

Slides:

<http://livetotry.com/athena/class3.pdf>

HTML/CSS Class 3: CSS for Page Layout

Alexis Goldstein

@alexisgoldstein

alexis@autfaciam.com

Review

Review: Jargon so far

Let's review the terms & jargon we've learned thus far:

- HTML terms:
 - **Tag**
 - **Element**
 - **Attribute**
- CSS terms:
 - **Element Selector**
 - **Class Selector**
 - **Id Selector**

Brief review of terms

Tag

Tags are used to denote the start of an element (i.e. `<p>`) or the end of an element (i.e. `</p>`). A tag is either a start tag or an end tag.

Examples of tags: **``**, **`<html>`**, **`</p>`**, **`</body>`**

Element

An element is the start tag + its content + the end tag:

`<p>This is some paragraph text</p>`

Attribute

Attributes provide additional information about HTML elements.

- Attributes are formatted like this: `attr="value"`
- The attribute always goes in the **opening tag**, never in the closing tag.
- Examples:
 - In `go to google`, **href** is the attribute.
 - In ``, **src** is the attribute.

Property:Value;

The hallmark of all CSS is the combination of two things:

- A property
- A value
 - We separate the property from the value with a **colon**.
 - We end the value with a **semicolon**.

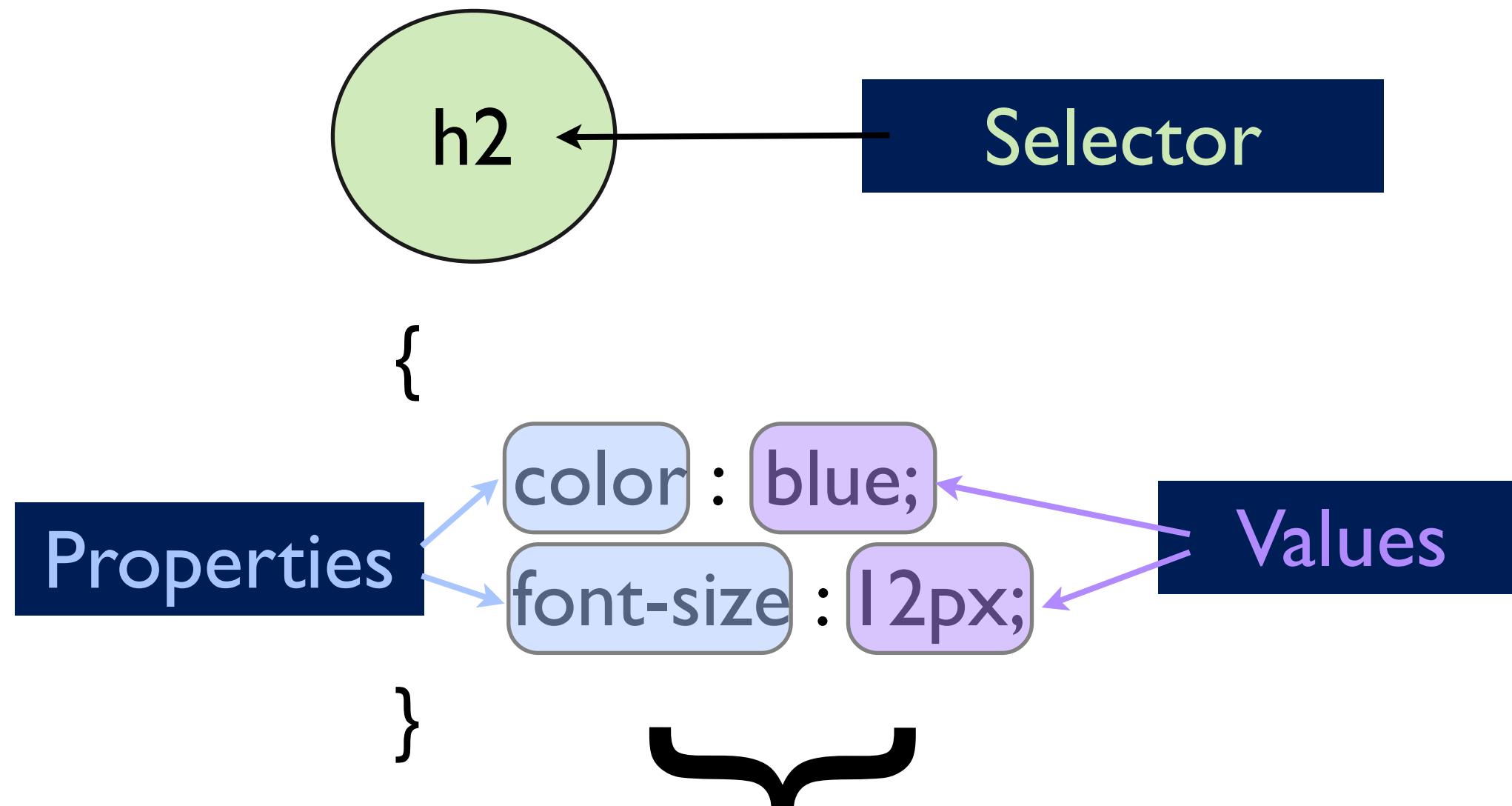
Property

:

Value

;

CSS Syntax



Declaration = property: value;

Brief review of terms

Element Selector

CSS element selectors let you style all elements of a certain type on an HTML page. For example, you can style all paragraphs by using the “p” selector.

Id Selector

CSS id selectors let you set “labels” on one specific item of a page. Unlike class selectors, you cannot reuse id selectors. Id selectors should only be used once.

CSS Element Selectors

h2

Element
Selector

```
{  
    color : green;  
}
```

```
<html>  
<head></head>  
<body>  
<h1>Title!</h1>  
<h2>Hello!</h2>  
<p>test test test...</p>  
<h2>Another heading!</h2>
```


CSS ID Selectors

#title

ID Selector

{

color : red;

}

.first {

Class Selector

color : blue;

}

h1 {

Element Selector

color : green;

}

```
<html>
```

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

```
<body>
```

```
<h1 id="title"
```

```
class="first">Title!</h1>
```

```
<h2>Hello!</h2>
```

```
<p>test test test...</p>
```


```
<h2
```

```
class="first">Another  
heading!</h2>
```

Id Selector Example

```
#wrapper  
{  
  width: 700px;  
  margin: 0px auto;  
  border: 2px solid black;  
}
```

```
.  
. .  
<body>  
  <div id="wrapper">  
    <p>hello!</p>  
  </div>  
</body>  
</html>
```



CSS Precedence

- The most specific CSS Selector will “win” and override any duplicate properties specified by other selectors.
- That’s why:
 - The class=“first” and .first Class Selector changed the color of the second H2 element from green to blue.
 - The #title ID Selector changed the color from blue to red-- ID selectors are seen as more specific than Class selectors.

CSS Pseudoclasses

a:link

```
{  
    text-decoration: none;  
}
```

a:hover

```
{  
    text-decoration:  
    underline;  
}
```

```
<html>  
<head></head>  
<body>  
<h1 id="title"  
class="first">Title!</h1>  
<h2>Hello!</h2>  
<p>test test test...</p>  
<h2 class="first">Another  
heading!</h2>  
<a href="">I'll be underlined  
ONLY when you hover</a>  
</body>  
</html>
```

Class Selectors

Class Selectors

- **Class Selectors** let you set “labels” on elements, and style each labelled element differently.
- You can reuse the same class selector as many times as you want.

CSS Class Selectors

You set these labels in HTML by assigning a **class** attribute:

```
<p class="blueParagraph">hi!</p>
```

```
<p class="greenParagraph">hellooooo!</p>
```

How do we define this in CSS?

CSS Class Selectors

HTML:

```
<p>this text will be black</p>  
<p class="blueParagraph">hi!</p>  
<p class="greenParagraph">hellooooo!</p>
```

CSS:

```
p  
{  
    font-family: Monaco, Arial, sans-serif;  
    text-align:right  
}  
.blueParagraph  
{  
    color: blue;  
}  
.greenParagraph  
{  
    color: green;  
}
```


CSS Precedence

- With all these different types of selectors, is it possible that some may override others?
Yes!
- To read more about the precedence, go here: <http://www.vanseodesign.com/css/css-specificity-inheritance-cascaade/>
- The short way to avoid precedence problems?
- If your CSS isn't working the way you expect, add **!important** to the end of your property. This will force it to have highest precedence.

Stylesheet Review

Review of types of Stylesheets

There are three ways to insert styles on an HTML page:

- External Stylesheet
- Internal Stylesheet
- Inline Styles

Three ways to use Stylesheets

Inline Style:

```
<p style="font-family: Monaco, Arial, sans-serif;color:blue;"> This is my  
first paragraph of text.</p>
```

Internal Stylesheet:

```
<head>  
  <style>  
    p  
    {  
      font-family: Monaco, Arial, sans-serif;  
      color:blue;  
    }  
  </style>  
</head>  
<body>  
  <p>This is my first paragraph of text.</p>  
  <p>This is my second paragraph of text.</p>  
  <p>This is my third, also super! exciting!!, paragraph of text.</p>  
</body>
```

Three ways to use Stylesheets

External Stylesheet:

HTML file:

```
<head>  
  <link rel="stylesheet" type="text/css" href="stylesheet.css">  
</head>  
<body>  
  <p>This is my first paragraph of text.</p>  
  <p>This is my second paragraph of text.</p>  
  <p>This is my third, also super! exciting!!, paragraph of text.</p>  
</body>
```

stylesheet.css file:

```
p  
{  
  font-family: Monaco, Arial, sans-serif;  
  color:blue;  
}
```

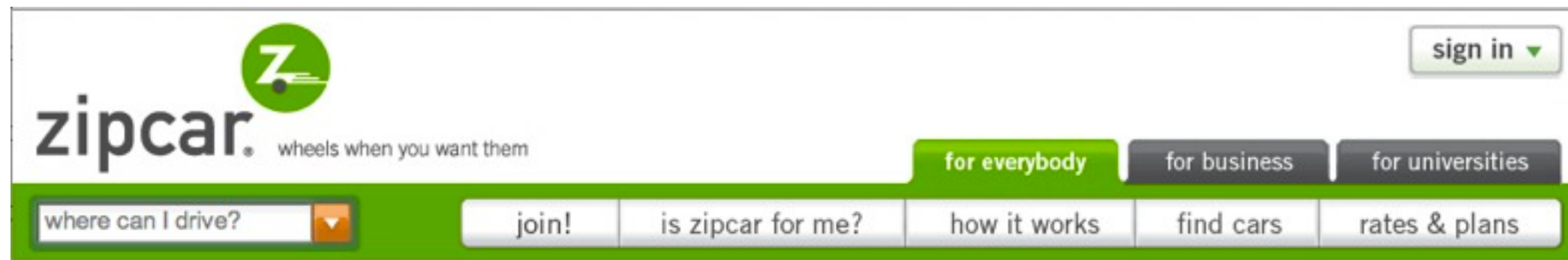
Typical Web Layouts

- Before we conclude the HTML section, let's talk about typical web layouts.
- A common pattern is for a website to have a header and a footer.



Headers

- Headers typically contain a logo image, the website title, and often a series of navigation links (“Home” “About” “Products” etc.)



Footers

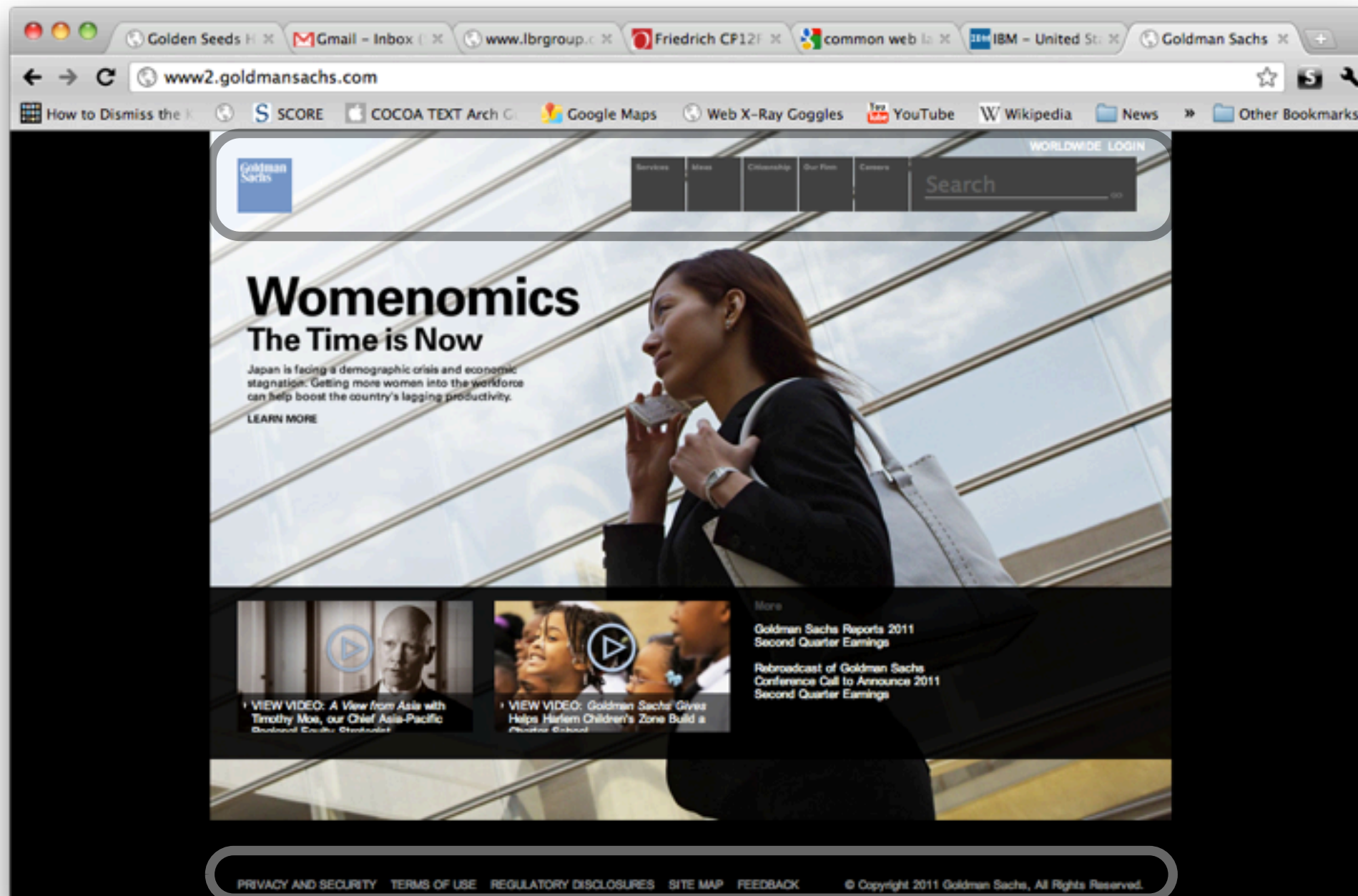
- Footers may contain things like copyright information, contact details, links to the privacy policy, etc.

Copyright © 2011 Apple Inc. All rights reserved. [Terms of Use](#) | [Privacy Policy](#)



[Choose your country or region ▶](#)

Header and Footer Example



Grouping content

- In between the header and the footer is typically where the main content goes.
- How we style these three (and often more) sections of pages, is often with the use of another HTML element, the **<div>**.

<div>

- The div tag is a great way to apply styles to a bunch of elements all at once.
- With the **<div>**, we can group several elements together, so once we start applying CSS, we can apply the same CSS to several elements at once.

Practicing <div>

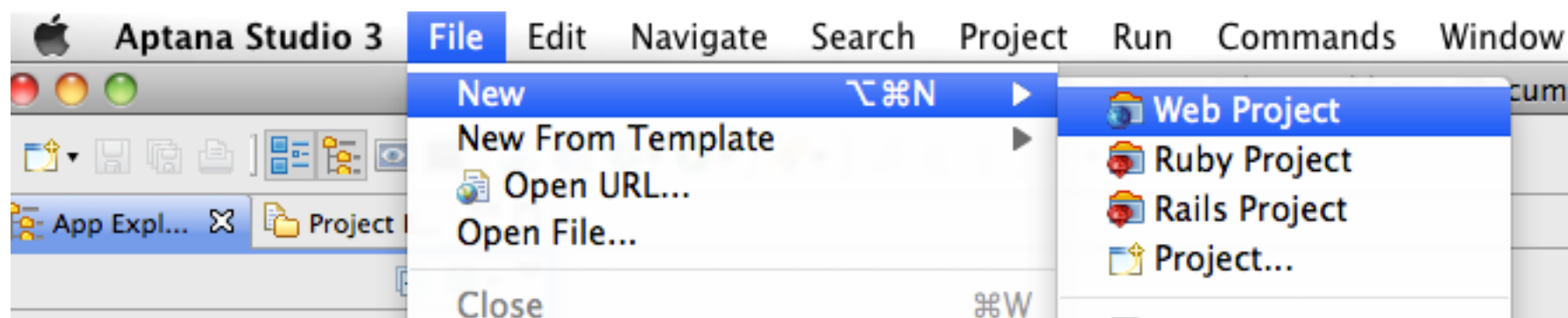
- Let's practice both the use of **div** and the concept of External Stylesheets.
- We'll do this by creating two files to start:
 - newpage.html
 - base.css

Setting up in Aptana 2

- To get started in Aptana 2:
 - Go to File > New > Untitled HTML file.
 - Name the file index.html
 - Save, making a note of **which folder** you're saving to.
 - Go to File > New > Untitled CSS File
 - Name the file style.css
 - Save it to the **same folder** you saved your HTML file to!

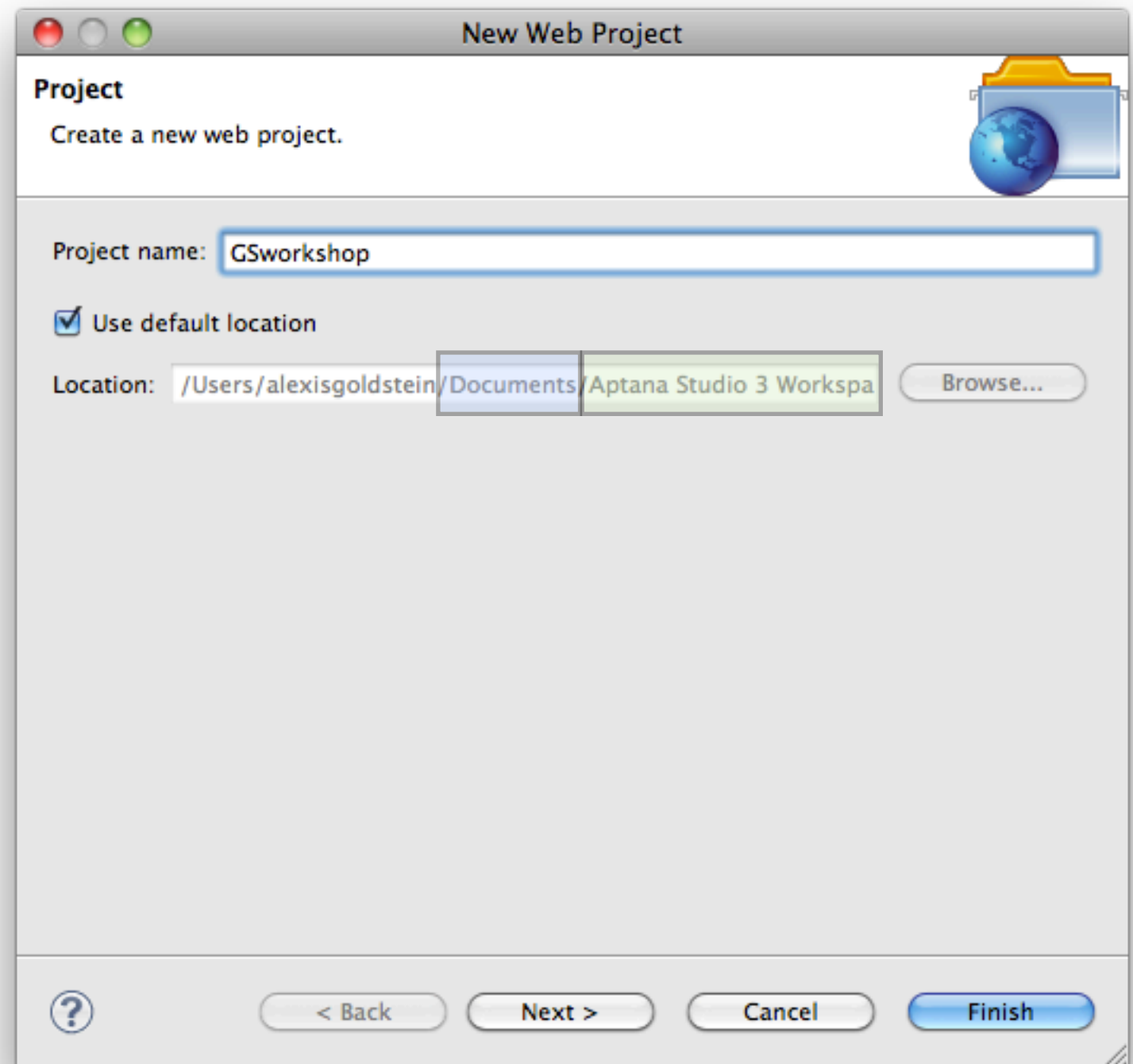
Creating a new project with Aptana v3

- The first step is to go to **File > New > New Web Project**



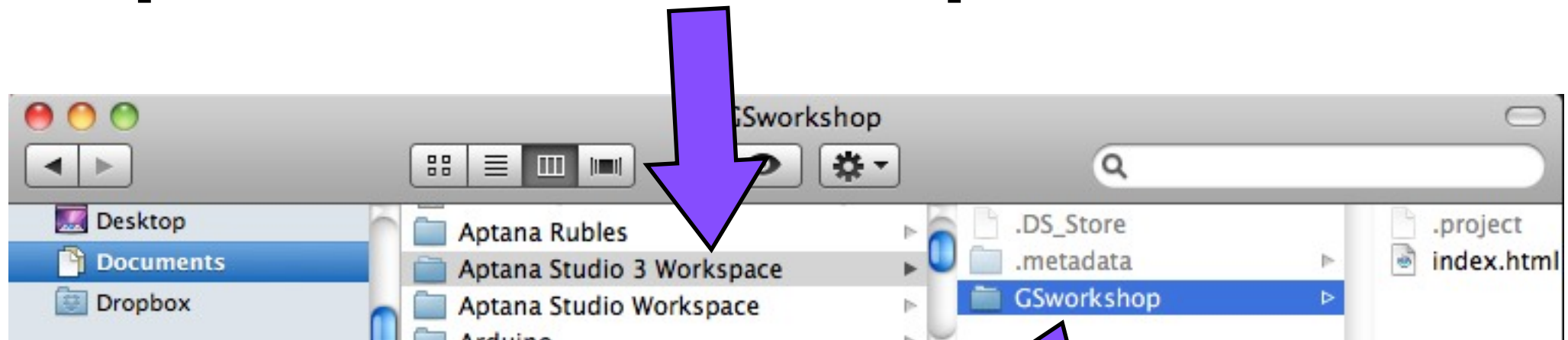
Where Projects are saved

- The second step is to give you Project a name
 - By default, the new Project will be saved to your Documents folder, under a folder named “Aptana Studio 3 Workspace”
- Don't click **Finish** just yet!



Where Projects are saved

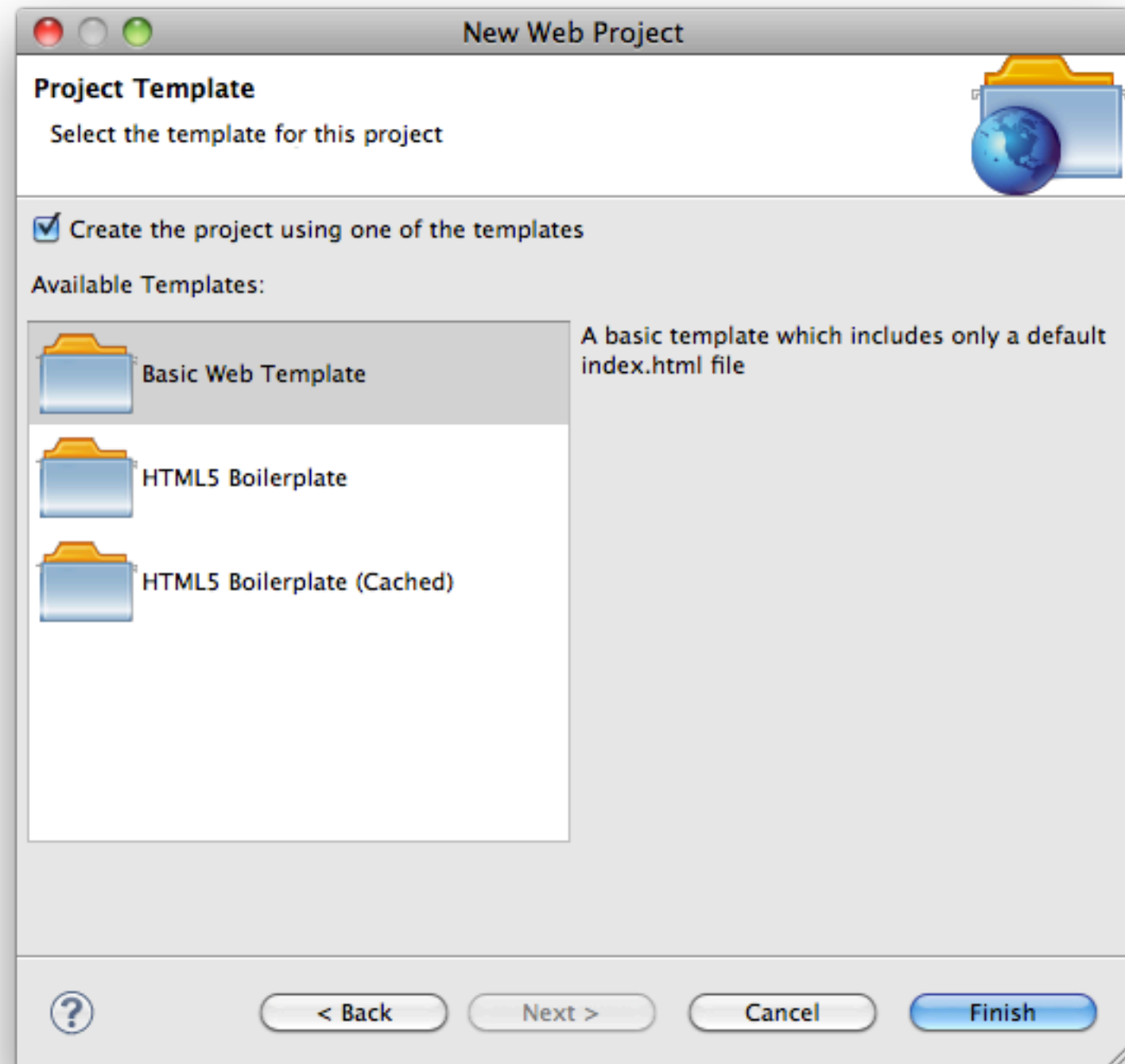
- By default, the new Project will be saved to your Documents folder, inside a folder named **Aptana Studio 3 Workspace**



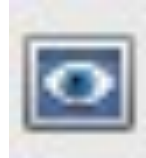
- I named my project GSworkshop. This folder will hold my webpage files.

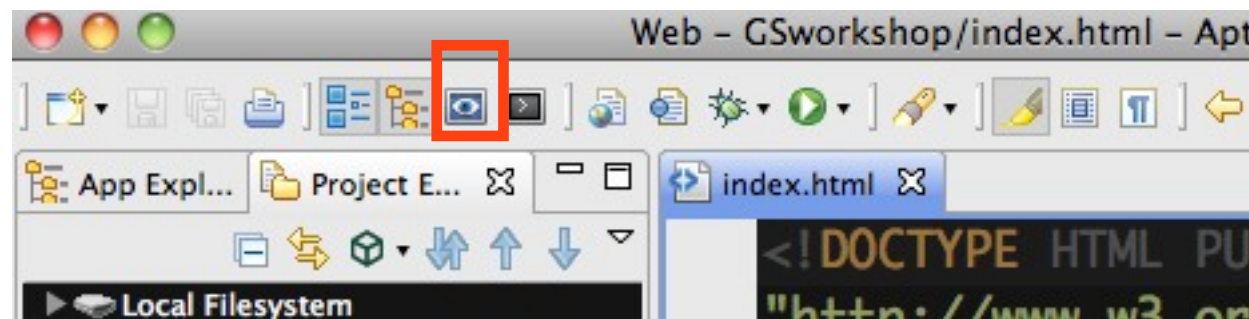
Make the Project use the Basic Web Template

- Click “Create the project using one of the templates”
- Choose “Basic Web Template”
- Click **Finish**



Setting up our Preview window

- To view a preview of your page in Aptana version 3, click the icon that looks like an eye. 
- It's at the top of Aptana in the toolbar



Setting up our Preview window

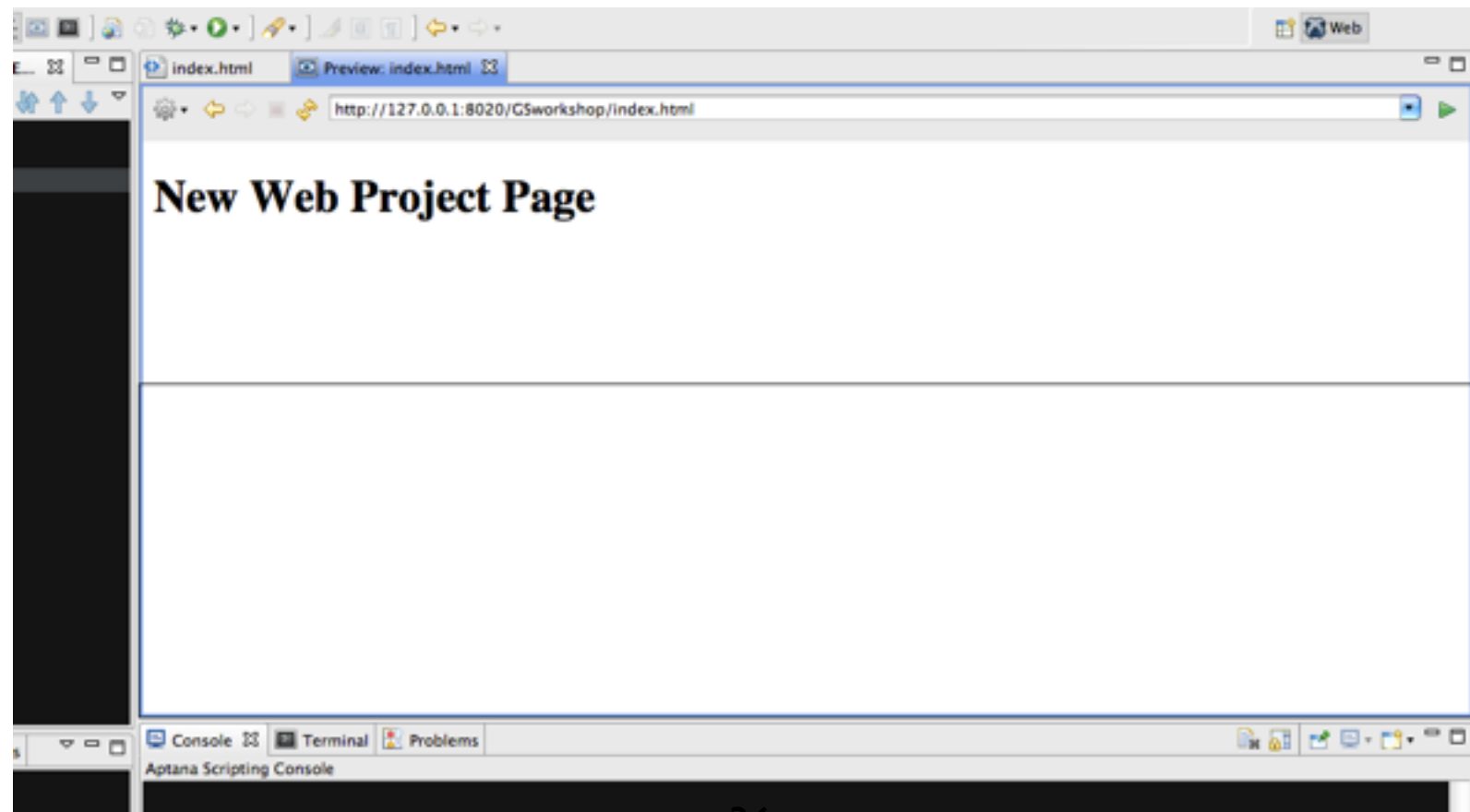
- Clicking this eye icon will create a new tab called “Preview”



- This will allow us to view the changes we are making to our HTML file.

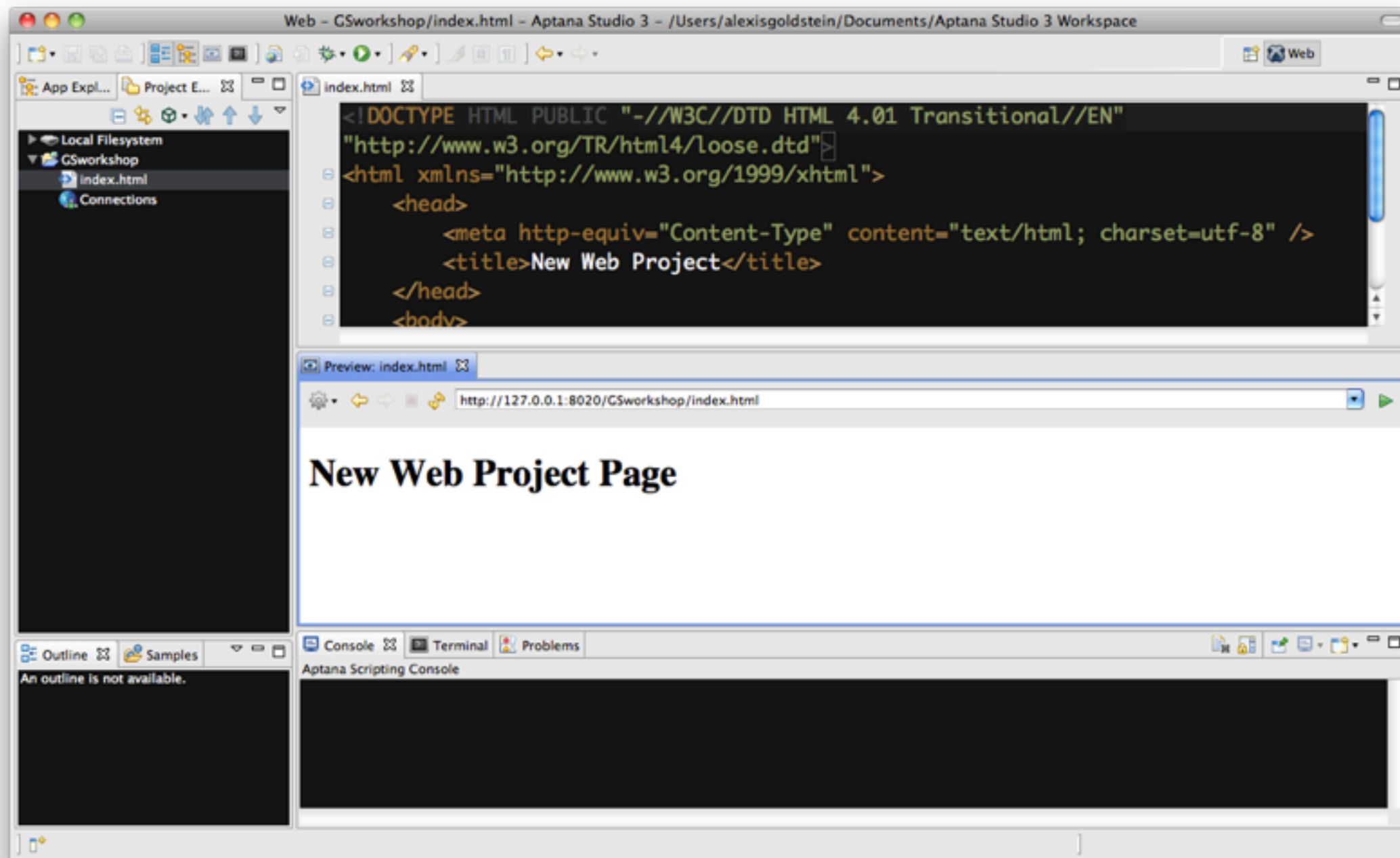
Setting up our Preview window

- Instead of having the preview in a separate tab, I prefer to see the preview underneath my HTML file.
- To achieve this, click and drag the preview tab down towards the bottom of the screen, then let go when you see a black rectangle:



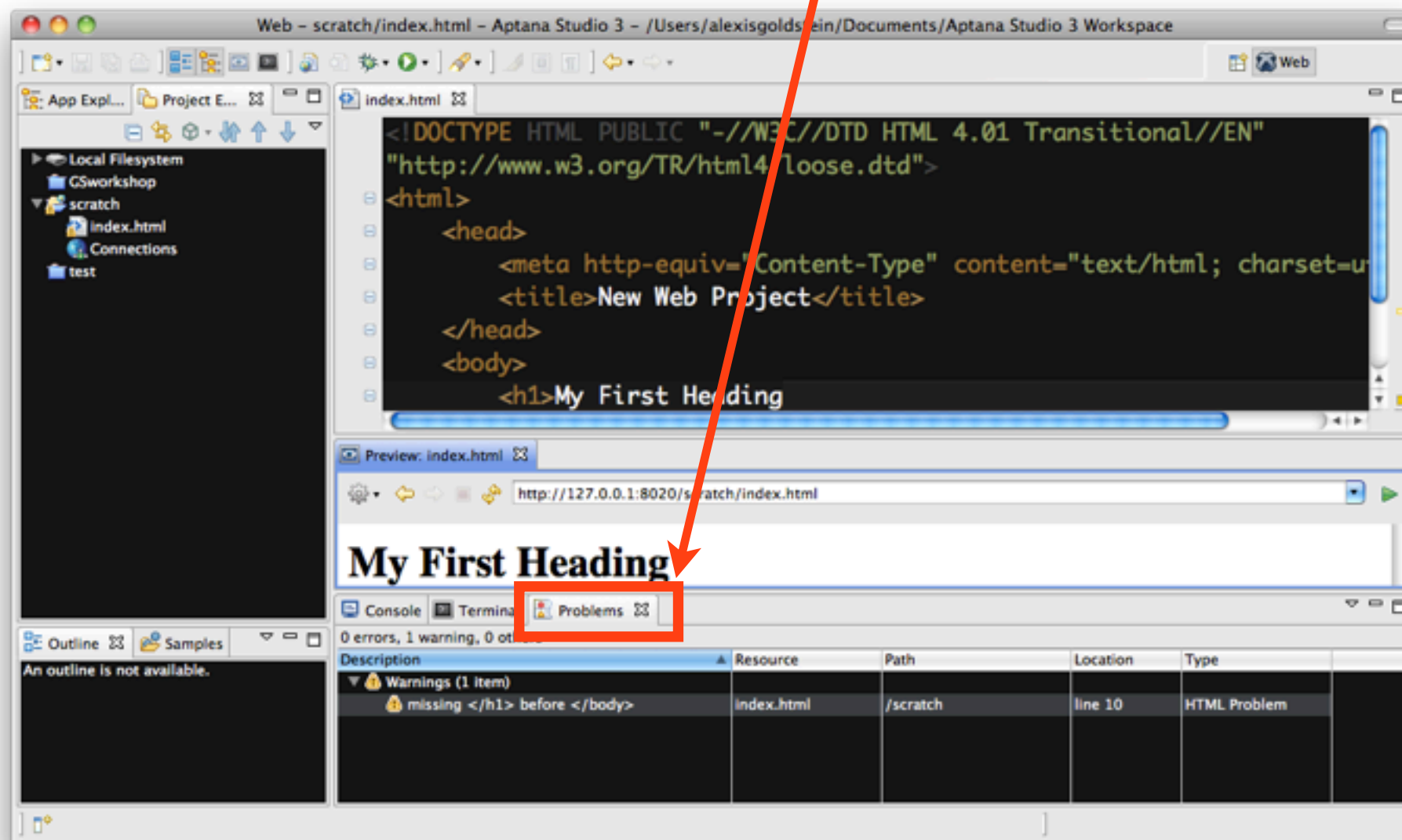
Setting up our Preview window

- The preview tab should anchor itself below your code, so you get a nice split screen.



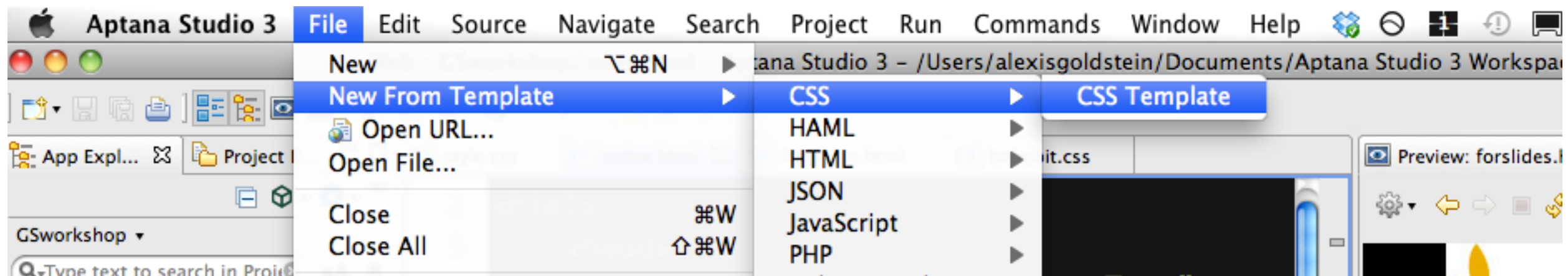
When things go wrong

- If you have any errors in your HTML, they will show up in the **Problems** tab:



Creating a separate CSS file

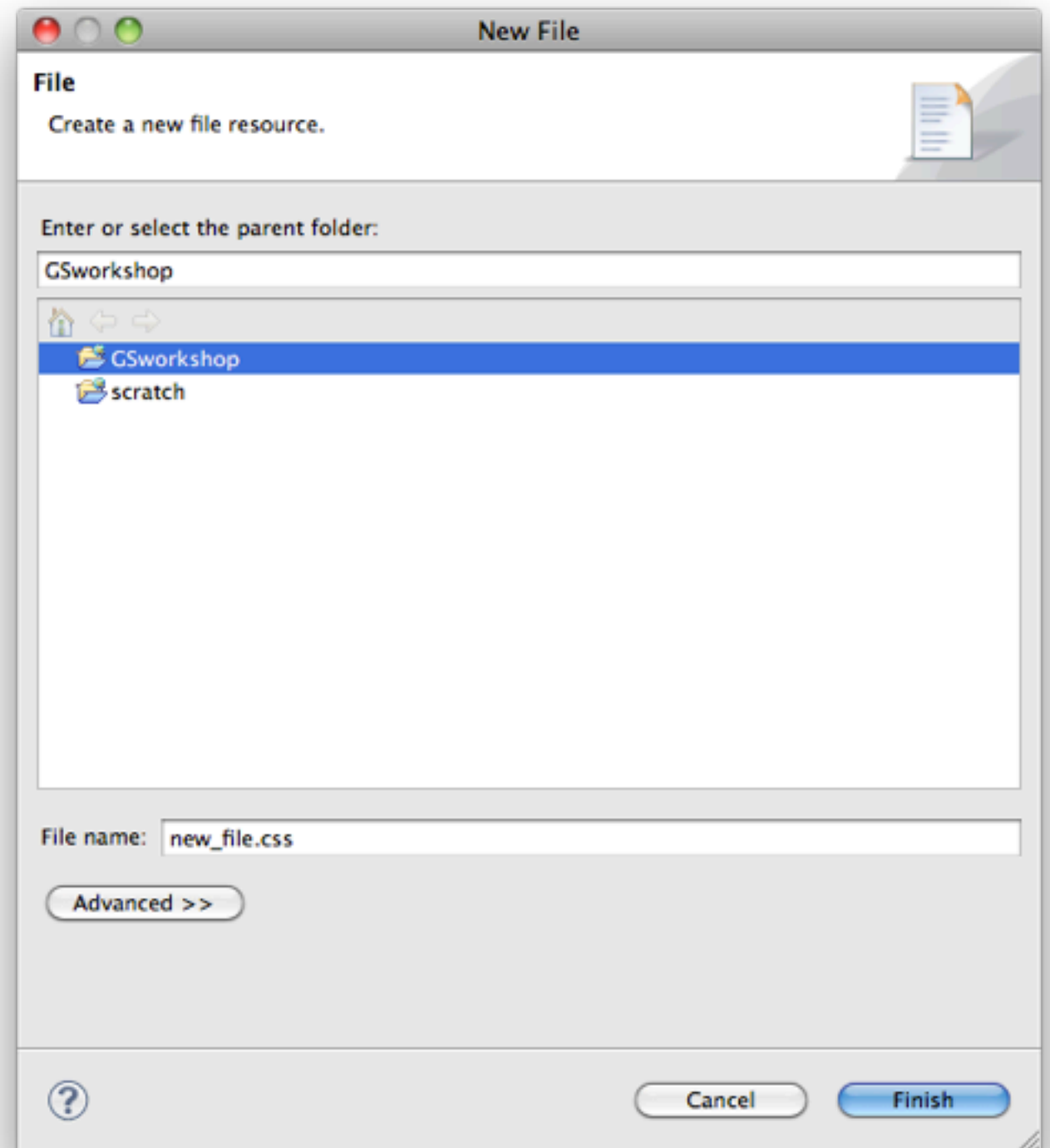
To add a new CSS file to our project, go to **File > New from Template > CSS > CSS Template**



Creating a separate CSS file

A window will open asking you:

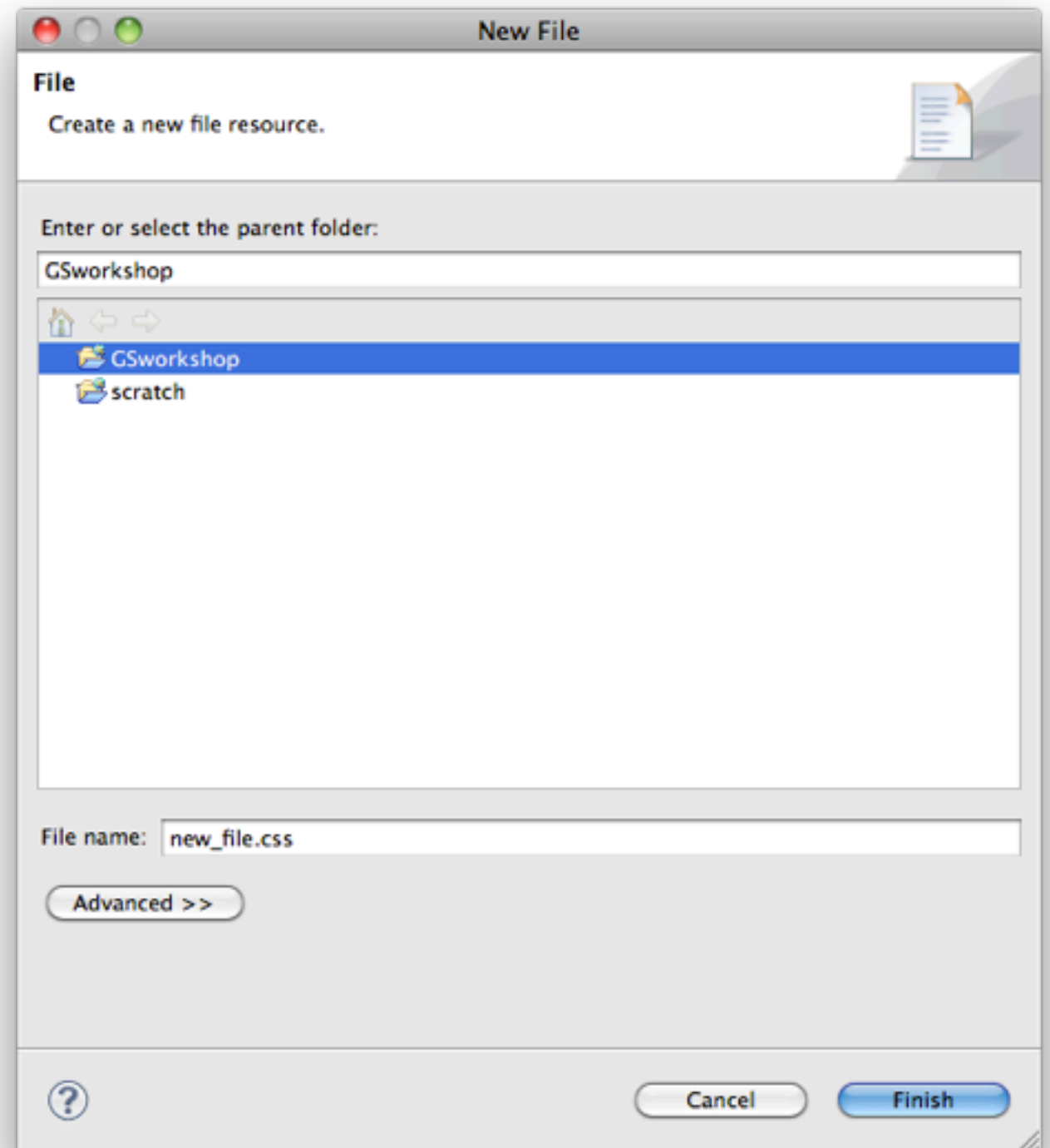
- Which project to add the file to
- What to name the file
 - Make sure you add the new file to the same project you've been working on



Creating a separate CSS file

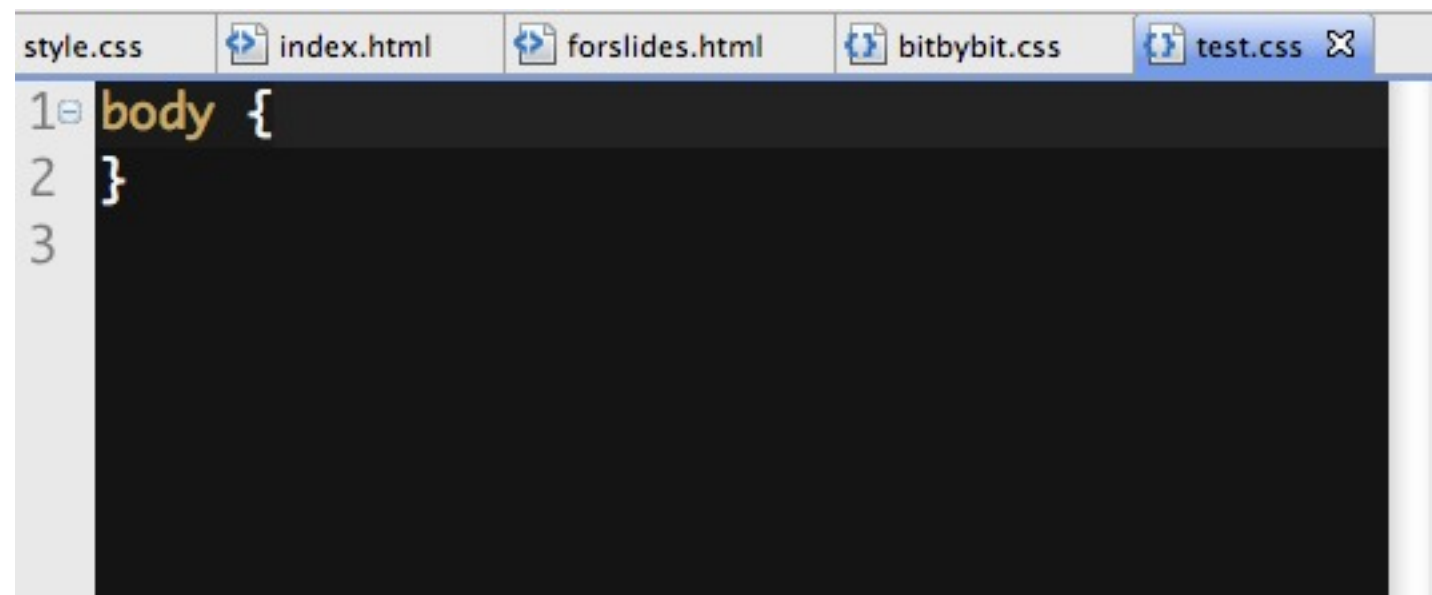
A window will open asking you:

- Which project to add the file to
- What to name the file (I suggest “style.css”)
 - Make sure you add the new file to the same project you’ve been working on



Our new CSS file

Our new CSS file will look like this:



```
1 body {  
2 }  
3
```

The image shows a code editor window with several tabs at the top: style.css, index.html, forslides.html, bitbybit.css, and test.css. The test.css tab is active. The editor area is dark, and the code is written in a light color. The code consists of a single CSS rule for the 'body' selector, which is currently empty, enclosed in curly braces. Line numbers 1, 2, and 3 are visible on the left side of the editor.

Linking our HTML file to our CSS file

- We need to link our HTML file to our new CSS file.
- We do this via the **<link>** element.
 - <link> is a self-closing tag
 - <link> goes in the <head> section of our HTML file.

Linking our HTML file to our CSS file

- We need to link our HTML file to our new CSS file via the **<link>** element.
 - <link> requires two attributes, **rel** and **href**.

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

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html401/dtd/html401.dtd">
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html" />
    <title>Golden Seeds HTML/CSS Workshop</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <div id="header">
```

Setting up the header div

Let's begin by adding a div that will hold our header.

```
<body>  
  <div id="header"></div>  
  
</body>
```

Setting up the main div

Next, let's add a div that will hold the main content of our page:

```
<body>  
  <div id="header"></div>  
  
  <div id="main"></div>  
</body>
```

Setting up the footer div

Finally, let's add a div that will hold the footer of our page:

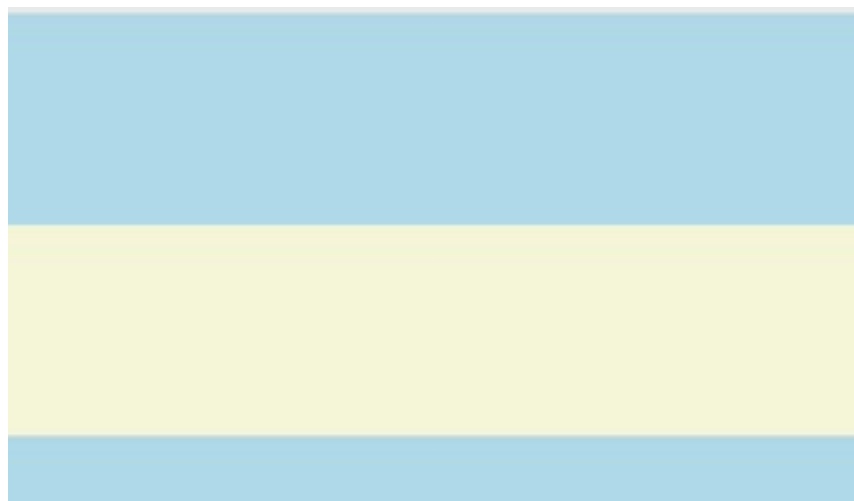
```
<body>  
  <div id="header"></div>  
  <div id="main"></div>  
  
  <div id="footer"></div>  
</body>
```

Styling our divs

Let's add CSS to each of the div **ids** via **id selectors** so we can see where they begin and end

HTML

```
<body>
  <div id="header"></div>
  <div id="main"></div>
  <div id="footer"></div>
</body>
```



```
#header{
  background-color: lightblue;
  min-height: 100px;
}

#main
{
  background-color: beige;
  min-height: 100px;
}

#footer
{
  background-color: lightblue;
  min-height: 50px;
}
```


Adding content to our layout divs

Let's add a logo image and an h1 to our header:

```
<body>
  <div id="header">
    
    <h1>Soho Soda Fountain</h1>
  </div>
  <div id="main"></div>
  <div id="footer"></div>
</body>
```

Adding content to our layout divs

- The image sits above the h1 by default.
- It would be nice to move the h1 next to the image.
- We can do that with CSS



Soho Soda Fountain

Styling our header

- First, let's change the display style of our `h1` to be `display: inline` (the default for `h1` is **block**).



Soho Soda Fountain

Moving our h1

- Right now, our h1 is aligned with the bottom of the image.
- We're going to use two CSS properties to fix this:
 - padding
 - vertical-align

vertical-align

- vertical-align allows us to set the vertical alignment of an element, moving it up or down.
- The default vertical placement is at the baseline of its parent element.

```
img  
{  
    vertical-align: middle;  
}
```



Soho Soda Fountain

Styling the header with borders

- Finally, let's change the background-color and add two borders to the entire header by the header **id selector**:

```
#header {  
  background-color: black;  
  height: 100px;  
  border-top: 10px solid black;  
  border-bottom: 10px solid black;  
  color: white;  
}
```



Getting rid of the browser default margin

- There's a bit of whitespace around our header.
- This is because the browser applies a default margin to the **body** element.
- We can fix this by adding **margin: 0px;** to our **body** selector (at the top of your CSS file):

```
body
{
    font-family: "Helvetica Neue", Helvetica, Arial, sans-serif;
    margin: 0px;
}
```

Styling the header with borders

- There's not enough space between the image and the h1:



- We could style all **h1** elements via an element selector, but is there a better way?
- What if we could target *only* h1s that live inside **id="header"** ?

CSS Nesting

- Nesting allows us to target elements **only** if they are **nested** inside of other elements.
- We specify a nested selector by listing it after another “parent” selector, separated by one space:

Nesting lets us grab an element within another element.

```
#header h1  
{  
  padding-left: 20px;  
}
```

This nested selector says: Style an **h1** element, but **ONLY** if it lives inside an element with id=“header”

Adding content to the footer

- Now let's add content to the footer
- I added some links and a paragraph containing a copyright statement.

```
<div id="footer">  
  <a href="">About</a>  
  <a href="">Menu</a>  
  <a href="">Location</a>  
  <p>  
    &copy; 2011 Soho Soda Fountain  
  </p>  
</div>
```

Styling Links in the Footer

- Let's use pseudoclasses to style the links in the footer

```
#footer a {  
    font-size: 20pt;  
    text-decoration: none;  
    font-weight: bold;  
    color: white;  
}  
  
#footer a:hover {  
    background-color: white;  
    color: black;  
}
```

Styling the footer

- Let's practice CSS nesting again by targeting *only* the links that live inside **div id="footer"**:

Our footer isn't at the bottom of the page

- Q: How can we force the footer to stick to the bottom of the page?



- A: With CSS Positioning.

CSS for Page Layout

Styling Images with Float

- With CSS float, an element can be pushed to the left or right, allowing other elements to wrap around it. When an element is set to float, text and other content will flow around the floated element.

The **float** property specifies whether or not an element should float. It also specifies which direction it should float (left, right). Example:

```
.alignLeft  
{  
    float: left;  
}
```

Styling Images with Float

This is most commonly used with images, in order to align them left or right so text flows around an image. It is also useful when working with layouts.

Let's give this a try:

1. http://www.w3schools.com/css/tryit.asp?filename=trycss_float
2. http://w3schools.com/css/tryit.asp?filename=trycss_float_elements

Styling Images with Float: Lab

<http://jsfiddle.net/fiddlefiddle/rFjRq/>

Completed version: <http://jsfiddle.net/rFjRq/II/>

Using clear

The clear property specifies which sides of an element where other floating elements are not allowed.

Best described visually! See this in action: http://www.w3schools.com/css/tryit.asp?filename=trycss_float_clear

Positioning with CSS

- CSS provides four main ways to position your content:
 - Static
 - Fixed
 - Relative
 - Absolute
- If you use these in combination with the CSS Box model, and the use of a property called **float**, you can layout your websites in myriad ways.

Positioning with CSS

- `position: fixed;`
- `top: 10px;`
- `right: 10px;`
- `left: 10px;`
- `bottom: 10px;`

Static and Fixed Positioning

Static Positioning

HTML elements are positioned static by default; there is no need to set them to static. Static positioned elements are positioned according to the normal flow of a page. They ignore anything specified by top, bottom, right or left properties.

Fixed Positioning

An element with fixed position is positioned relative to the browser window.

It will not move even if the window is scrolled--it will always stay in the same, fixed location on the screen.

See this in action:

- http://www.w3schools.com/Css/tryit.asp?filename=trycss_position_fixed
- <http://alexisgo.com/teaching/resistor/fixedPosExample.html>

Fixed Positioning Lab!

For practice with fixed positioning, we're going to build a menubar at the bottom of a new page, similar to the one at <http://moma.org>

Starting file: <http://jsfiddle.net/fiddlefiddle/uge2Q/>

Finished file: <http://jsfiddle.net/fiddlefiddle/uge2Q/6/>

Static and Fixed Positioning

Relative Positioning

A relative positioned element is positioned relative to its normal position. You use the properties top, right, bottom and left to position an element.

For example, `position:relative; left:-20px;` will set an element 20 pixels to the left of its normal position; it subtracts 20 pixels from its normal left position.

See this in action:

- http://www.w3schools.com/Css/tryit.asp?filename=trycss_position_relative

Absolute Positioning

Absolute Positioning

The position of an absolutely positioned element is determined by its offset values in the properties: top, right, bottom and left.

But, unlike relative positioning, where the offsets are measured relative to its normal position, an absolutely positioned element is offset from its "container block."

- WTF is a "container block"? It's the first parent element that has a position other than static. If no such element is found, the containing block is <html>.

Absolutely positioned elements can overlap other elements.

Unlike a Fixed element, an absolute element will move as you scroll away from it.

See this in action:

<http://alexisgo.com/teaching/class3/simpleAbsPos.html>

http://www.w3schools.com/Css/tryit.asp?filename=trycss_position_absolute

Absolute Positioning

See this in action:

- http://www.w3schools.com/Css/tryit.asp?filename=trycss_position_absolute
- <http://alexisgo.com/teaching/class3/simpleAbsPos.html>

Making our footer stick to the bottom

- We can use the concept of Absolute Positioning to force our footer to be on the bottom of the page.
- Before we do that, we need to add a width to our #footer selector:

```
#footer
{
    background-color: black;
    min-height: 20px;
    padding-top: 10px;
    width: 100%;
}
```

Making our footer stick to the bottom

- We can use the concept of Absolute Positioning to force our footer to be on the bottom of the page.
- Let's set the footer to be absolutely positioned, 0px from the bottom and 0px from the left:

```
#footer
{
    background-color: black;
    min-height: 20px;
    padding-top: 10px;
    width: 100%;

    position: absolute;
    bottom: 0px;
    left: 0px;
}
```

Absolute Positioning Lab

Starting file: <http://jsfiddle.net/fiddlefiddle/s45mk/>

Finished file: <http://jsfiddle.net/fiddlefiddle/s45mk/2/>

If you want all the files instead of the JSFiddles, you can find them here:
<http://bit.ly/GDI3code>

Using CSS in Tumblr

Another HTML tag: blockquote

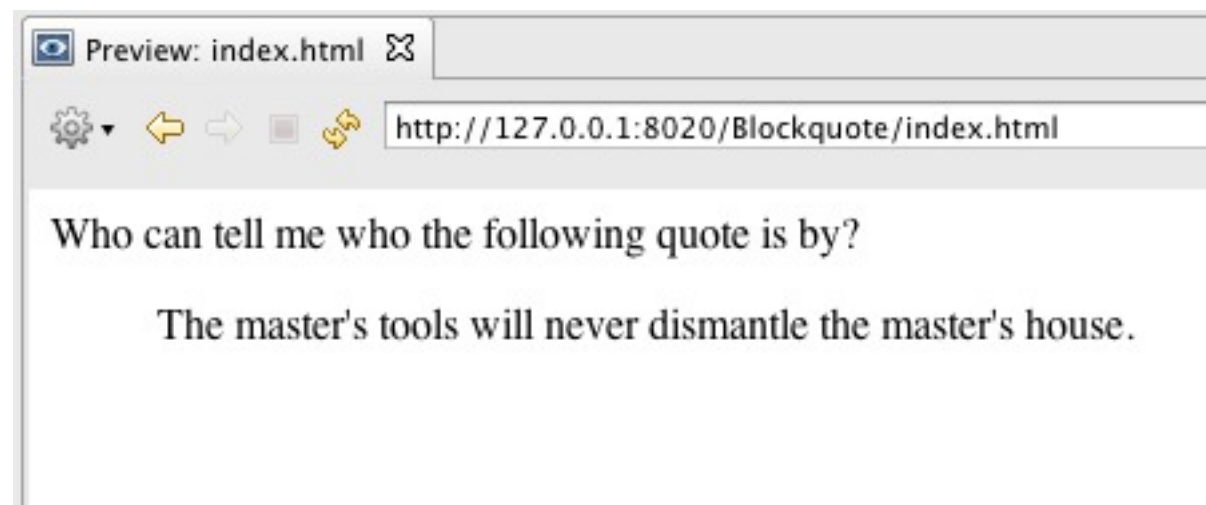
- One HTML tag we have not covered is **blockquote**
- Blockquote's default style is to have the text indented over by one tab:

`<body>`

Who can tell me who the following quote is by?

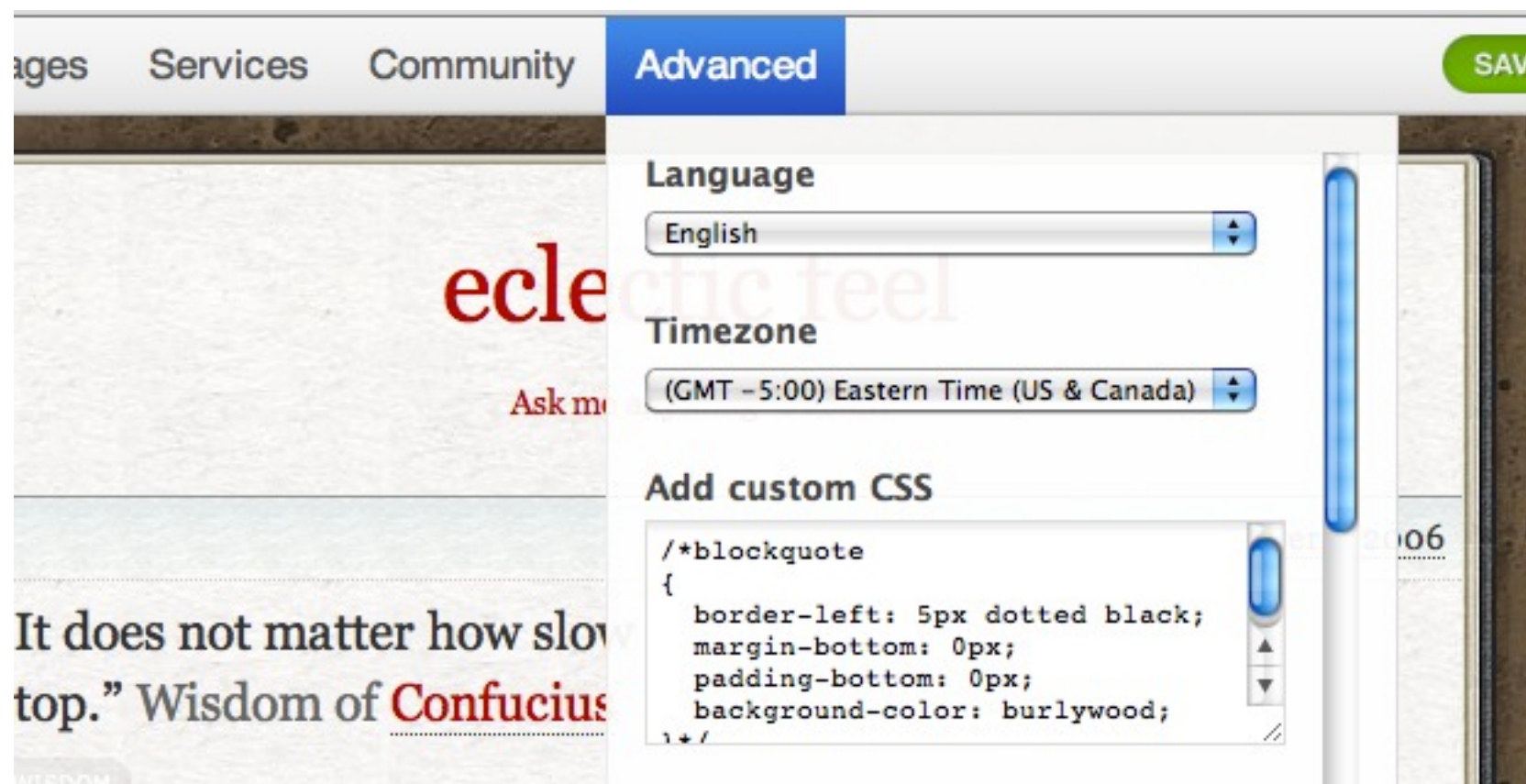
`<blockquote>`The master's tools will never
dismantle the master's house.`</blockquote>`

`</body>`



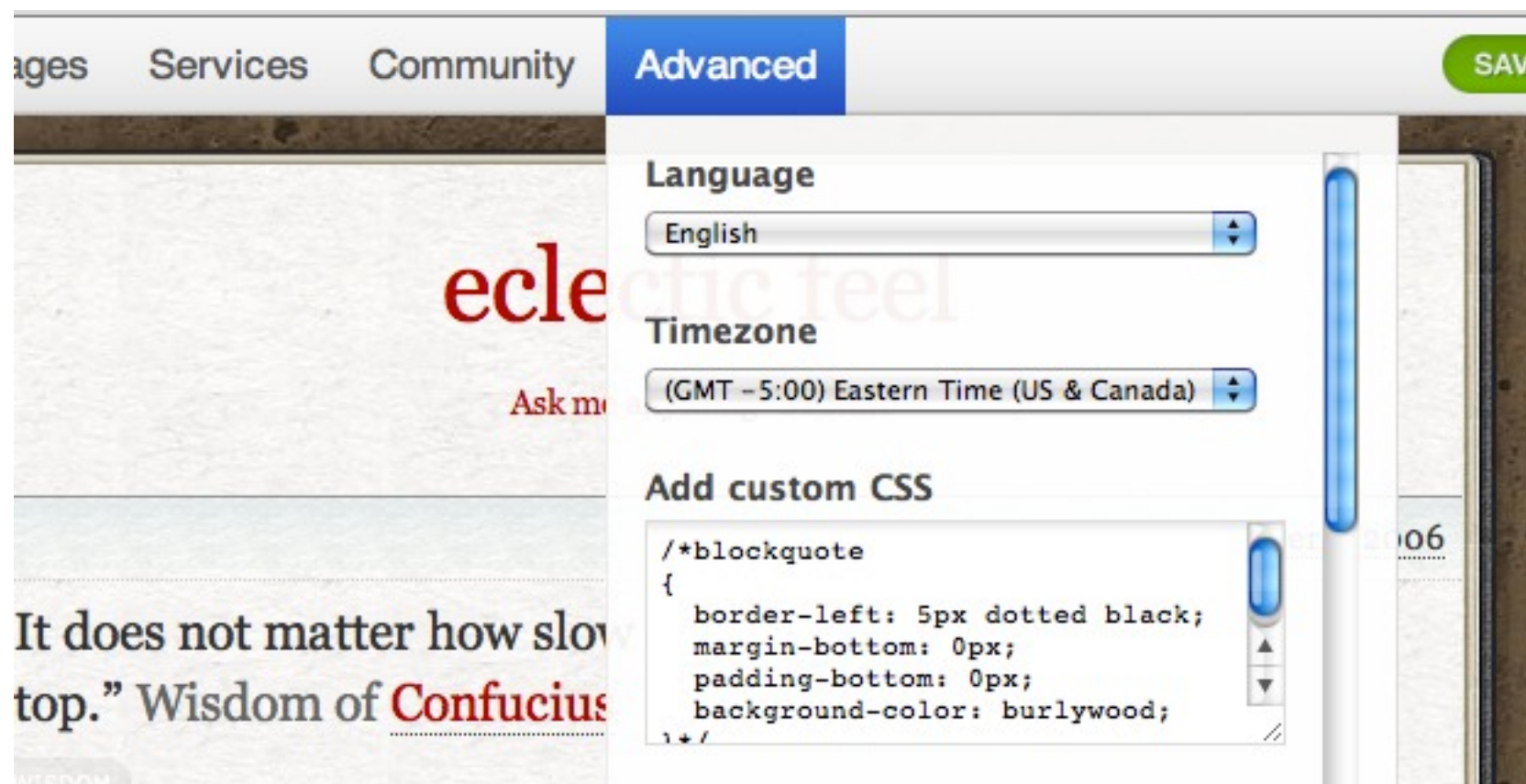
Changing default styles in Tumblr

- To change a CSS element selector's styling in Tumblr, you can go to:
 - Account > Preferences > Customize your Blog
 - Then go to Advanced > Add custom CSS



Changing default styles in Tumblr

- For example, if you don't like the way the **blockquote** elements on your tumblr are styled, you can add a new **blockquote** element selector in this field, and it will override the default.



Changing default styles on Wordpress.org

- You can only change the stylesheet of a wordpress theme if you are using wordpress.org or hosting wordpress on your own site.
- If you are using the free wordpress.com, you have to pay \$15/year to be able to customize the CSS.

Other CSS Concepts

CSS Grouping

- If you find yourself repeating the same styles for two different selectors, we can use a technique called **grouping** to save time.
- With grouping, we apply the same style to two selectors at once:

```
h1, h2
{
    margin: 0px;
}
```

commas allow us to **group** selectors.

When a comma separates two selectors, both selectors will get the style in brackets.

Grouping & Nesting Exercise

- Let's create a simple example that uses:
 - CSS Pseudo-classes
 - CSS Nesting
 - CSS Grouping
- We'll start with this JSFiddle, and modify it: <http://jsfiddle.net/fiddlefiddle/zU6M6/1/>
- A shorter version of the same link: <http://bit.ly/CSSgroup>

Playing with Borders

- Next, let's examine the file in the class4files folder called "borderTricks.html"
- We're going to learn a trick to use borders to make an arrow, for use ultimately in a menuBar, like this:



Playing with Borders

- borderTricks.html is the final file. You can use it as a cheat sheet if you get stuck following along.
- Let's start with a div element in a new html file. Let's give it an id called "multiColor":

- ```
<html>

<head>
 <title>Border Experiments</title>
</head>
<body>

 <div id="multiColor">
 </div>
</body>
</html>
```

# Playing with Borders

- Now, let's set the style for our `<div id="multiColor">` using an internal stylesheet.

```
<head>
 <style>
 #multiColor
 {
 width:100px;
 height:100px;
 border-top:orange solid 20px;
 border-bottom: green solid 20px;
 border-left: red solid 20px;
 border-right: yellow solid 20px;
 }
 </style>
</head>
```



# Playing with Borders

- Notice those diagonal edges where the borders change color? We're going to leverage that later.

```
<head>
 <style>
 #multiColor
 {
 width:100px;
 height:100px;
 border-top:orange solid 20px;
 border-bottom: green solid 20px;
 border-left: red solid 20px;
 border-right: yellow solid 20px;
 }
 </style>
</head>
```





# Playing with Borders

- Now, let's create a new div with a similar style, but make it more narrow.

HTML: `<div id="multiColorNarrow">`  
`</div>`

CSS: `<style>`

```
#multiColorNarrow
{
 width:1px;
 height:100px;
 border-top:orange solid 20px;
 border-bottom: green solid 20px;
 border-left: red solid 20px;
 border-right: yellow solid 20px;
}
```

`</style>`



# Playing with Borders

- Let's make a new div that's even smaller

HTML: `<div id="multiColorSmushed">`  
`</div>`

CSS: `<style>`

```
#multiColorSmushed
{
 width:1px;
 height:0px;
 border-top:orange solid 20px;
 border-bottom: green solid 20px;
 border-left: red solid 20px;
 border-right: yellow solid 20px;
}
</style>
</head>
```



# Playing with Borders

- Let's see what happens if we remove the top border

HTML: `<div id="multiColorArrows">`  
`</div>`

CSS: `<style>`

```
#multiColorArrows
{
 width:1px;
 height:0px;
 /* remove the top border */
 border-top:none;
 border-bottom: green solid 20px;
 border-left: red solid 20px;
 border-right: yellow solid 20px;
}
```

`</style>`



# Playing with Borders

- Now let's play with **transparency** to make JUST a green arrow!

**HTML:** `<div class="greenArrow"> </div>` `<!-- we'll use a class instead of a div so we can reuse this style more than once -->`

**CSS:**

```
.greenArrow
{
 width:1px;
 height:0px;
 border-top:none;
 border-bottom: green solid 20px;

 border-left: transparent solid 20px;
 border-right: transparent solid 20px;
}
```



We will keep the right and left borders, so as to preserve the bottom green arrow, but make them transparent. This will cause us to see ONLY the bottom green border, which will look like a green arrow!

# Playing with Borders

- Now let's make a green line we can put under our green arrow

HTML: `<div class="navBar"></div>`

CSS:

`.navBar`

{

`width: 600px;`

`margin: 0 auto;`

`border: green solid 5px;`

}



# Playing with Borders

```
<div class="menuBar">
 About
<div class="greenArrow"></div>
<div class="navBar"></div>
</div>
```



About

# Playing with Borders

- Let's add a menu bar with an “about” item

HTML: `<div class="menuBar">About</div>`

CSS:

`.menuBar`

```
{
 background-color: #99C299;
 margin: 0 auto;
 width: 600px;
 color: white;
 font-weight: bold;
 padding: 5px 0px;
}
```



About

# Playing with Borders

```
<html>
 <head>
 <title>Border Experiments</title>
 <style>
 .greenArrow
 {
 width:1px;
 height:0px;
 border-top:none;
 border-bottom: green solid 20px;
 border-left: transparent solid 20px;
 border-right: transparent solid 20px;

 }
 .navBar
 {
 width: 600px;
 margin: 0 auto;
 border: green solid 5px;
 }
 .menuBar
 {
 background-color: #99C299;
 margin: 0 auto;
 width: 600px;
 color: white;
 padding: 5px 0px;
 }
 </style>
 </head>
```



# Playing with Borders

```
<body>
 <div class="menuBar">About
 <div class="greenArrow">
 </div>
 <div class="navBar"></div>
 </div>
</body>
</html>
```

# Playing with Borders

- Add it all together!

HTML: `<div class="menuBar">About  
    <div class="greenArrow"></div>  
    <div class="navBar"></div>  
</div>`

We have all the CSS we need already defined!

