

WEEK 1 | PROGRAMMING HTML AND ANATOMY OF WEB SITES

Shawn Park & Jeff Zhan



Web Design DeCal
DESIGN MEETS PROGRAMMING



Class Timeline: Programming





What Websites are Made Of

HTML

- Structure
- = Skeleton

CSS

- Design
- = Clothes

Javascript

- Function
- = Muscle

Pure HTML

Extra Topic



Menu
Programming Section
Extra Topic
Week 12 | May 1

[About](#)

Design Section
Final Project Presentations

[Lectures](#)

Programming Section
Final Project Presentations

[Outlook](#)

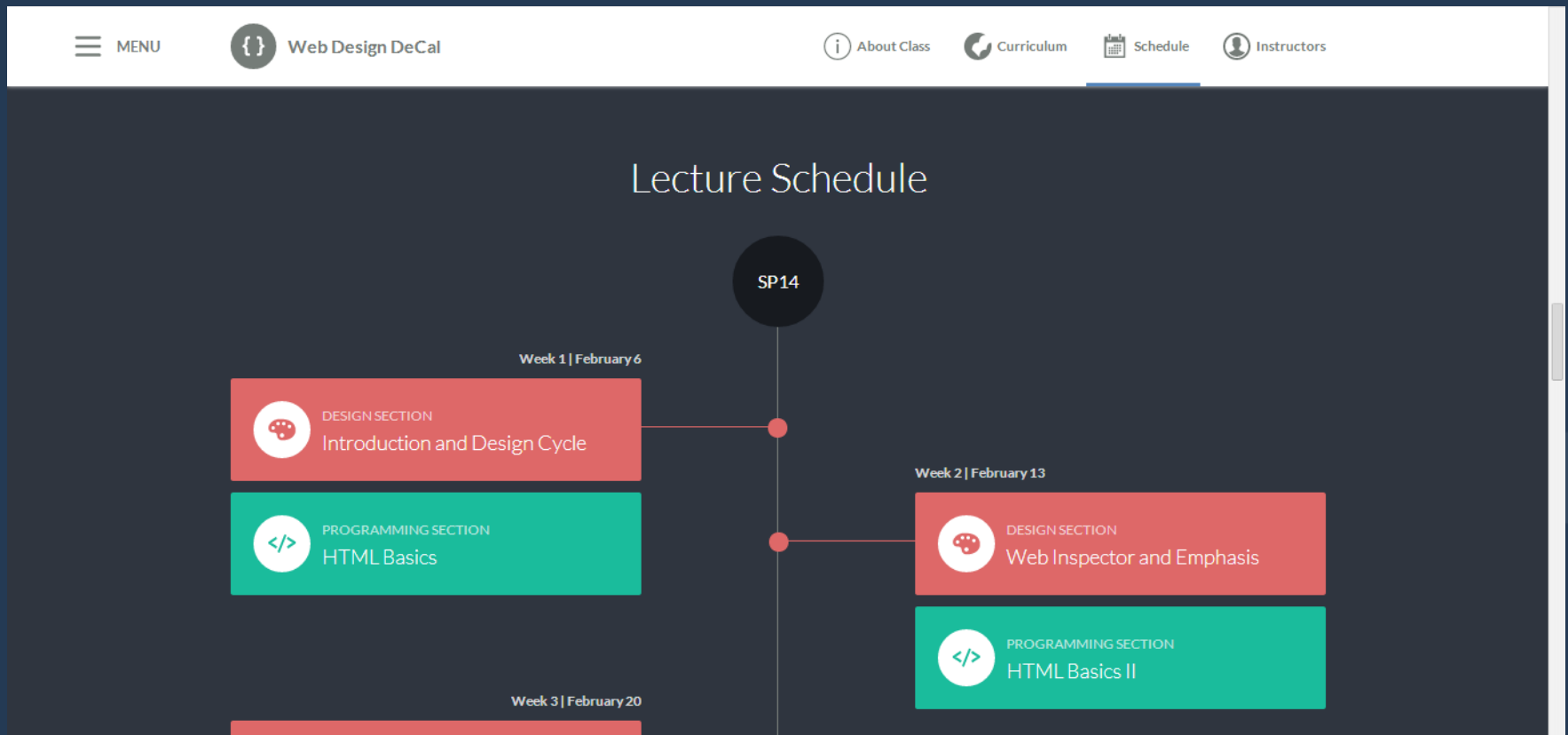
We are Cal students passionate about web design who want to share the passion with others.

Shown Park

[Piazza](#)

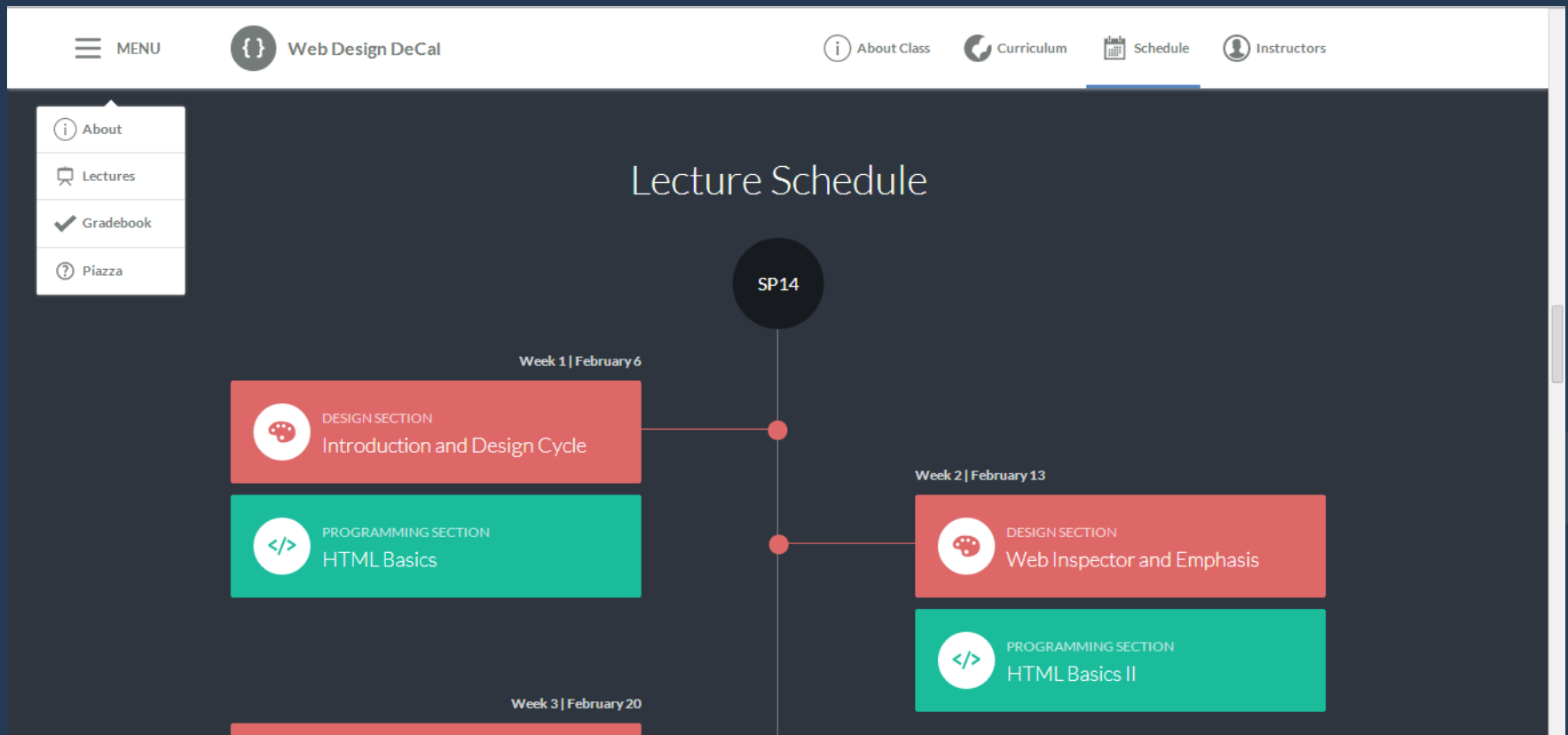


HTML + CSS





HTML + CSS + JS





Today's Outline

1. Editors & File Structure
2. HTML Structure
3. HTML Syntax
4. 1st In-Class Activity (yay!)





Goal Today:
Getting up and running in HTML



Editors & File Structure

Editors

- A **text editor** is a program that edits plain text files
- For us, we want one that is **code-friendly** and highlights a code's syntax
- Our picks:
 - Sublime Text 2 (free!), TextMate (mac, free!), Coda 2 (\$\$)

```
1  div#page>div.logo+ul#navigation>li*5>a
2
3  <div id="page">
4      <div class="logo"></div>
5      <ul id="navigation">
6          <li><a href=""></a></li>
7          <li><a href=""></a></li>
8          <li><a href=""></a></li>
9          <li><a href=""></a></li>
10         <li><a href=""></a></li>
11     </ul>
12 </div>
```



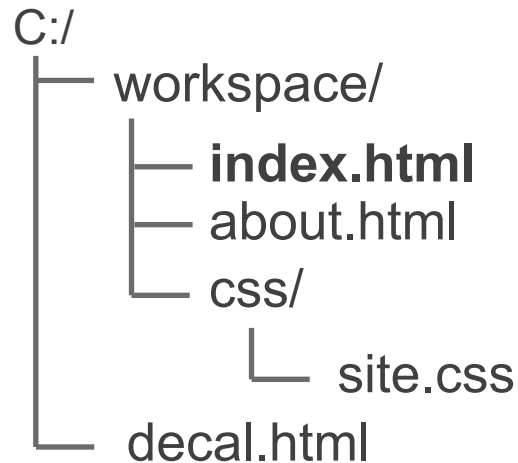
Relative and Absolute Paths

- Before we start, know these 2 key differences
- **Relative Path**
 - Path to a file *relative* to current file
 - Example: *Link to “page.html” or “pictures/sp14/decal.png”*
- **Absolute Path**
 - Complete path to a file or webpage
 - Example: *Link to “C:/Users/Jeff/Desktop/page.html” or “http://www.youtube.com”*



Relative and Absolute Paths

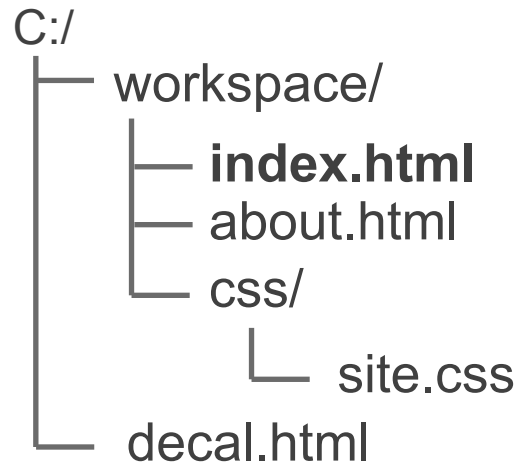
- **Example:** You are currently in *index.html*. How would you access *site.css* by relative path?



- Answer: “*css/site.css*”

Relative and Absolute Paths

- **Example:** You are currently in *index.html*. How would you access *site.css* by absolute path?



```
graph LR; C["C:/"] --- workspace["workspace/"]; C --- decal["decal.html"]; workspace --- index["index.html"]; workspace --- about["about.html"]; workspace --- css["css/"]; css --- site["site.css"]
```

A file system tree diagram starting from the root 'C:/'. It branches into 'workspace/' and 'decal.html'. Under 'workspace/', it further branches into 'index.html', 'about.html', and 'css/'. Under 'css/', it branches into 'site.css'.

- Answer: “*C:/workspace/css/site.css*”

Relative and Absolute Paths

- To go **up a folder** (going back), use two dots and a slash (`../`)

```
C:/
├── workspace/
│   ├── index.html
│   ├── about.html
│   └── css/
│       └── site.css
└── decal.html
```

- To access *decal.html* from *index.html*: “`../../decal.html`”
 - “Up two levels”



HTML Structure

Tags

- Tags start with a left bracket `<` and end with a right bracket `>`
- This is a paragraph tag: `<p>`
- There are *opening* tags and *closing* tags. Closing tags start with `</` instead
 - Example: `</p>`
 - Together, a set of tags is called an **element**
 - `<p></p>` is a paragraph element
- You can add *content* between a opening and closing tag
 - Example: `<p>Hello Class!</p>`

Tags

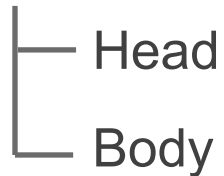
- **Opening tags must have a closing tag**
 - Not acceptable: `<p>Hello!`
 - Close it: `<p>Hello!</p>`
- There are *exceptions* for special tags
 - Example: `
`
 - This is a *line break* tag
 - Also: `<link type="text/css" rel="stylesheet" href="style.css">`
 - This is a *link* tag to your css file (we will go over this later)



Basic HTML Structure

- All webpages have **3 core elements** to them:
 - HTML tag
 - Head tag
 - Body tag
- The *head* and *body* tags are within the HTML tags

HTML



Basic HTML Structure

- All HTML pages must start with the `<html></html>` tags
 - This tells the browser to render meaningful HTML code
- Inside the HTML tags, put `<head></head>` tags

```
<html>  
  <head>  
  </head>  
</html>
```

- Notice we indented the `<head></head>` tags. This is for readability purposes (you could put everything in one line too, but we indent to make it easier for us to read)



Basic HTML Structure - Header

- What goes in `<head></head>` ?
 - Title of page, links to css & javascript files, search engine keywords and description, website info, etc.
 - Everything **not rendered** on the page (the content)
- **Title**
 - `<title></title>` tags
 - Defines the title displayed on browser windows/tabs



Basic HTML Structure - Header

```
<html>  
  <head>  
    <title>First Webpage</title>  
  </head>  
</html>
```



Basic HTML Structure - Body

- What goes in `<body></body>` ?
 - Content for your page
 - This is what the **viewer sees**
- **Heading Tags**
 - `<h1></h1>`, `<h2></h2>`, ..., `<h6></h6>`
 - 6 default heading sizes. 1 is the largest, 6 is the smallest
- **Paragraph Tags**
 - `<p></p>`
 - Adds some space above and below your paragraph



Basic HTML Structure - Body

- Other Useful Tags
 - `
` for a line break (jumps to next line for a text)
 - `...` for bold text
 - `...` for italicized text
 - `<hr>` for a single horizontal line



Images & Links

Images & Links

- **Links**

- `<a>...` tags
- Also called **anchor tags**
- Example:
 - `Youtube`
- `href` is an example of an element **attribute**
 - Attributes are followed by `=` and `"..."`, with something in `"..."`
 - `href` needs a url inside the `"..."`. Can be relative or absolute
- Between the `<a>...` tags is the text displayed on the browser

Hello World!

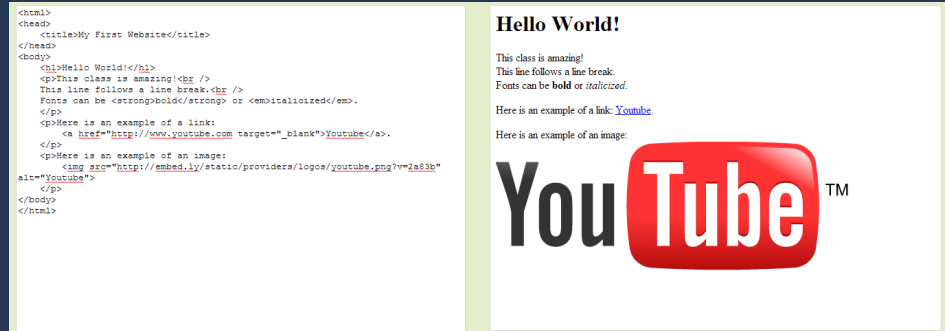
This class is amazing!

This line follows a line break.

Fonts can be **bold** or *italicized*.

Here is an example of a link: [Youtube](#).

Images & Links



- **Images**

- `` tag
- Image tags rely a lot on HTML attributes (like *href* in anchor tags)
- *src* attribute
 - Defines the image source (either relative or absolute path)
 - Example: ``
- *height* & *width* attributes
 - ``
 - Note: Number is in # of pixels
- *alt* attribute
 - Example: ``
 - Displayed if image is unavailable, or is read if using screen reader

Lists

- **Unordered Lists** (bullets)

```
<ul>
  <li>Jeff Zhan</li>
  <li>Kevin Liang</li>
  <li>Shawn Park</li>
</ul>
```

- **Ordered Lists** (numbers)

```
<ol>
  <li>Jeff Zhan</li>
  <li>Kevin Liang</li>
  <li>Shawn Park</li>
</ol>
```

Instructors:

- Jeff Zhan
- Kevin Liang
- Shawn Park

Instructors:

1. Jeff Zhan
2. Kevin Liang
3. Shawn Park




Lists

```
<html>
<head>
  <title>My First Website</title>
</head>
<body>
  <h1>Hello World!</h1>
  <p>This class is amazing!<br>
  This line follows a line break.<br>
  Fonts can be <strong>bold</strong> or <em>italicized</em>.
  </p>
  <p>Here is an example of a link:
    <a href="http://www.youtube.com" target="_blank">Youtube</a>.
  </p>
  <p>Here is an example of an image:
    
  </p>
  <hr>
  Instructors:
  <ul>
    <li>Jeff Zhan</li>
    <li>Kevin Liang</li>
    <li>Shawn Park</li>
  </ul>
</body>
</html>
```

Hello World!

This class is amazing!
This line follows a line break.
Fonts can be **bold** or *italicized*.

Here is an example of a link: [Youtube](#).

Here is an example of an image: 

Instructors:

- Jeff Zhan
- Kevin Liang
- Shawn Park

Summary

- **Relative Paths** are paths to a file *relative* to your current location
- **Absolute Paths** are the complete paths to a file
- HTML pages have **3 main elements**:
 - HTML tags
 - Head tags
 - Body tags
- **Head** tags define things *not visible* to the user
- **Body** tags define things *visible* to the user
- Images and Links take in **attributes** and depend on relative/absolute paths

All lecture material, handouts, and homework can be found at:
<http://www.thewebdesignworkshop.co> (an absolute path!)



Bonus Slides!

Structure of your Environment

- Many ways to structure your website. Here is one way:

```
workspace/  
├── index.html  
└── assets/  
    ├── js/  
    │   └── site.js  
    ├── css/  
    │   └── site.css  
    └── images/  
        └── image1.png
```

Head Tags

- Besides `<title></title>`, what else can you do?
- **Meta Tags**
 - `<meta>` tags define information on your page
 - It is an exception to the opening/closing tag pair
 - Search Engine Description:
 - `<meta name="description" content="Google shows this">`
 - Takes a *name* and *content* attribute
 - “Favicon” – The icon next to your website title on the browser:
 - `<meta property="og:image" content="/path/to/image.png"/>`
 - Takes a *property* and *content* attribute
 - *Content* is a relative/absolute path to your 16x16 or 32x32 favicon image (in .png or .gif preferably)

Div and Span Tags

- We will see this in the next few lectures
- `<div></div>` tags are the primary tags we will use in the future
 - They are essentially tags with no special properties (like `` or `<h1>`)
 - You customize them with CSS
 - They are stacked **vertically**, meaning you cannot have two divs side by side unless you alter the CSS
- `` tags are used from time to time
 - Like a div tag, they have no special properties
 - You customize them with CSS
 - They are stacked **horizontally**, meaning you can have two spans side by side