Git

Used to back up and share code, and to document the development process.

Getting files from Git

- Fork files to your repository (git website), record URL
- Clone to desktop
  `git clone URL`
- Use and update files
- Send files back to Git
  - Stage for committing (creates a list of files needing to be updated)
    `git add --all` *adds all files in folder to the list*
    `git status` *shows you what has been changed*
  - Commit to save snapshot of files
    `git commit –m "message"` *saves a snapshot to local computer, stores with the message entered*
  - Push to the cloud
    `git push origin master` *sends commit to the repository in the cloud, prompts for git username and password*
  - Pull request to merge with original document (for collaborative projects)

Python

Set up local http server. From the command line, navigate to the appropriate folder and then type:

cd *change directory – use this when you need to type in your file path, as shown below*

cd c:\users\erica\desktop\VizTechFall2017\docs

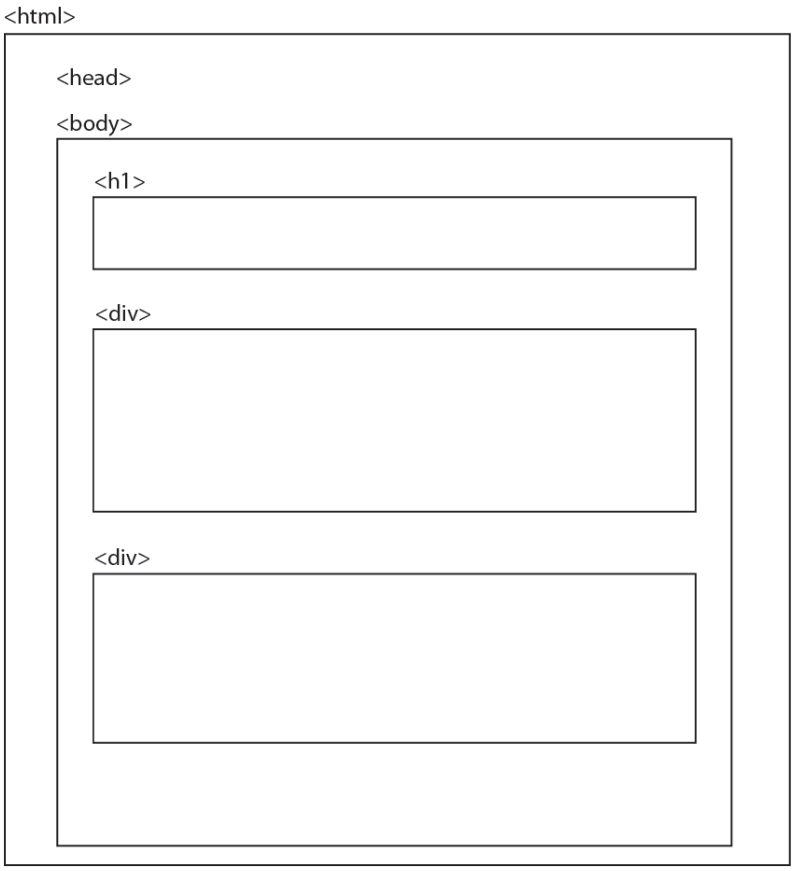
`python –m http.server` (`SimpleHTTPServer for a Mac`)

`Ctrl + C` *to break communication at the end of the session*

Use address localhost:8000 in the browser to view locally served webpage. (May need to replace 8000 with the number printed in the command line after the server initiates.)

HTML

Use tags to build site structure. Elements that end up inside other elements will have their tags nested inside those of the outer element (the h1 and div tags will show up inside of the body tag, and the body and head tags will show up inside of the html tag)



General tag format:

```
<tag> things that describe tag </tag>
```

```
<tag/>
```
use a self-terminating version when the tag does not need descriptors

To begin, declare a document of type html

```
<! doctype HTML>
```

You can add comments to explain what the code does

```
<!—comment -->
```
nothing inside the comment will be run as code

Common tags:

`<html>` all of the tags that define the page structure go in here `</html>`

`<head>` links to files and settings needed to run the page go in here `</head>`

Things that you might include in <head>:

`<meta charset = "utf-8">` *sets the kind of characters used on the page*

`<title>` title goes here `</title>` *creates the title text that shows up at the top of the tab or browser window*

`<link href='URL' rel="relation to your webpage">` *links to a file stored online – href means hypertext reference. Rel defines the relationship of the reference to your webpage - often "stylesheet." For font definitions, also include type="text/css"*

`<link href="style.css" rel="stylesheet"/>` *loads css stylesheet called style.css, which is stored in the parent directory for the webpage, and sets it as the stylesheet for the webpage*

Links we used this semester:

```
<link
href='https://fonts.googleapis.com/css?family=Droid+Serif:400,400italic'
rel='stylesheet' type='text/css'>

<link href='../bootstrap.min.css' rel="stylesheet"/>

<link href="style.css" rel="stylesheet"/>

<link
href='https://fonts.googleapis.com/css?family=Droid+Serif:400,400italic'
rel='stylesheet' type='text/css'>

 <link href="../bower_components/bootstrap/dist/css/bootstrap.min.css"
rel="stylesheet"/>
```

`<body>` contains all of the page elements *title, text blocks, lists, etc* `</body>`

Things that you might include in <body>:

Header – contains text, pictures, or information about the page that is separate from the main content. Elements are nested inside header using tags that depend on the kind of information and the style/formatting.

`<header>`

<h1> text that shows up at the top of the page </h1> *sets the text style*

</header>

Divs – separate content of the page into sections. Divs are also often nested. Each div can also be given a class, which allows you to access them separately. Class names cannot have spaces. Each item can have more than one class, separated by a space (see lists, below).

Headings – text, displayed with different styles defined for each heading level. Can have multiple levels, and styles are set in the CSS document.

<h1>text goes here</h1>

Lists – create lists with special formatting (different linespacing, bullets, etc.). Can also assign classes to lists, and individual list items can be given an id, which allows you to call an individual item rather than a whole class of items.

<ul class="optional-class-definition secondary-class"> *creates an "unordered list" with no number ordering*

<li class="class" id="id"> list item text goes here

Spans allow you to change the formatting of a specific word or section of text without changing anything around it. If you wanted to turn one word blue, you could use a span, and set the style for that span class to blue in the CSS document. Using the example above,

<li class="class" id="id"> list item with one word in blue *note that the word blue inside the span is the one that will be highlighted, using the class "blue" to set the style in the CSS document*

Embedding scripts in your webpage allows you to run Javascript functions in your website. This is necessary for creating any interactive elements. The code for those elements will be written in a Javascript file called script.js, which is loaded into the HTML document using a script tag.

<script src="filename"></script> *filename here can be either a file stored in the parent directory, or a webpage*

Scripts we used this semester:

<script src="script/script.js"></script>

```
<script src="http://d3js.org/d3.v3.min.js" charset="utf-8"></script>

<script src="script/lib/queue/queue.min.js"></script>

<script src="script/lib/jquery-1.11.3.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/d3/3.5.6/d3.min.js"
charset="utf-8"></script>
```

Hyperlinks are used to create a clickable link to another document or webpage (this is what most people think of as a "link" in the internet).

```
<a href="destination URL">link text that the reader sees</a>
```

Paragraphs are used to set paragraph formatting

```
<p>text goes here</p>
```

Breaks insert spaces in the document

```
<br>
```

Insert an image from a file stored in the site parent directory

```
<img src="image.png" alt="not sure what this does" width = "200", height
= "200">
```

An SVG element creates a drawing canvas for making shapes – more on this in the Javascript section. Pixels are the default unit for svg elements, but you can explicitly call em, pt, in, cm, mm. The origin is located at the upper left corner of the screen.

```
<svg width="50" height ="50">
```

SVG elements can be drawn and styled using HTML, but we will usually do this from within the Javascript and/or CSS files. Any SVG element can also be assigned a class, which can be used for CSS styling. There are no layers in SVG, so whatever element is called first, will be drawn first. Transparency can be achieved with RGB-A colors, or with the opacity attribute. The use of RGB-A can cause strange effects with color mixing, so it is often better to use opacity. You can also mix transparency and opacity to multiply their effects.

```
<circle cx="25" cy="25" r="22" fill="blue" stroke="gray"
stroke-width="1px"/>
```
cx and cy are the coordinates of the circle center, by default

```
<rect x="0" y="0" width="500" height="50" />
```
x and y give the coordinates of the upper left corner of the rectangle

```
<ellipse cx="250" cy="25" rx="100" ry="25" />

<line x1="0" y1="0" x2="500" y2="50" stroke="black"/>
```

```
<text x="250" y="25" font-family="serif" font-
size="25">text to display goes here</text>
```
x is the left edge of the text box, y is the text baseline, by default. Text will inherit CSS styling, unless otherwise specified.

HTML styling

Most of the time, it is better to set the style for HMTL objects in the CSS file. However, it is also possible to set the style in the HTML code. In many cases, this simply involves setting the style attribute using the CSS code enclosed in quotation marks.

```
<body style="background-color:lightgray">

<h1 style="color:purple">

<p title="test" style="color:green">

<p style="font-family:Times">
```

CSS

Uses HTML element tags, classes, and ids to set the style of objects that show up on the webpage. This includes setting all spacing between items, text style and size, and the background colors for the page.

Comments

```
/*  your comment here */
```

General CSS format:

In CSS, you make a selection of HTML elements using a selector, and then you assign values for different properties associated with that element. Property names are pre-defined, and values must be given in the correct unit and type (must be a string if the property expects a string, or a number if it expects a number)

```
selector {

      property-name: property-value;  (the colon and semicolon are
important here!)

}
```

Selectors call HTML objects based on tag, class, and id set in the HTML file, and select all elements with that particular tag, class, or id.

by tag: `tagname { put style definition here }`

by class: `.class { put style definition here }`

by id: `#id { put style definition here }`

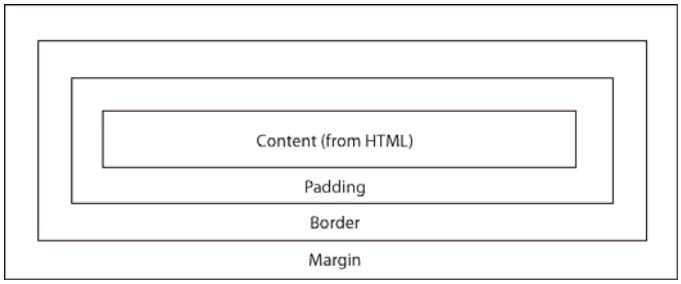
by class nested inside of a particular tag: `h1.class { put style definition here }` grabs objects of class "class" inside the h1 tag.

`.lists>.list` restricts inheritance to one level down – this is useful when a style is defined as a percentage and might be applied several times. Since the percentage is based on the containing box size, an element contained in a nested div might be scaled up or down more than desired. This command says that you apply the formatting to any class higher in the DOM than the .list class, but not to anything below it, so that items further down the hierarchy will not be scaled repeatedly.

CSS properties that you might want to change:

Object size and spacing

All CSS objects are boxes that contain the content to be displayed, some amount of padding or white space around the content, a border that may or may not be displayed, and a margin that defines more whitespace between the border and other things.



The content is determined by the HTML document, but the other attributes are set in the CSS. Each attribute is expecting a certain kind of data, in a specific order. Do not insert spaces between a number and its unit!

`width:200px;` *sets the width of the HTML element*

`height:200px;`

`height:100%;` *sets the height of the HTML element to be 100% of the parent element*

`padding: 20px;`

`border: 2px solid black;` *number in pixels, line style, line color*

`margin:20px;` *assumes a uniform 20 pixel margin all around*

`margin: 0;` *assumes default unit of px*

It is also important to tell an HTML element how it should relate to other elements on the page. Can use either absolute pixel values or percentages (better for responsive design). Percentages will be calculated based on the size of the containing box, not the overall screen size.

`position: absolute;` *defines a fixed position relative to the rest of the screen – useful for tooltips. Width has to be set manually*

`top:100px;` *define the distances for absolute placement*

`left:100px;`

`bottom:100px;`

`right:100px;`

`position: relative;` *stacks elements from top to bottom in the screen, unless otherwise specified*

`float: left;` *tells the element that it should float toward the top of the screen, and stack from left to right. This does allow elements to hide behind non-floating elements*

`clear:both;` *tells floating elements to pay attention to non-floating elements*

`margin-bottom:10px;`

`margin:top right;`

`margin:top right bottom left;`

`padding:top right bottom left;`

Setting color

There are several ways or representing color. The simplest way is to use the html color names, a set of built-in colors that will be recognized by your browser (http://www.w3schools.com/html/html_colornames.asp). Alternatively, you can use the `rgb(100,100,100)`, `rgba(100,100,100,.2)`, `hsl(0,100%,97%)`, or `hsv(0,15%,25%)` formats, or hexadecimal codes (#F0F8FF) to represent color values.

```
background:black;

fill:yellow;

fill:none;

stroke:black;

fill-opacity:.05;
```

Line styling

```
stroke-width:1px;
```

`stroke-dasharray:3px 3px;` *creates a dashed line with dashes of 3px and spaces of 3 px*

Font

```
font-size:12pt;
```

`font-size:100%;` *sets font size as a percent of the default size*

`font-family: Helvetica, Arial, sans-serif;` *sets preferred font family, an alternate class in case the preferred one is not available, and the style of type. If the font used is not a system font, use quotes for the name*

`font-size:16px;` *could also use 1.5em*

Formatting lists

`list-style:none;` *removes bullets from an unordered list*