	High performance applications
C#.NET	Web applications and games and database applications
Visual Basic.NET	General purpose language for web applications, GUI applications, and database applications
Java	General purpose language for web applications, GUI applications, and database applications

HOW THE INTERPRETER WORKS



COMMON SCRIPTING LANGUAGES

Scripting Language	Uses
JavaScript	Make webpages more interactive
PHP	Create web applications
Python	General purpose language

SCRIPTING VS. COMPILED LANGUAGE

	Scripting Language	Compiled Language
Advantages	• It is easy to maintain, as only source code file is needed. • Scripts are easier to test.	• Execution is fast once the program has been compiled and machine code has been generated. • The program can be distributed in machine code format, making it *harder to copy*.
Disadvantages	• It has slower execution, as machine code must be generated for each execution. • It cannot be distributed in machine code format, as machine code is not saved. This means that code *can* be copied.	• It is hard to maintain, as there are two files to manage—source code file and machine code file.

PROGRAMMING LANGUAGES VS. MARKUP LANGUAGES

	Computer Language	Markup Language
Implement logic structures	Yes	No
Generate data	Yes	No
Format data	No	Yes
Aggregate data	Yes	No

CLI VS. GUI APPLICATIONS

Command Line Interface (CLI)

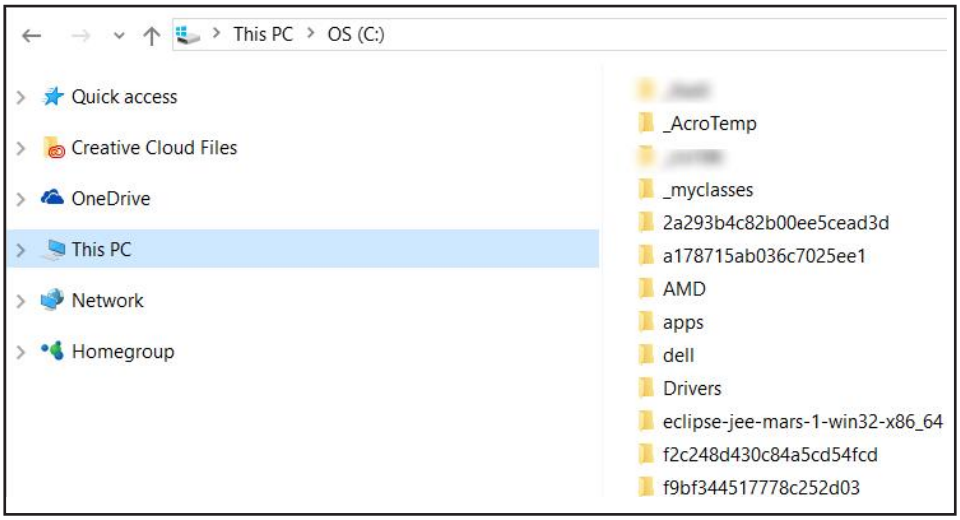
```
C:\>dir
 Volume in drive C is OS
 Volume Serial Number is D2FD-F6C6

 Directory of C:\

12/15/2013  04:01 AM    <DIR>          2a293b4c82b00ee5cead3d
10/11/2013  03:01 AM    <DIR>          a178715ab036c7025ee1
12/07/2015  02:12 PM    <DIR>          eclipse-jee-mars-1-win32-x86_64
09/06/2015  08:15 AM    <DIR>          personal
02/14/2016  01:22 PM    <DIR>          Program Files
02/20/2016  03:01 PM    <DIR>          Program Files (x86)
04/04/2015  01:55 PM    <DIR>          Python34
08/13/2013  01:05 PM    <DIR>          Temp
12/20/2015  05:11 AM    <DIR>          Users
11/07/2007  08:09 AM         1,442,522 VC_RED.cab
11/07/2007  08:12 AM           232,960 VC_RED.MSI
01/05/2016  05:36 PM    <DIR>          wamp
02/14/2016  04:00 AM    <DIR>          Windows
              26 File(s)      3,176,602 bytes
              32 Dir(s)  803,691,581,440 bytes free

C:\>
```

Graphical User Interface (GUI)



(Microsoft product screen shot(s) reprinted with permission from Microsoft Corporation.)

FILE EXTENSION DISPLAY

Display with File Extensions Visible	Display with File Extensions NOT Visible
Acme_Industries.accdb	Acme_Industries
Final Grades.xlsx	Final Grades
Fundamentals of HTML.docx	Fundamentals of HTML
My First Web Page.html	My First Web Page
text_document.txt	text_document
Wonders of the World.pptx	Wonders of the World

(Microsoft product screen shot(s) reprinted with permission from Microsoft Corporation.)

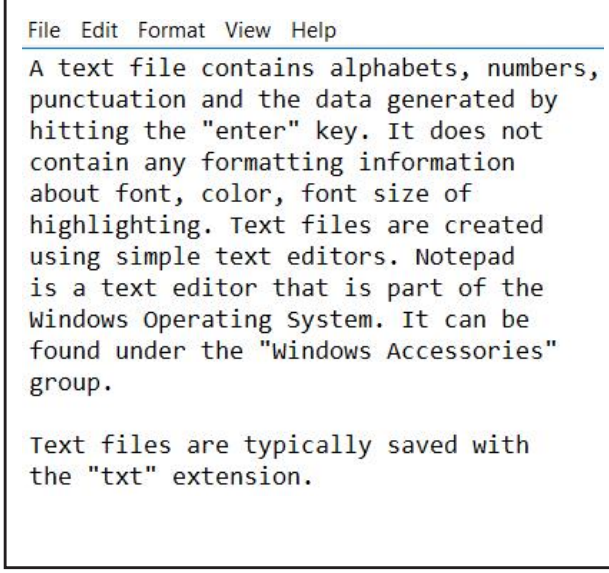
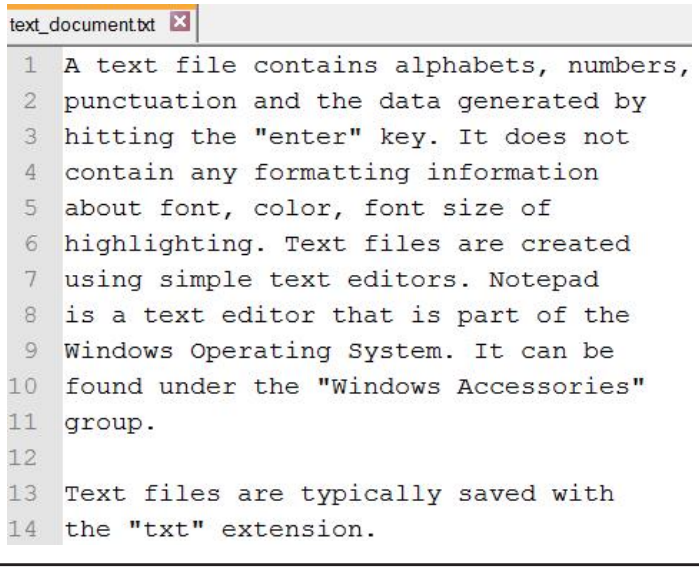
FILE EXTENSIONS AND REGISTERED APPLICATIONS

File Extension	Application to which the File Extension is Registered	Icon Displayed Across File
.docx	Microsoft Word	
.doc	Microsoft Office 1997 to 2003	
.pptx	Microsoft PowerPoint	
.html	HTML files are registered to one of the following web browser products: • Google Chrome • Firefox • Microsoft Edge	 (Microsoft Edge icon)
.txt	TXT files are registered to one of the following web browser products: • Notepad • NotePad++	 Notepad icon  NotePad++ icon

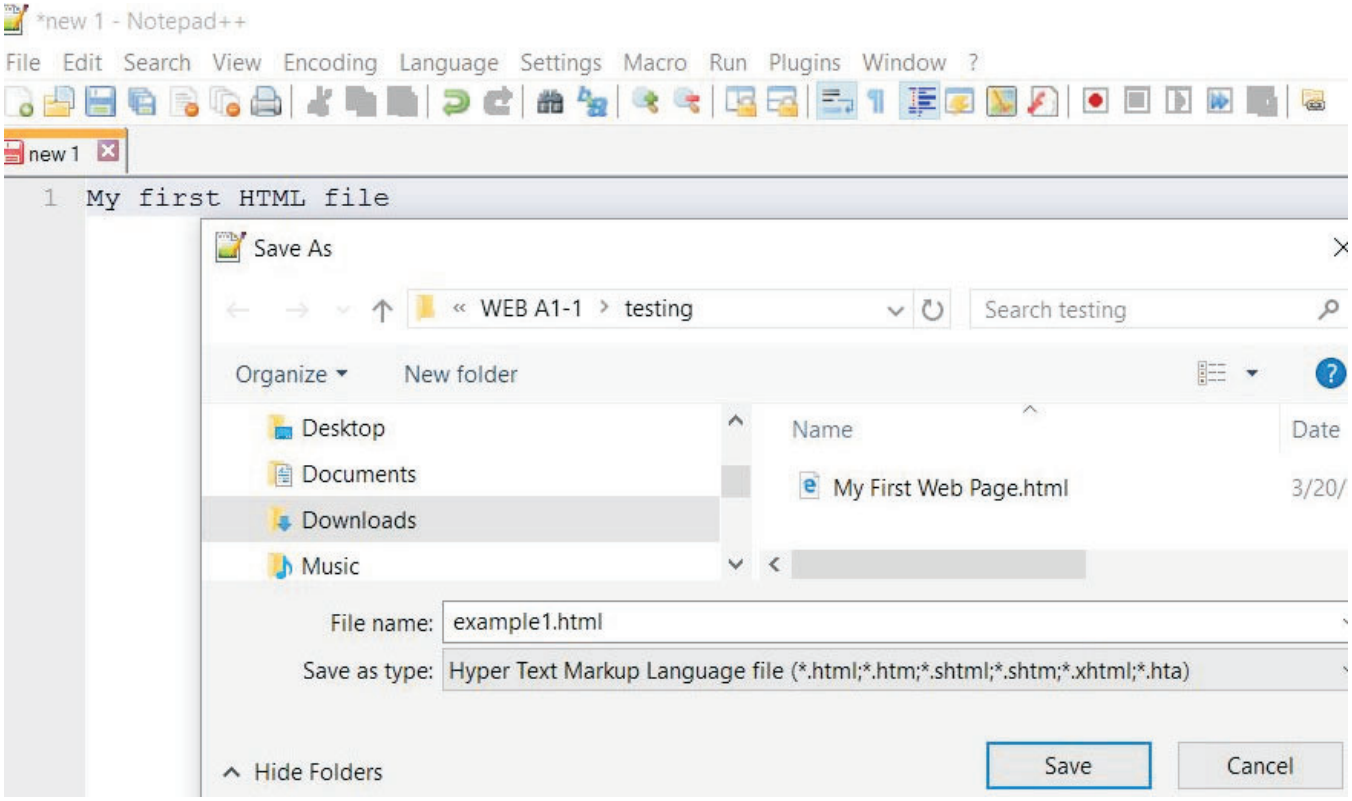
(Microsoft product icon(s) reprinted with permission from Microsoft Corporation.)

(Notepad++ icon(s) reproduced under the GNU General Public License. Copyright Don Ho 2016)

NOTEPAD VS. NOTEPAD++

Display in NotePad	Display in NotePad++
	

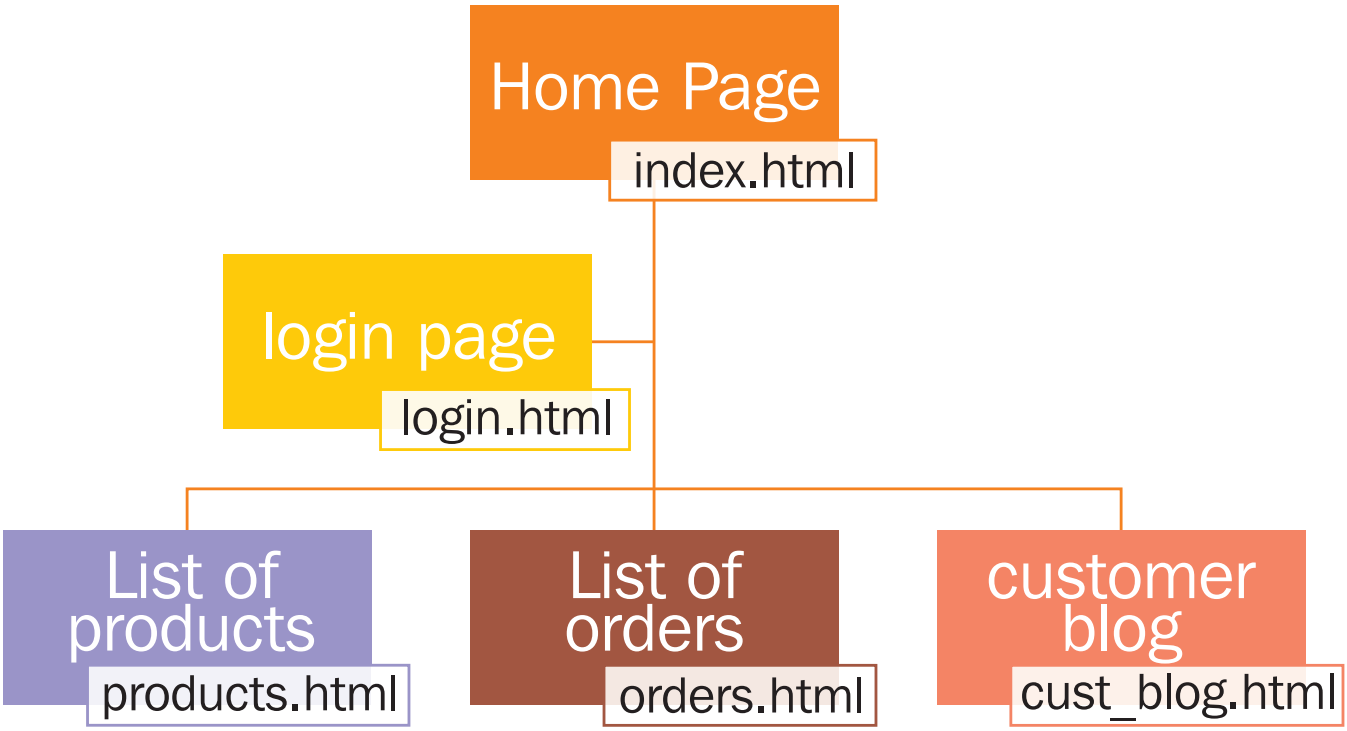
HOW TO SAVE HTML FILES IN NOTEPAD++



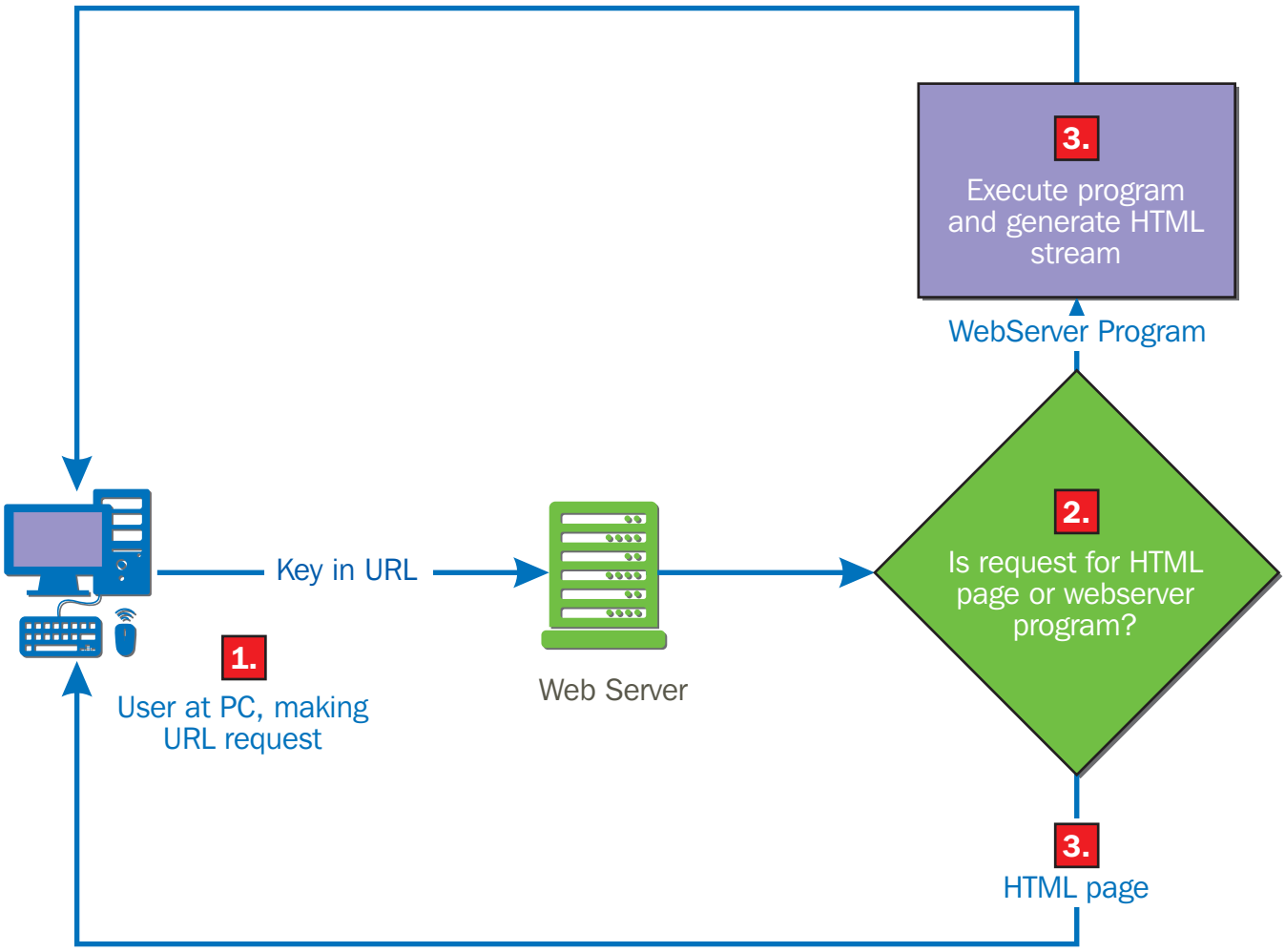
STORYBOARDING SOFTWARE

Software / Website Name	Use
Microsoft Office 2013—PowerPoint	Insert organization chart SmartArt graphic
http://creately.com/	Create website layouts online to share with others
https://www.gliffy.com/	Create free website layouts and save on the web

STORYBOARDING WEBSITE LAYOUT



HOW DO WEB REQUESTS WORK?



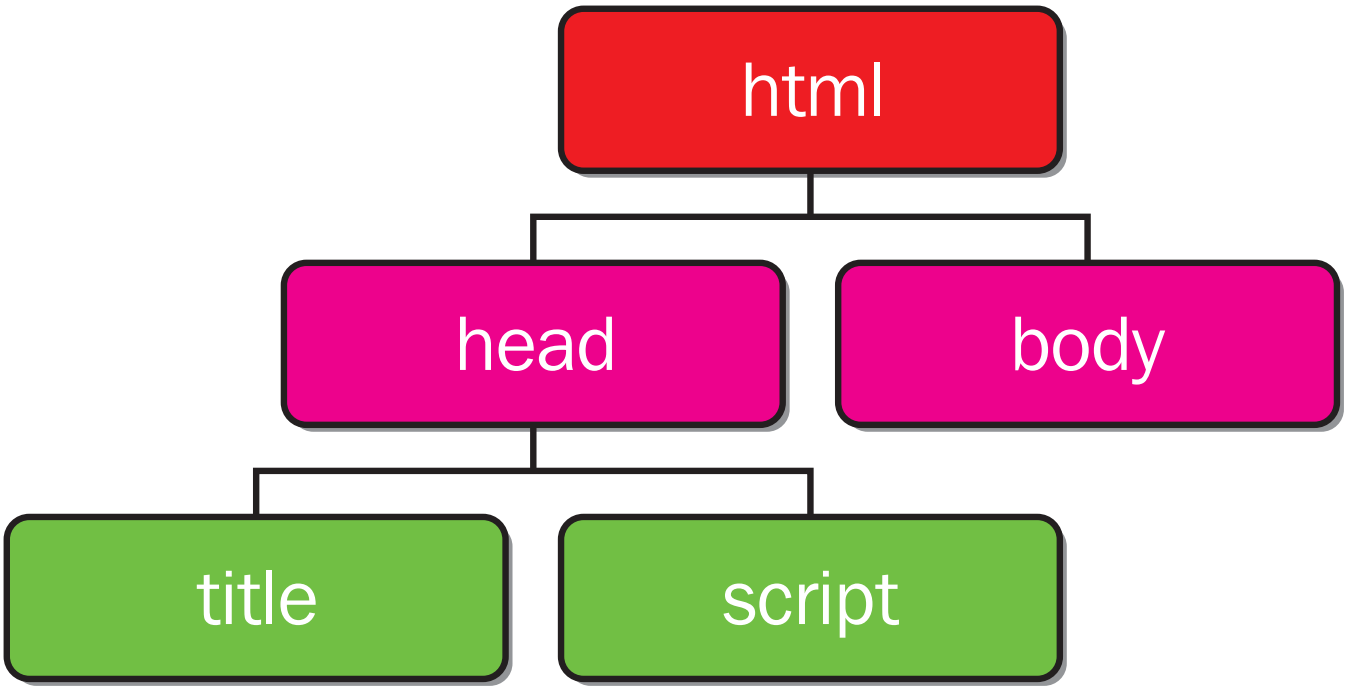
WEB SERVER PROGRAMMING LANGUAGES

File Extension	Language
.aspx	Web server program written in Visual Basic.NET or C#.NET
.jsp	Web server program written in Java
.php	Web server program written in PHP

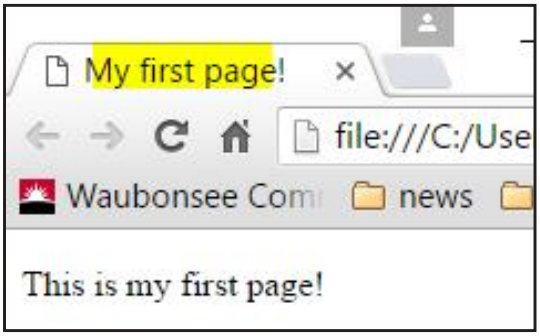
HTML VERSIONS

Version	Date
HTML 2.0	1995
HTML 3.2	1997
HTML 4.0	1998
HTML 4.01	1999
XHTML	2000
HTML5	2014

HTML FILE PARTS



A SIMPLE HTML FILE

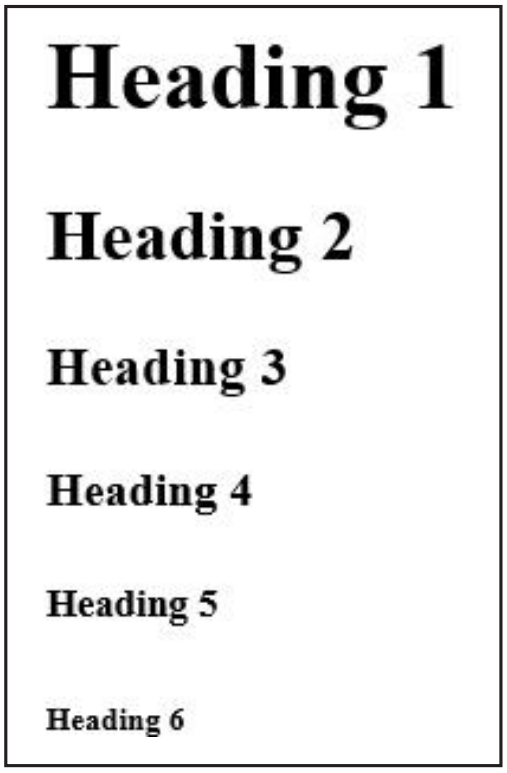
HTML Code	Browser Window Display
<!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>My first page!</title> </head> <body> This is my first page! </body> </html>	

HTML HEADING ELEMENTS

HTML Element	Description
	Heading 1
<h2> <h2>	Heading 2
	Heading 3
<h4> <h4>	Heading 4
	Heading 5
<h6> <h6>	Heading 6

HTML Code	Browser Display

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="UTF-8">
      <title>My page!</title>
   </head>
   <body>
      <h1>Heading 1</h2>
      <h2>Heading 2</h2>
      <h3>Heading 3</h3>
      <h4>Heading 4</h4>
      <h5>Heading 5</h5>
      <h6>Heading 6</h6>
   </body>
</html>
```



THE PARAGRAPH ELEMENT

HTML Code	Browser Display
<!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>My page!</title> </head> <body> This is a sample paragraph. It goes into many lines . This is a second paragraph! </body> </html>	This is a sample paragraph. It goes into many lines . This is a second paragraph!

LIST ELEMENTS

HTML Code	Browser Display

```
<!DOCTYPE html>
<html>
   <head>
    <meta charset="UTF-8">
      <title>My page</title>
   </head>
   <body>
      <!-- unordered list -->
      <ul>
         <li>Fall. Fall is my favorite season
         of the year and I love the
         brown gold colors of leaves
         in this season.</li>
         <li>Winter</li>
      </ul>
      <!-- ordered/numbered list -->
      <ol>
         <li>Spring</li>
         <li>Summer</li>
      </ol>
   </body>
</html>
```

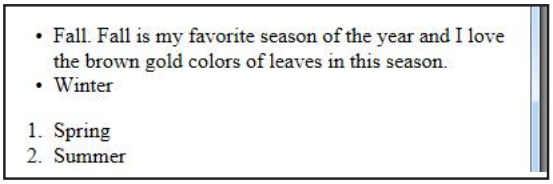


TABLE ELEMENTS

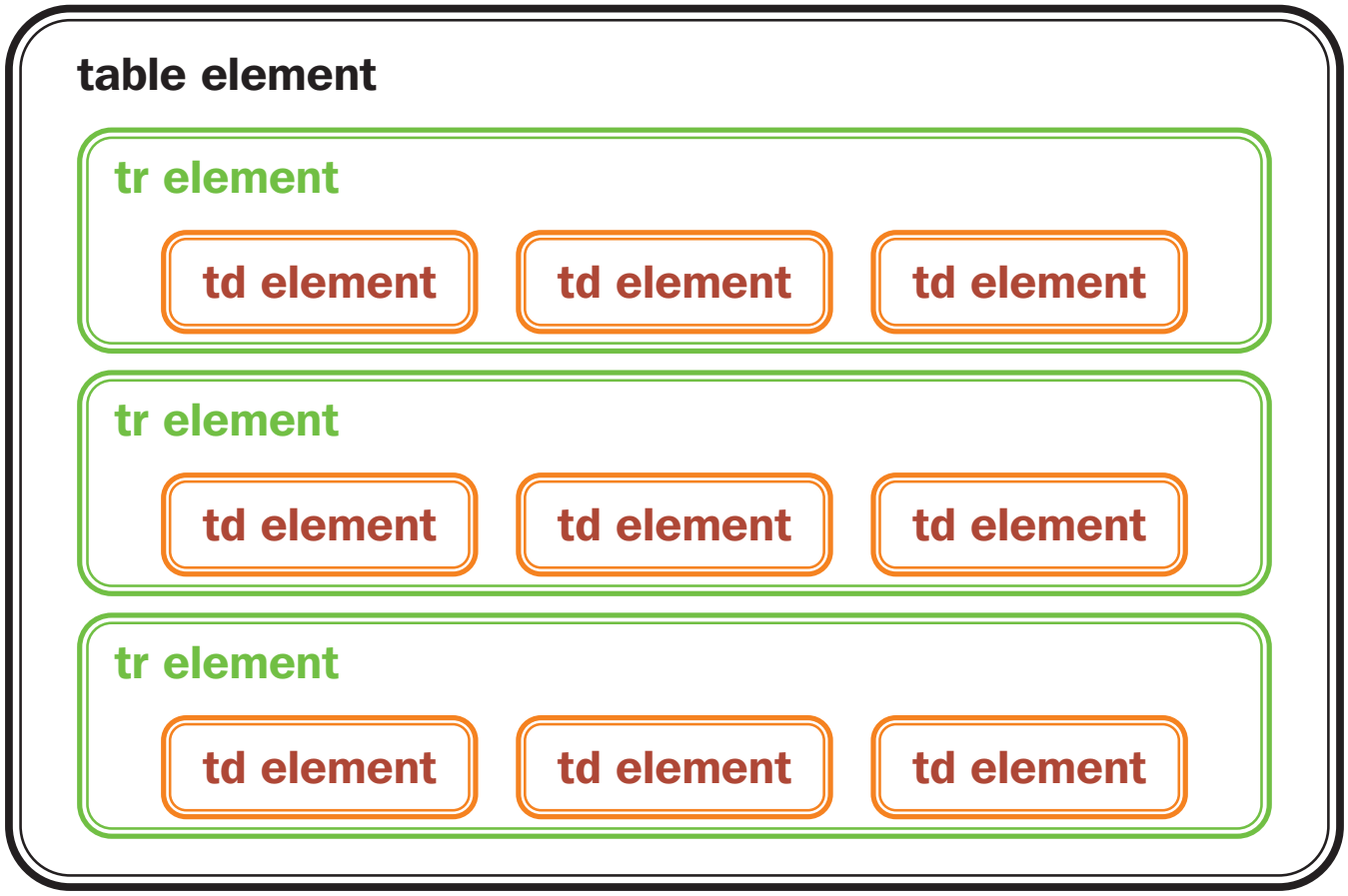


TABLE ELEMENTS IN THE BROWSER

HTML Code	Browser Display

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>My page</title>
  </head>
  <body>
    <table>
      <tr>
        <td>Fall</td>
        <td>Spring</td>
        <td>Summer</td>
      </tr>
      <tr>
        <td>Brown</td>
        <td>Yellow</td>
        <td>Blue</td>
      </tr>
      <tr>
        <td>Gold</td>
        <td>Light green</td>
        <td>Pink</td>
      </tr>
    </table>
  </body>
</html>
```

Fall	Spring	Summer
Brown	Yellow	Blue
Gold	Light green	Pink

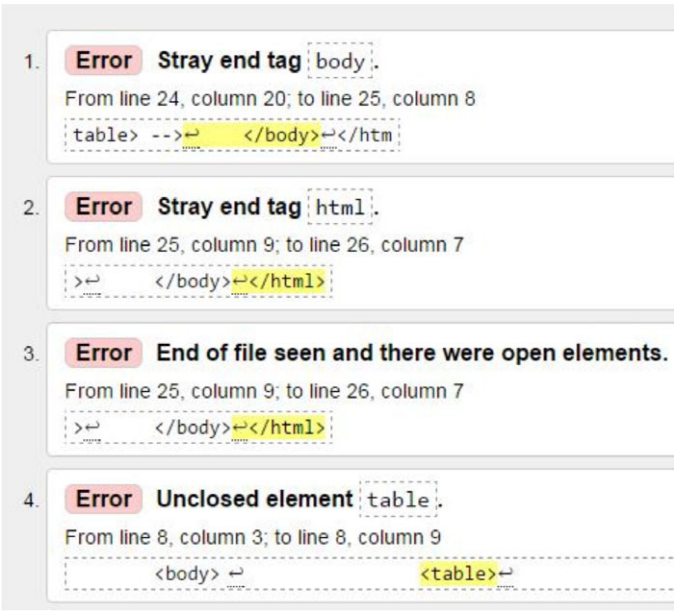
HTML FILE WITH MISSING TAGS

HTML Code	Browser Display
<!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>My page</title> </head> <body> This is a sample paragraph. It goes into many lines. <!-- --> This is a second paragraph! <!-- </body> --> </html>	This is a sample paragraph. It goes into many lines. This is a second paragraph!

TABLE TAG ERRORS

HTML Code	Display in https://validator.w3.org/

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="UTF-8">
      <title>My page</title>
   </head>
   <body>
      <table>
         <tr>
            <td>Fall</td>
            <td>Spring</td>
            <td>Summer</td>
         </tr>
         <tr>
            <td>Brown</td>
            <td>Yellow</td>
            <td>Blue</td>
         <!-- </tr> -->
         <tr>
            <td>Gold</td>
            <td>Light green<!--</td> -->
            <td>Pink</td>
         </tr>
      <!-- </table> -->
   </body>
</html>
```



CODING STYLES: WITH AND WITHOUT INDENTATION

Coding Styles

HTML Code WITHOUT Indents	HTML Code WITH Indents

```
<!DOCTYPE html>
<html>
<head><meta
charset="UTF-8"><title>My page
</title>
</head>
<body><table>
<tr><td>Fall</td><td>Spring</td>
<td>
Summer
</td></tr><tr><td>Brown</td>
<td>Yellow</td><td>Blue</td></tr>
<tr>
<td>Gold</td><td>Light green</td>
<td>
Pink
</td>
</tr>
</table>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="UTF-8">
      <title>My page</title>
   </head>
   <body>
      <table>
         <tr>
            <td>Fall</td>
            <td>Spring</td>
            <td>Summer</td>
         </tr>
         <tr>
            <td>Brown</td>
            <td>Yellow</td>
            <td>Blue</td>
         </tr>
         <tr>
            <td>Gold</td>
            <td>Light green</td>
            <td>Pink</td>
         </tr>
      </table>
   </body>
</html>
```

CODING STYLES: VIEW CHART SOURCE DISPLAY

Coding Styles

HTML Code WITHOUT Indents	View Chart Source Display

```
<!DOCTYPE html>
<html><head><meta charset="UTF-8">
<title>
My page
</title>
   </head>
<body>
<table><tr><td>Fall</td><td>
Spring
</td>
<td>Summer</td></tr><tr>
<td>Brown</td><td>
Yellow</td><td>Blue</td></tr><tr>
<td>Gold</td><td>
Light
green</td>
<td>Pink</td>
</tr>
</table></body>
</html>
```

```
<!DOCTYPE html>
<HTML>
  <head>
  <meta charset="UTF-8">
    <title>
    My page
    </title>
  </head>
  <body>
    <table>
    <tbody>
      <tr>
        <td>
        Fall
        </td>
        <td>
        Spring
        </td>
        <td>
        Summer
        </td>
      </tr>
      <tr>
        <td>
        Brown
        </td>
        <td>
        Yellow
        </td>
        <td>
        Blue
        </td>
      </tr>
```

LS–A

Name _____

Create a Simple HTML Introduction Page

Purpose

The purpose of this activity is to practice HTML code creation.

Objectives

1. Set up code to define an HTML page.
2. Create HTML elements in the file to display information.
3. Participate in a class discussion of the HTML introduction page code.

Materials

- lab sheet
- computer with a text editor
- browser software
- class notes
- paper
- writing utensil

Procedure

1. Review your class notes about HTML statements.
2. Read the following instructions. Then write the HTML statements that fulfill the instruction. (Practice on your paper first, as needed.) Save the file with an HTML extension.
    a. Write the HTML declaration statement as the first statement in the file and the start html tag on your paper.

b. Set up a head element with a title element and a meta-tag on your paper.
c. Write a heading line with a page heading.
d. Write a paragraph with an introduction.
e. Save the file by writing .html as the extension.

3. Participate in a class discussion of the HTML introduction page code.
4. Turn in your completed lab sheet to your instructor.

Create a Simple HTML Introduction Page

2a. <!DOCTYPE html>

<html>

2b. <head><title>My page</title><meta charset="UTF-8"></head>

2c. <h2>My first page</h2>

2d. This is a brief introduction to I, me, myself!

LS–B

Name _____

Work with Browser Software

Purpose

The purpose of this activity is to understand differences in browser software.

Objectives

1. Open a saved HTML file in two browser applications.
2. Write a paragraph explaining the differences in how the page was rendered.
3. Participate in a class discussion of the differences in the page renderings.

Materials

- lab sheet
- computer with multiple browser applications (e.g., Google Chrome, Firefox, and Internet Explorer)
- webpage created in LS–A
- paper
- writing utensil

Procedure

1. Open the saved HTML file created in LS–A in a minimum of two browser applications.
2. Write a paragraph explaining the differences in how the page was rendered using various types of browser software.
3. Participate in a class discussion of the differences in the page renderings.

Work with Browser Software

Pages will look different in different browser software applications.

Name _____

Validate HTML Files

Purpose

The purpose of this activity is to practice using webpage validation tools.

Objectives

1. Create a webpage with a table.
2. Omit the <tr> and </table> tags.
3. Save the file and view in browser window.
4. Validate the HTML file using the W3C validator.

Materials

- lab sheet
- class notes
- computer with browser software and text editor (e.g., NotePad or NotePad++)

Procedure

1. Review your class notes about validating HTML files.
2. Read the following instructions, and write the HTML statement that fulfills each instruction.
    a. Create an HTML file with a table. The table should contain two rows, and each row should contain three columns. Each cell holds the name of a person.
    b. When coding the table, omit the <tr> tag for the first row and the </table> tag.
    c. Save the file with the .html extension, and validate it at the https://validator.w3.org/ site.
    d. Document the errors in the file.
    e. Fix the errors, and revalidate the file.
3. Participate in a class discussion of validating HTML files.
4. Turn in your completed lab sheet to your instructor.

Validate HTML Files

2d. The incorrect code is as follows:

```
<!DOCTYPE html>
<html>
        <head>
        <meta charset="utf-8">
        <title>Test</title>
    </head>
    <body>
        <table>

                <td>Tim</td>
                <td>John</td>
                <td>Jonathan</td>
            </tr>
            <tr>
                <td>Jane</td>
                <td>Jaimie</td>
                <td>Karla</td>
            </tr>

    </body>
</html>
```

2e. The correct code is as follows:

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Test</title>
    </head>
    <body>
        <table>
            <tr>
                <td>Tim</td>
```

```
            <td>John</td>
            <td>Jonathan</td>
        </tr>
        <tr>
            <td>Jane</td>
            <td>Jaimie</td>
            <td>Karla</td>
        </tr>
    </table>
</body>
</html>
```