

what is a server?

HTTPS

Hypertext Transfer Protocol Secure - is an extension of the Hypertext Transfer Protocol (HTTP). It is used for secure communication over a computer network, and is widely used on the Internet.

SSH

Secure Shell or Secure Socket - a network protocol that gives users, particularly system administrators, a secure way to access a computer over an unsecured network. **SSH** also refers to the suite of utilities that implement the **SSH** protocol.

Git

What is Git?

A version control system meant to make it easier to have multiple versions of code, sometimes across multiple developers or teams

At its most simple, git helps with the '**indexv1.html, indexv2.html, indexv3FINAL.html**' problem

At its most complex, git allows developers to work together worldwide on code without stepping on each other's toes

GitHub

Github is a service to host yr projects on the web.

Code is pushed (uploaded) from a local directory (folder) called a repository or rep.

example - our class site:

<http://www.github.com/rebleo/webDevSpring2020>

Git vs GitHub.com

git is a version control system that takes snapshots of your code at certain points in development

These snapshots are stored in a '**repo**', or '**repository**' on your local machine

GitHub.com is a website that hosts git repositories on a remote server + is available for all the web to see, copy + implement.

Git Terminology

repository - where data is managed. the directory containing your files.

local - the copy that exists on your machine, no one else can access this

remote - the copy in your github account, anyone with access to your github repo can access the remote instance (we won't be doing this!)

push - once you make changes to the local copy you *upload the changes to the remote copy

pull - if someone else makes changes to the remote copy (we won't be doing this this semester)

clone a repository - download the entire codebase of the repo you can pull in changes + and push your own changes if you are given access

Github pages

github.io

easily allows you to host web pages using github servers + workflow

url (uniform resource locator)

http://

yrUsername.github.io



yrUsername.github.io

< HTML >

Hypertext Markup Language

Describes the **content** + **structure** of a web page;
NOT a programming language

< HTML >

[box model]

block vs. **inline** display

The key to understanding how **HTML** + **CSS** works is to imagine that there is an invisible box around every **HTML** element.

Block level elements are outlined w/ red + inline elements in green.

<body> creates 1st box, then **<h1>**, **<h2>**, **<p>**, **<i>** + **<a>** each create their own boxes within it.

The Cottage Garden

The *cottage garden* is a distinct style of garden that uses an informal design, dense plantings, and a mixture of ornamental and edible plants.

The Cottage Garden originated in England and its history can be traced back for centuries, although they were re-invented in 1870's England, when stylized versions were formed as a reaction to the more structured and rigorously maintained English estate gardens.

The earliest cottage gardens were more practical than their modern descendants, with an emphasis on vegetables and herbs, along with some fruit trees.

The Cottage Garden

The *cottage garden* is a distinct style of garden that uses an informal design, dense plantings, and a mixture of ornamental and edible plants.

The Cottage Garden originated in England and its history can be traced back for centuries, although they were re-invented in 1870's England, when stylized versions were formed as a reaction to the more structured and rigorously maintained English estate gardens.

The earliest cottage gardens were more practical than their modern descendants, with an emphasis on vegetables and herbs, along with some fruit trees.

< HTML >

3 categories of HTML elements


1 - **block**: large blocks of content has height + width
<p>, <h1>, <blockquote>, , , <table>

2 - **inline**: small about of content, no height or width
**<a>, , ,
, , <time>**

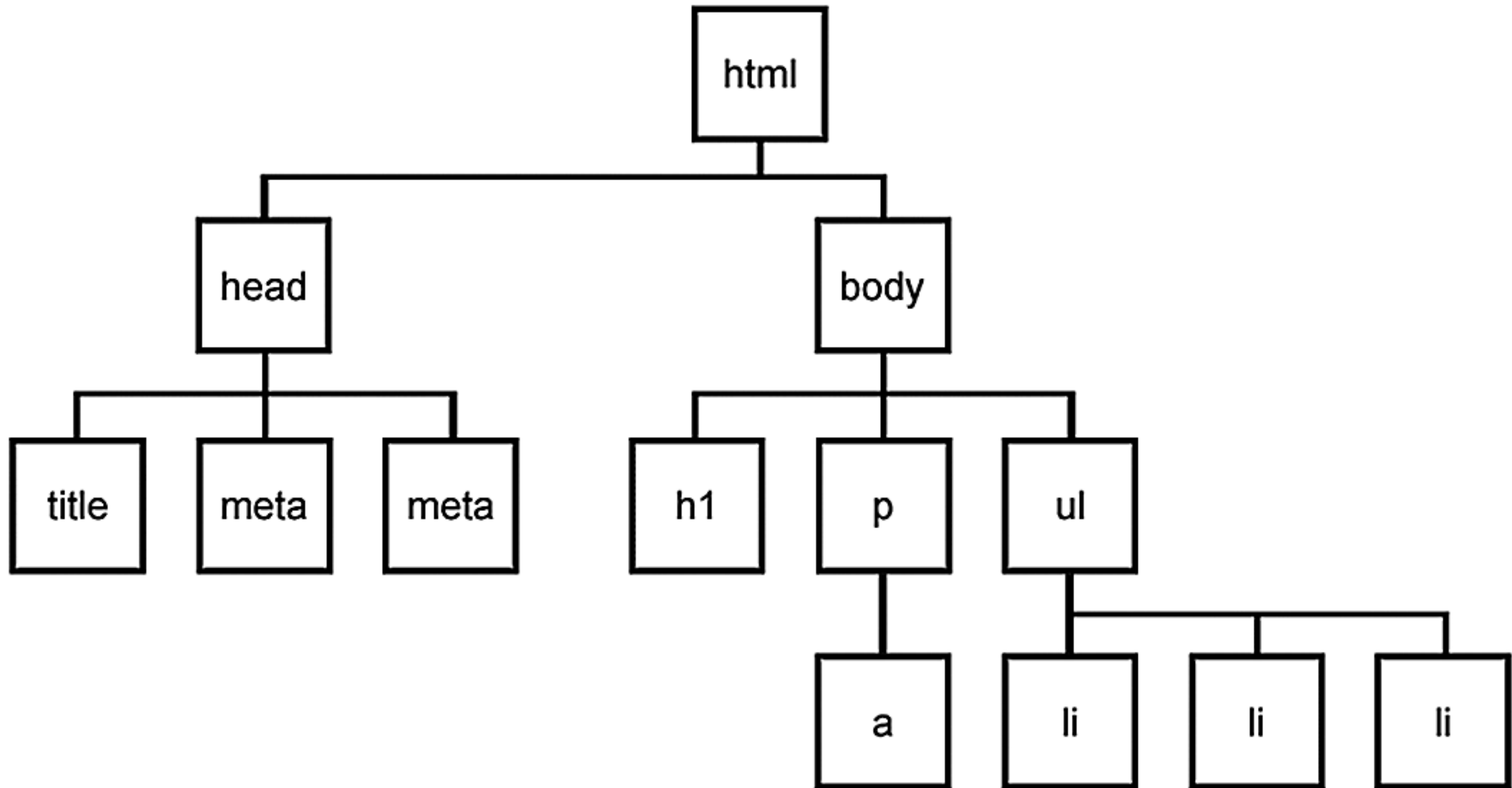
a. **inline block**: inline content w/ height + width

3 - **metadata**: information about the page, usually not visible
<title>, <meta>, <script>

HTML - Hyper Text Mark Up

```
<!DOCTYPE html>
<html>
  <head>
    <title>  Internet + Web Week 1</title>
  </head>
  <body>
    this is a webpage of wonderful text
  </body>
</html>
```

Parent / Child Element Structure



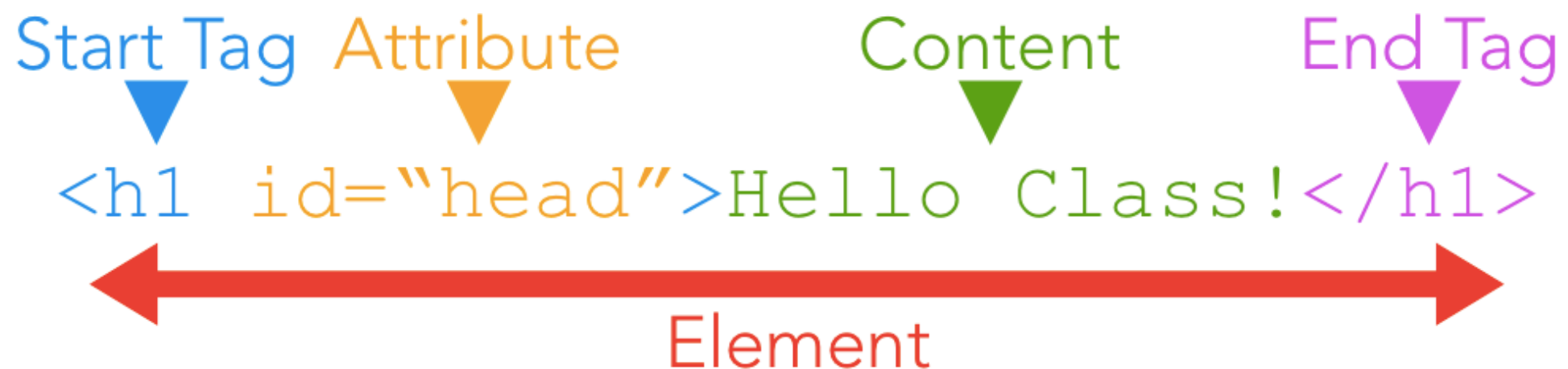
Parent + Child

```
<!doctype html>
  <head>
    head is the parent of title
    <title> Week 1 </title>
  </head>
  <body>
    div is the child of body
    <div>
      Here's a Great Site.
    </div>
  </body>
</html>
```

body is the child of html

HTML Elements / Tags, Attributes, Content

- Elements and tags used interchangeably



The `<head>` element contains the metadata for a web page. Metadata is information about the page that isn't displayed directly on the web page. Unlike the information inside of the `<body>` tag, the metadata in the head is information about the page itself.

Text tags

- **h1, h2, h3, h4, h5, h6** are text tags for headings
- **p** is a tag for paragraphs
- **b** is for bold, **i** is for italics
- **** is for **bold** **** is for *italics*
- **ul, ol, li** are used for making lists
 - **ul**: unordered lists
 - **ol**: ordered lists
 - **li**: an individual list tag
- **
** will break to a new line

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
```

`<div>`s can contain any text or other HTML elements, such as links, images, or videos. Remember to always add two spaces of indentation when you nest elements inside of `<div>`s for better readability.

Semantic HTML

HTML should be coded to represent the data that will be populated and not based on its default presentation styling. Presentation (how it should look), is the sole responsibility of CSS.

Some of the benefits from writing semantic markup are as follows:

- Search engines will consider its contents as important keywords to influence the page's search rankings (see SEO)
- Screen readers can use it as a signpost to help visually impaired users navigate a page
- Finding blocks of meaningful code is significantly easier than searching through endless divs with or without semantic or namespaced classes
- Suggests to the developer the type of data that will be populated
- Semantic naming mirrors proper custom element/component naming

<https://developer.mozilla.org/en-US/docs/Glossary/Semantics>

`<p>`

`<h1>` - `<h6>`

Semantic elements

`<main>`

dominant content of the `<body>` element

`<article>`

A document, page or site. This is usually a root container element after body

`<section>`

Generic section of a document

`<header>`

Intro section of a document

`<footer>`

Footer at end of a document or section

`<nav>`

Navigational section

Use these **before** div when appropriate.

Semantic elements

`<aside>`

represents a portion of a document whose content is only indirectly related to the document's main content. Asides are frequently presented as sidebars or call-out boxes.

`<details>`

creates a disclosure widget in which information is visible only when the widget is toggled into an "open" state.

`<figcaption>`

represents a caption or legend describing the rest of the contents of its parent `<figure>` element.

`<mark>`

represents text which is marked or highlighted for reference or notation purposes, due to the marked passage's relevance or importance in the enclosing context.

`<summary>`

element specifies a summary, caption, or legend for a `<details>` element's disclosure box. Clicking the `<summary>` element toggles the state of the parent `<details>` element open and closed.

`<time>`

represents a specific period in time.

The `` tag will generally render as *italic* emphasis.

The `` will generally render as **bold** emphasis.

`
`

The line break element is unique because it is only composed of a starting tag. You can use it anywhere within your HTML code and a line break will be shown in the browser.

tag attribute value

```
<video src= "filepath/file.mov" alt= "this is the video" height="300"></video>
```

```
<html attribute= "value" attribute= "value" attribute= "value"> </html>
```

Structure of a link

OPENING
LINK TAG

URL WE ARE
DIRECTED TO

TEXT WE
CLICK ON

CLOSING
TAG

```
<!-- link -->  
<a href="https://www.fordham.edu/" target="_blank">Fordham University</a>
```



FORDHAM
UNIVERSITY

< a href — stands for *hyperlink reference*

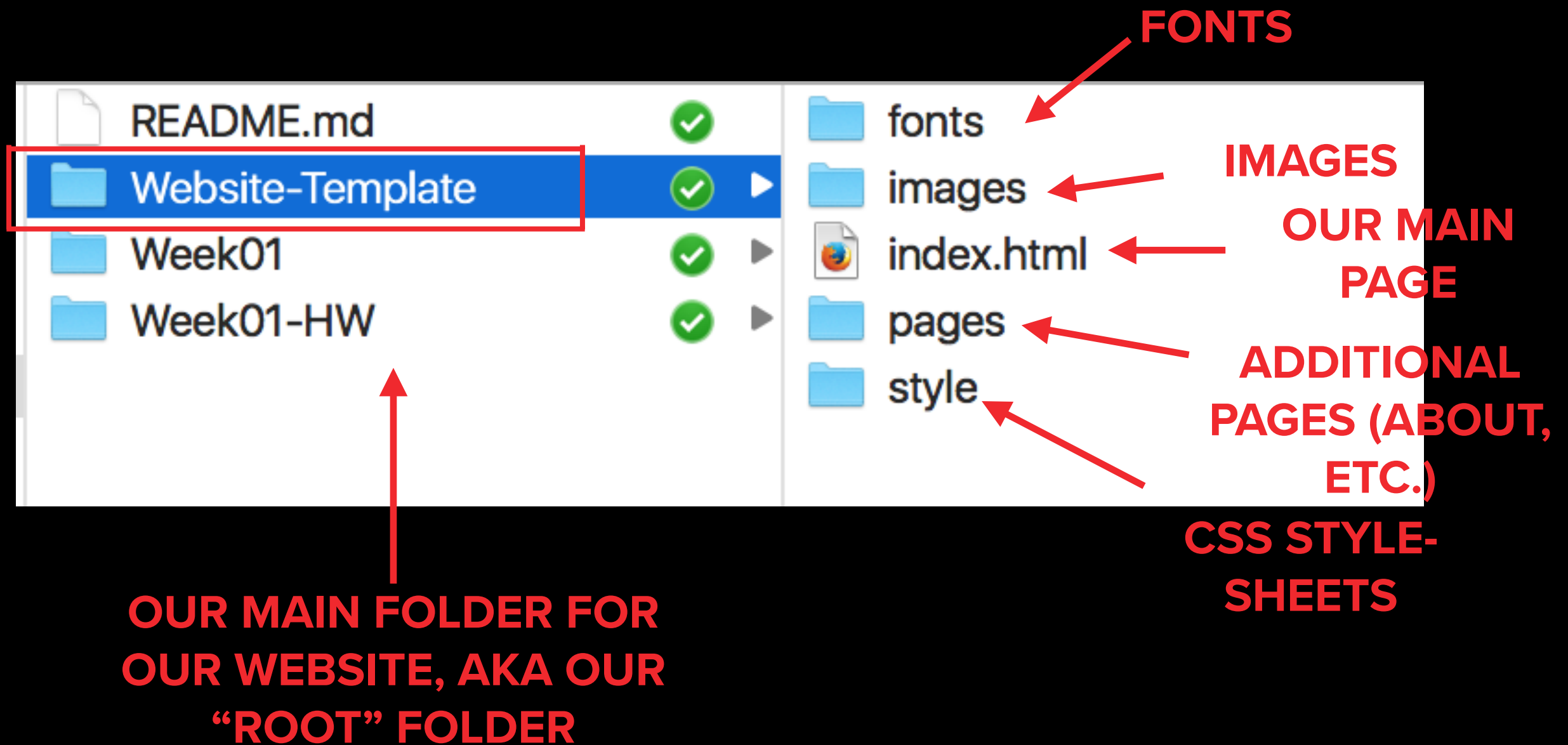
Linking to pages on the same site

RELATIVE URLS

Link types:

- parent folder: `Homepage`
- same folder: `Homepage`
 - child folder: `Photos`
 - id attribute: `Different element on page`

```
<p>  
  <!-- linking ot another page on the same site -->  
  <button type="button" onclick="window.location.href='/theHTML.html'">another web  
  page on this site</button>  
</p>  
<p>  
  <!-- linking to an id attribute on the same page -->  
  <button type="button" onclick="window.location.href='#theEnd'">Go to the End</  
  button>  
</p>
```

Why **index.html**?

The main homepage of a site written in HTML (and the homepage of each section in a child folder) is called `index.html`.

Web servers are usually set up to return the `index.html` file if no file name is specified. Therefore, it's always a good idea to name your "home" page `index.html`

The `` tag has a required attribute called `src`. The `src` attribute must be set to the image's source, or the location of the image. In some cases, the value of `src` must be the *uniform resource locator* (URL) of the image. A URL is the web address or local address where a file is stored.

Images: Relative (local) vs. URL

- The **** tag is for images, which can be on your local directory or on another webpage. Read all about **** tag [here](#)

```
<!-- An image on the local directory -->
```

```

```

```
<!-- Or with size specs -->
```

```

```

```
<!-- Image from another site -->
```

```

```

The **alt** attribute, which means alternative text, brings meaning to the images on our sites. The **alt** attribute can be added to the image tag just like the **src** attribute. The value of **alt** should be a description of the image.

```

```

1. If an image fails to load on a web page, a user can mouse over the area originally intended for the image and read a brief description of the image. This is made possible by the description you provide in the **alt** attribute.
2. Visually impaired users often browse the web with the aid of screen reading software. When you include the **alt** attribute, the screen reading software can read the image's description out loud to the visually impaired user.
3. The **alt** attribute also plays a role in Search Engine Optimization (SEO), because search engines cannot "see" the images on websites as they crawl the internet. Having descriptive **alt** attributes can improve the ranking of your site.

Like the `` tag, the `<video>` tag requires a `src` attribute with a link to the video source.

Unlike the `` tag however, the `<video>` element requires an opening and a closing tag.

<video /> structure

main
tag

poster

width/
height

control
attributes

```
<body>

  <!-- Adding video tag -->
  <video poster="media/listen.jpg" width="400px" preload loop autoplay controls>
    <source src="media/listen.mp4"/>
    <source src="media/listen.webm"/>
  </video>
</body>
```

different
sources

After the `src` attribute, the `width` and `height` attributes are used to set the size of the video displayed in the browser.

The `controls` attribute instructs the browser to include basic video controls: pause, play and skip. Unlike the `` tag however, the `<video>` element requires an opening and a closing tag.

The text, "Video not supported", between the opening and closing video tags will only be displayed if the browser is unable to load the video.

<audio /> structure

main
tag

control
attributes

```
<audio controls autoplay loop>
  <source src="audio/virginia.mp3" />
  <source src="audio/virginia.ogg" />
  <p>This browser does not support this audio format</p>
</audio>
```

different
sources

text is the
file cannot
be found

Some Media Attributes

- **Preload** - what preloads when the page loads
- **Controls** - if the play/stop buttons are visible
- **Autoplay** - if the video should start playing automatically
- **Loop** - if the video should loop on completion

Attributes

If we want to expand an element's tag, we can do so using an attribute. Attributes are content added to the opening tag of an element and can be used in several different ways, from providing information to changing styling. Attributes are made up of the following two parts:

- 1) The **name** of the attribute
- 2) The **value** of the attribute

One commonly used attribute is the `id`.

We can use the `id` attribute to specify different content (such as `<div>s`) and is really helpful when you use an element more than once.

```
<div id="intro">  
    <h1>Technology</h1>  
</div>
```

**** contains short pieces of text or other HTML. They are used to separate small pieces of content that are on the same line as other content.

```
<div>  
  <h1>Technology</h1>  
</div>
```

```
<div>  
  <p> Wherever there's a  
    <span>computer</span>, there's a skilled  
    person developing, maintaining, hacking,  
    advancing or simply using it.</p>  
</div>
```

Table structure

- **<table>** element is used to create a table (written out row by row)
- **<tr>** indicates each row
- **<td>** indicates each cell of a table

```
index.html
1  <!doctype html>
2  <html>
3    <head>
4      <title>Tables</title>
5    </head>
6    <body>
7      <!-- basic table structure -->
8      <table>
9        <tr>
10         <td>1</td>
11         <td>2</td>
12         <td>10</td>
13       </tr>
14       <tr>
15         <td>3</td>
16         <td>4</td>
17         <td>11</td>
18       </tr>
19       <tr>
20         <td>5</td>
21         <td>6</td>
22         <td>12</td>
23       </tr>
24     </table>
25
26   </body>
27 </html>
```

Table headings

- **<th>** is used to represent the heading for either a column or a row
- Even though there is no content, you should still use it to represent an empty cell
- Add **<scope>** to indicate if it's a heading for row or column

```
<!-- table with headings -->
<table>
  <tr>
    <th scope="col">Day of a week</th>
    <th scope="col">Sports activity</th>
    <th scope="col">Km</th>
  </tr>
  <tr>
    <th scope="row">Monday</th>
    <td>Run</td>
    <td>5</td>
  </tr>
  <tr>
    <th scope="row">Tuesday</th>
    <td>Run</td>
    <td>10</td>
  </tr>
  <tr>
    <th scope="row">Wednesday</th>
    <td>Run</td>
    <td>3</td>
  </tr>
</table>
```


Spanning columns

- Sometimes you may need the entries in a table to stretch across more than one column
- You can add ***colspan*** attribute on **<th>** or **<td>** to indicate how many columns that cell should run across

	Morning	Lunch	Afternoon	Evening
Monday	Run	Meeting	Work	Meeting friends
Tuesday	Workout and breakfast		Work	Relax
Wednesday	Day off			

Spanning rows

- Add *rowspan* attribute on `<th>` or `<td>` to indicate how many columns that cell should run across

	Morning	Lunch	Afternoon	Evening
Monday		Work	Work	Drinks
Tuesday	Run	Work	Relax	Dinner
Wednesday		Time off		Read

Text input

Username:

Password input

Username:

Password:

Text area

What is your favorite movie to watch?

What is your favorite movie to watch?

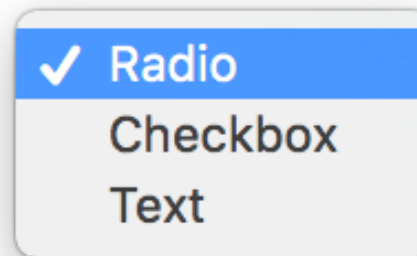
Checkbox

Select your favorite input type:

☐ Radio ☒ Checkbox ☒ Text

Drop down list

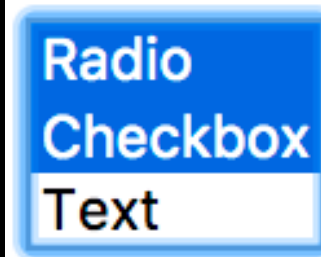
Select your favorite input type:



A drop-down menu with a light blue header and a white body. The header contains a checkmark icon and the text 'Radio'. The body contains the text 'Checkbox' and 'Text'.

Multiple select box

Select your favorite input type:



A multiple select box with a blue header and a white body. The header contains the text 'Radio'. The body contains the text 'Checkbox' and 'Text'.

Submit button

Are you ready to make that selection?

SUBMIT