

# Web Development Fundamentals

Some Link Formatting

CSS Selectors –

CSS Divs & Spans

Lecturer: Marie Brennan




# recap

- Basic html page layout : HTML-HEAD-BODY
- Adding some formatting with CSS
- HTML-HEAD - **style** - BODY

`<style> @import "style.css"; </style>`

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

- So far in css:
  - Font-style
  - Font-size
  - Font-family
  - Font-variant
  - Font-color



If the file is saved in a css folder then alter the path here to `css/style.css`



# Recap – font size

	<code>body { font-size: 100%; }</code>	<code>body { font-size: 120%; }</code>
<code>font-size: 1em</code>	The quick brown fox	The quick brown
<code>font-size: 12pt</code>	The quick brown fox	The quick brown fox
<code>font-size: 16px</code>	The quick brown fox	The quick brown fox
<code>font-size: 100%</code>	The quick brown fox	The quick brown

The percent unit seems to provide a more consistent and accessible display for users....but try and test all in the labs

# backgrounds

body

{

background-color: blue;

}



# Using an image as a background

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-image: url("paper.gif");
}
</style>
</head>
<body>





<h1>Hello World!</h1>


<p>This page has an image as the background!</p>

</body>
</html>
```

## Hello World!

This page has an image as the background!

 background images for websites   

 [Sign in](#)

All Images News Videos Maps More Settings Tools

View saved SafeSearch

- simple
- car
- professional
- web page
- house
- person
- business
- creative
- nature
- love
- wallpaper
- seamless
- hd wallpaper
- website design
- web development
- white p...



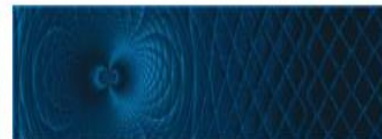
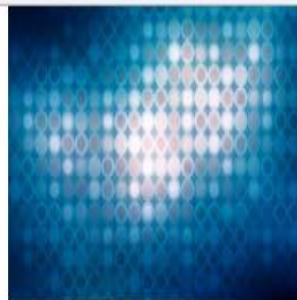
## Ahrx Wallpapers Background Wallpaper Desi...

fansshare.com - 1440 × 810 - Search by image

Shared image titled Ahrx Wallpapers Background Wallpaper Design at 1440x810 uploaded by rees39. 16624750

-  Visit
-  View image
-  Save
-  View saved
-  Share





- Open link in new tab
- Open link in new window
- Open link in incognito window
- Save link as...
- Copy link address
- Open image in new tab
- Save image as...
- Copy image
- Copy image address
- Search Google for image

Inspect

Ctrl+Shift+I

## background header images for websites | Sli...

Slide Background Edit - 2475 × 583 - Search by image

Background Images For Websites Header Clipartsgam Com

Visit

View image

Save

View saved

Share

### Related images:



View more

Body

```
{  
background-color:blue;  
background-image: url("wall1.jpg");  
}
```







```
body{  
background-color:blue;  
background-image: url("wall1.jpg");  
}
```

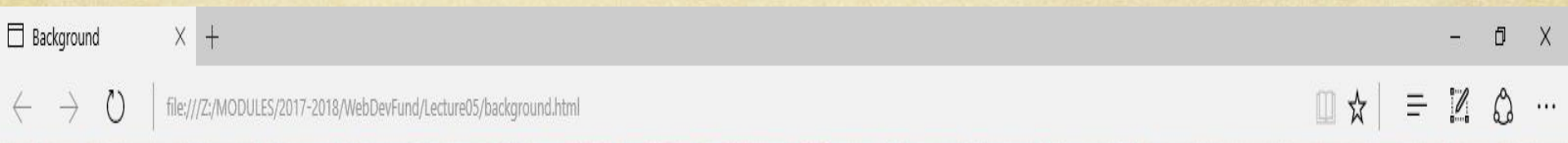
h1



```
{  
font-family: sans-serif;  
font-size: 3em;  
color: white;  
border-bottom: 1px dashed #57b1dc;  
margin-top: 30px;  
text-align: center;  
}
```

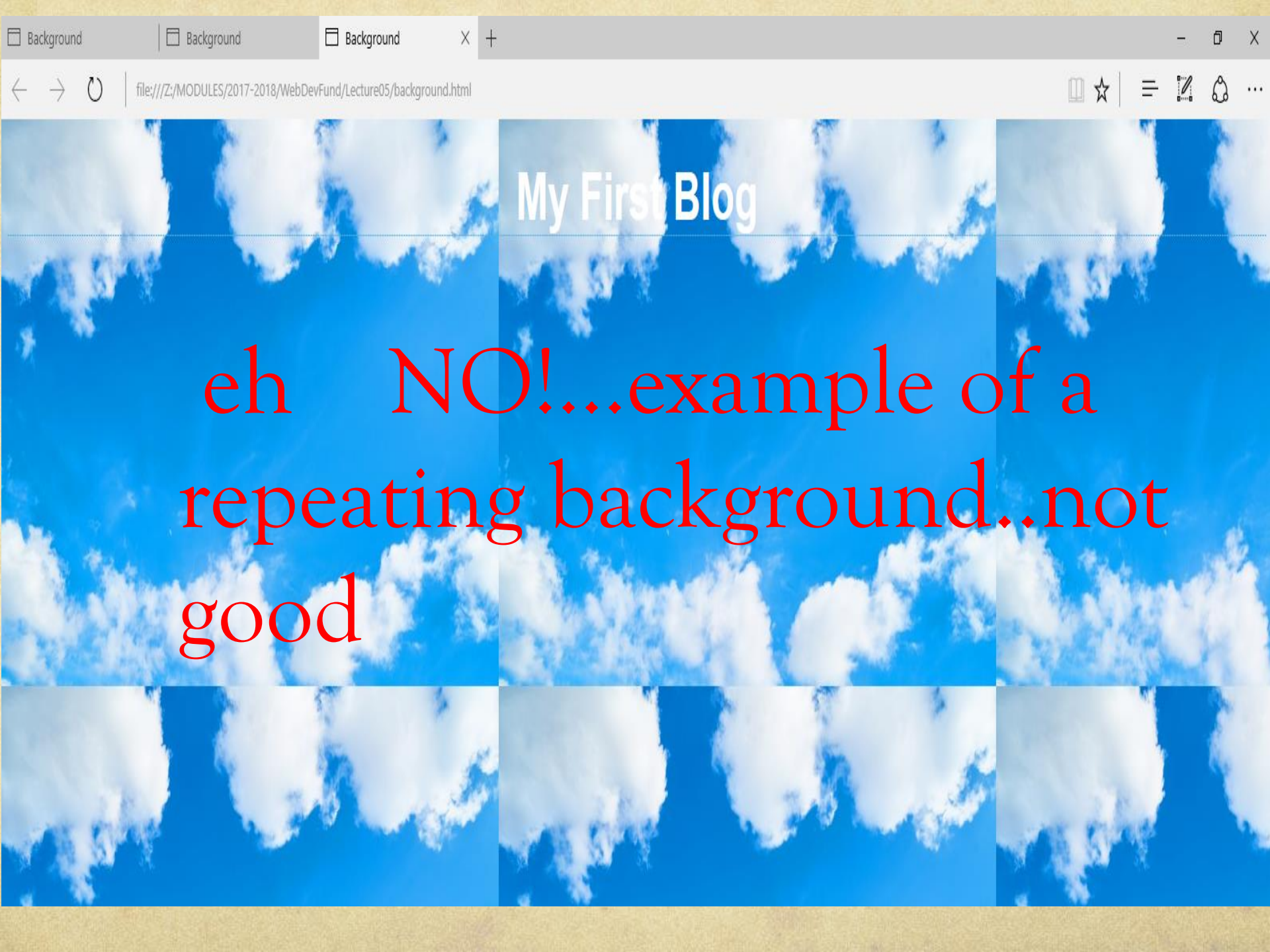
Lets add a h1 and see what it looks like with this background...formatted in css as follows. This is similar to code we used in this weeks lab





# My First Blog

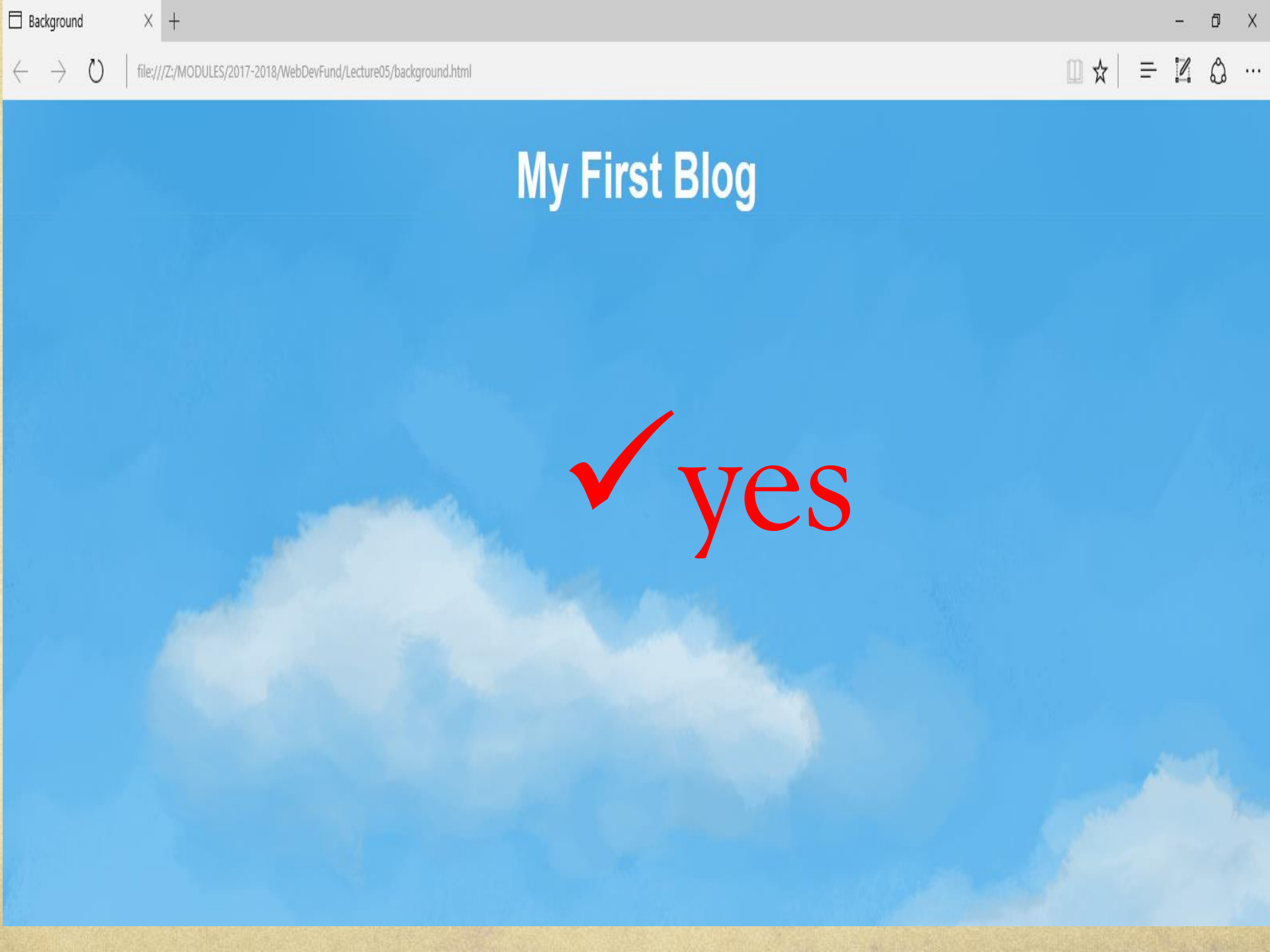
---



My First Blog

eh NO!...example of a  
repeating background..not  
good





# My First Blog

✓ yes

# Formatting links in CSS

In the syle sheet:     a { color:pink;}

```
<!DOCTYPE html>
<html>
<head>
<style>
  @import "style.css";

</style>
</head>
<body>

<p><b><a href="pink.html" target="_blank">This is a link</a></b></p>

</body>
</html>
```

This is a link



## Change the colour of the link once the page has been visited

```
/* unvisited link */
a:link {
    color: red;
}

/* visited link */
a:visited {
    color: green;
}

/* mouse over link */
a:hover {
    color: hotpink;
}

/* selected link */
a:active {
    color: blue;
}
```

[This is a link](#)

**Note:** a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective.

**Note:** a:active MUST come after a:hover in the CSS definition in order to be effective.

## Remove the underline from the link

In the css

page

```
a:link {  
    text-decoration: none;  
}  
  
a:visited {  
    text-decoration: none;  
}  
  
a:hover {  
    text-decoration: underline;  
}  
  
a:active {  
    text-decoration: underline;  
}
```

In the html

page .....

```
<p><b><a href="default.asp" target="_blank">This is a link</a></b></p>  
<p><b>Note:</b> a:hover MUST come after a:link and a:visited in the CSS definition  
in order to be effective.</p>  
<p><b>Note:</b> a:active MUST come after a:hover in the CSS definition in order to  
be effective.</p>  
  
</body>  
</html>
```

[This is a link](#)

**Note:** a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective.

**Note:** a:active MUST come after a:hover in the CSS definition in order to be effective.



## Change the background colour of the link...if you wish

```
a:link {  
    background-color: yellow;  
}  
  
a:visited {  
    background-color: cyan;  
}  
  
a:hover {  
    background-color: lightgreen;  
}  
  
a:active {  
    background-color: hotpink;  
}
```

[This is a link](#)

**Note:** a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective.

**Note:** a:active MUST come after a:hover in the CSS definition in order to be effective.

# Image as a link

```
<!DOCTYPE html>
<html>
<body>

<p>
An image that is a link:
<a href="http://www.itb.ie">

</a>
</p>

</body>
</html>
```

An image that is a link:



.....- remove the border

```
<!DOCTYPE html>
<html>
<body>

<p>
An image that is a link:
<a href="http://www.itb.ie">

</a>
</p>

</body>
</html>
```

An image that is a link:





# CSS Selectors

- Last week we looked at using CSS rules to format text.

- We saw that each rule consists of:

```
selector {  
    declaration(s);  
}
```

- We have been using element names as selectors e.g.

```
p {font-size: small; }  
h1 {color: red; }
```

- Disadvantage of this is that the property values specified in the declaration will apply to **all** occurrences of the element.
- So, *all* paragraph text is small, *all* **h1** headings red.

# Class Selectors

- What if we want to make just one heading red?
- Or make one paragraph in small font, the others in medium?
- There are a number of different selector types that allow us to do exactly this.
- We will look at two of these:
  1. **Class** selectors
  2. **Descendant** selectors
  3. Child Selectors



**<li class = "irish"> Irish </li> (HTML)**

## **Class**

- full stops

(css)

li.irish {

color: green;

}

**Same as  
before**

# Class

- The *class selector*, can be used to group *similar* elements together  
.....and style them accordingly
- This means we can apply one style rule to multiple elements of the same class  
.....Only those elements declared as being in that class

## ○ Examples in html

`<li class = "irish"> Irish </li>`

**We can apply the**

``



# Class

- To construct a style rule for **img** elements of the **hughimages** class do something like this:

```
img.catImages  
    {  
    border: none;  
    }
```

```
li.irish {  
    color: green;  
}
```

- Dot (.) symbol used to indicate class name in CSS rule selector

## Class - Example

- Suppose for example we are building a website for a shop that sells **books**, **CDs** and **DVDs**.
- The entire catalogue appears on one page but the client wants the three categories of item to look different in some way.
- Solve this by defining three classes – CD, book and DVD.  
.....Then for each item, specify which class it belongs to.
- See example on next slide.



# Class

The books will be green,  
the cd's will be navy and  
the dvd's will be orange

The HTML fragment

```
<ul>  
  <li class="book">No Country For Old Men</li>  
  <li class="book">The Gathering</li>  
  <li class="cd">Ok Computer</li>  
  <li class="book">The Mission Song</li>  
  <li class="dvd">Seinfeld Season 1</li>  
</ul>
```

The CSS to style each class of list item differently

```
li.book { color: green; }  
li.cd { color: navy; }  
li.dvd { color: orange; }
```

## Lecture 6 Example 2 - Mozilla Firefox

File Edit View History Bookmarks Tools Help

```
li.book { color: green; }      /* books in GREEN */  
li.cd { color: navy; }        /* books in NAVY BLUE */  
li.dvd { color: orange; }     /* books in ORANGE */
```

- ◆ No Country For Old Men
- ◆ The Gathering
- ◆ Ok Computer
- ◆ The Mission Song
- ◆ Seinfeld Season 1



# Class example 1

```
<!DOCTYPE html>
<html><head><style> @import "style.css";</style>
</head><body>
<h1>Welcome to My Homepage</h1>
<div class="intro">
  <p>My name is Marie.</p>
  <p>I live in New York.</p>
</div>
<p>My best friend is Donald.</p>
</body>
</html>
```

```
.intro
{
  background-color: yellow;
}
```

## Welcome to My Homepage

My name is Marie.

I live in New York.

My best friend is Donald.



# Class example 2

```
<!DOCTYPE html>
<html>
<head>
<style>
@import "class2.css";
</style>
</head>
<body>
<p>My name is Donald.</p>
<p class="hometown">I live in Ducksburg.</p>
<p>My name is Dolly.</p>
<p class="hometown">I also live in Ducksburg.</p>
</body>
</html>
```

declare the class

Format appropriately in css using the dot.

```
.hometown
{
background-color: yellow;
}
```

My name is Donald.

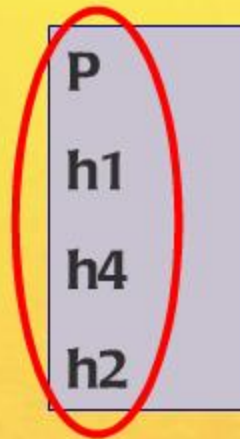
I live in Ducksburg.

My name is Dolly.

I also live in Ducksburg.

## CLASS .classname

- used to specify a style for different elements.
- allows you to set a particular style for any html elements with the same class.
- uses the html class attribute, and is defined with a " " .



## CLASS (example)

```
<html>
  <head>
    <style type="text/css">
      .center{text-align:center;}
    </style>
  </head>

  <body>
    <h1 class="center">center-aligned heading</h1>
    <p class="center">center-aligned paragraph.</p>
  </body>
</html>
```

**center-aligned heading**

center-aligned paragraph.



## **Anonymous Class**

– full stops

```
.irish {  
color: green;  
}
```

# The “anonymous class”

- You may want to apply a rule to *all* elements of class regardless of what type of element they are
- For example suppose we have list item elements specified as being of class “book” i.e.

```
<li class="book">The Mission Song</li>
```

- But in other places on the page also have anchor (a) elements specified as being of class book i.e.

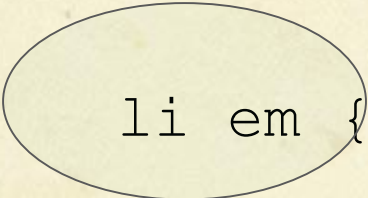
```
<a href=".." class="book">The Gathering</a>
```

- The following style rule will colour both red:  
**.book { color: red; }**



## Descendant Selectors

– SPACEs



```
li em {
```

```
color: red;
```

```
}
```



# Descendant Selectors

- `ul li a:hover`

```
{  
background-color: green;  
}
```

- A *descendent selector* targets elements that are contained *within* (and therefore descendants of) another element.
- They are indicated by a list of element types separated by spaces.
- For example to target **em** elements that appear inside list items we would use:

```
li em { color: olive; }
```

# Descendant Selectors

## How to 'read' them like a browser ...

- Read this:

```
dt strong {color: maroon; }
```

- As:

- for "strong" elements found inside "dt" elements  
make text colour 'maroon'

- (you do this one !) Read this:

```
li em { color: olive; }
```

- As:

For emphasised text within a  
list item make the text color  
olive

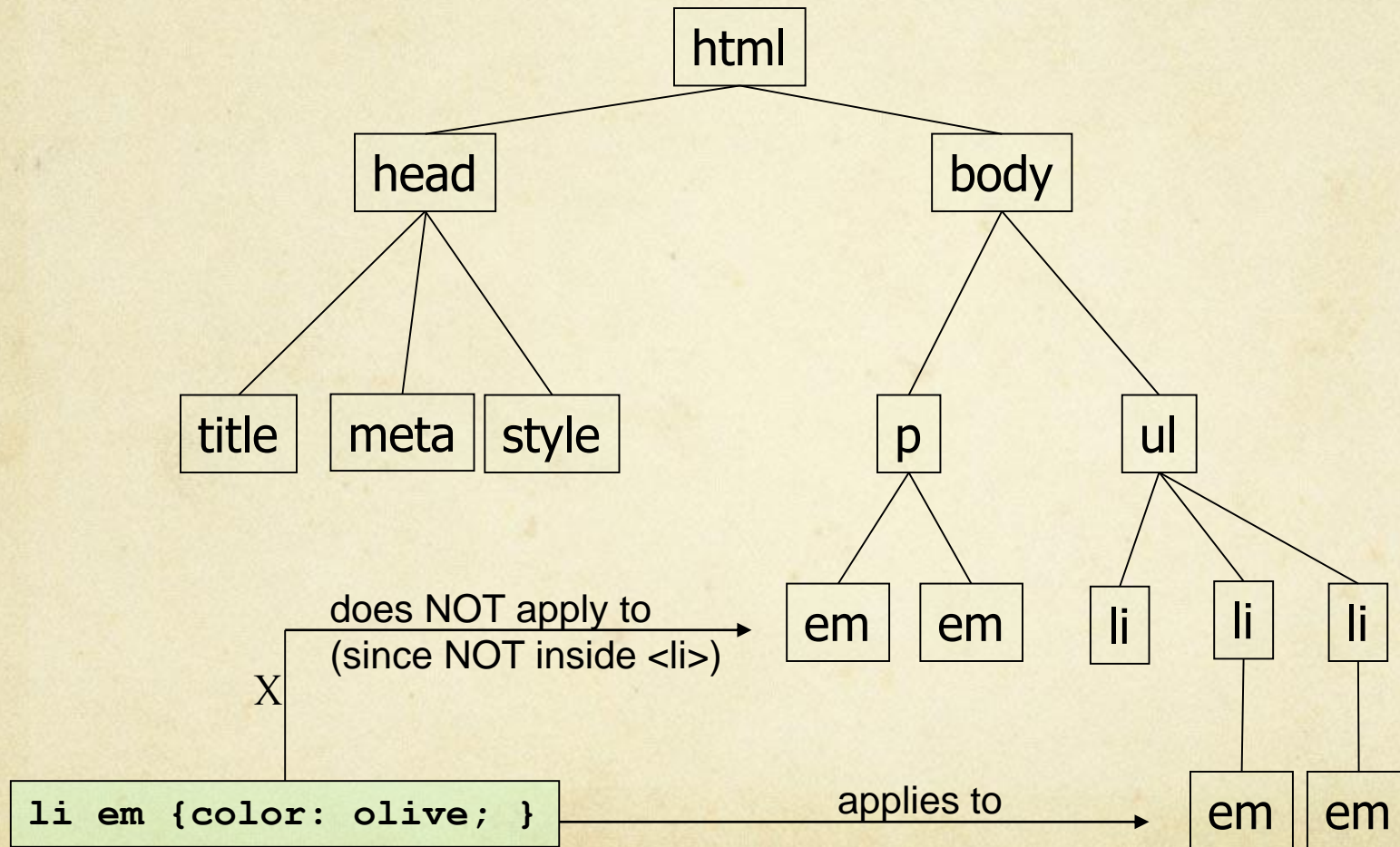


# Descendant Selectors - Example

```
/* descend.css */  
li em {color: olive; }
```

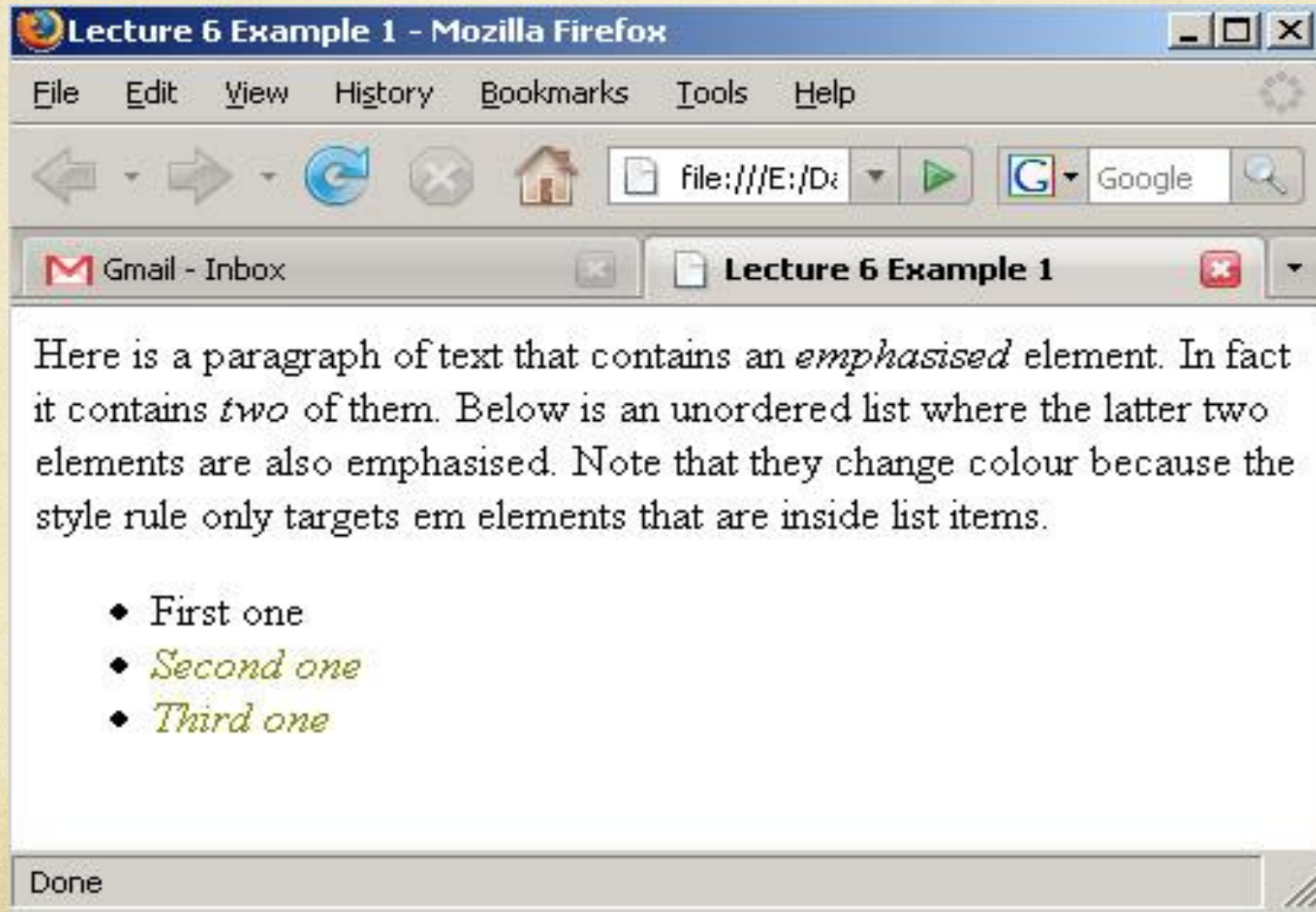
```
<!DOCTYPE html><html lang="en">  
<head><title>Lecture 6 Example 1</title>  
<style type="text/css">  
    @import "descend.css";  
</style></head><body>  
<p>Here is a paragraph of text that contains an  
<em>emphasised</em> element. In fact it contains  
<em>two</em> of them. Below is an unordered list where the  
latter two elements are also emphasised. Note that they  
change colour because the style rule only targets em  
elements that are inside list items.</p>  
<ul>  
    <li>First one</li>  
    <li><em>Second one</em></li>  
    <li><em>Third one</em></li>  
</ul></body></html>
```

# Descendant Selectors - Example





# Descendant Selectors - Example





# Descendant Selectors

- We can also have descendent selectors several layers deep.
- For example:

```
p a strong { font-variant: small-caps; }
```

- This selector targets **strong** elements that appear inside **a** elements that in turn appear inside **p** elements.

# **Descendant Selectors**

## **How to 'read' them like a browser ...**

- (you do this one !) Read this:

```
footer li strong { background-color: blue; }
```

For strong elements in a list item  
within the footer of my page Make the  
background blue



## Child selectors

- the greater-than sign

```
li>strong {  
background-color: blue;  
}
```



# > indicates a CSS 'Child' selector must be direct child of parent

- Must be direct child

```
li>strong { background-color: blue; }
```

CSS

- Strong elements that are direct children of <li>

```
<li> I am a <strong>child of li</strong></li>
```

HTML

- Would not target <strong> inside a <p> in an <li>, e.g.

```
<li>
```

```
<p>
```

```
I am a <strong>child of p</strong>
```

```
</p>
```

```
</li>
```

## reCap

- Class: style all elements of a class

- Selector with dot .

e.g.

**.book { . . . }**

- Descendent selectors (a child, or grandchild etc.)

- Selectors with spaces:

e.g.

**nav a { . . . }**



# **Element IDs**



# Class

- **ID** attribute similar to **class** attribute
- Key difference between ID and CLASS
  - can be MULTIPLE elements in a page of the same class
  - Can only be **ONE** element with a given **ID**

```

```

# ID

- Can use **id** attribute to define unique ID for an element
- Value of **id** must be used only once in a page,
  - Any element may have an ID
- For example:  

```
<li id = "isbn200453">The Gathering</li>
```
- Uniquely identifies that list item
- We can use more or less anything we want as values of the **id** attribute
  - LETTERS/numbers
  - Case sensitive – so always use lower case



# ID

- Can construct CSS style rule to target that element
- Use hash (#) symbol as follows:

```
li #isbn200453 { color: red; }
```

- This will make any list items with the **id** value of ISBN200453 (there should be only one), turn red.
- Because these id value are unique we can omit the element name e.g.

```
#isbn200453 { color: red; }
```


- Hash (**#**) symbol used to indicate ID value in CSS rule selector



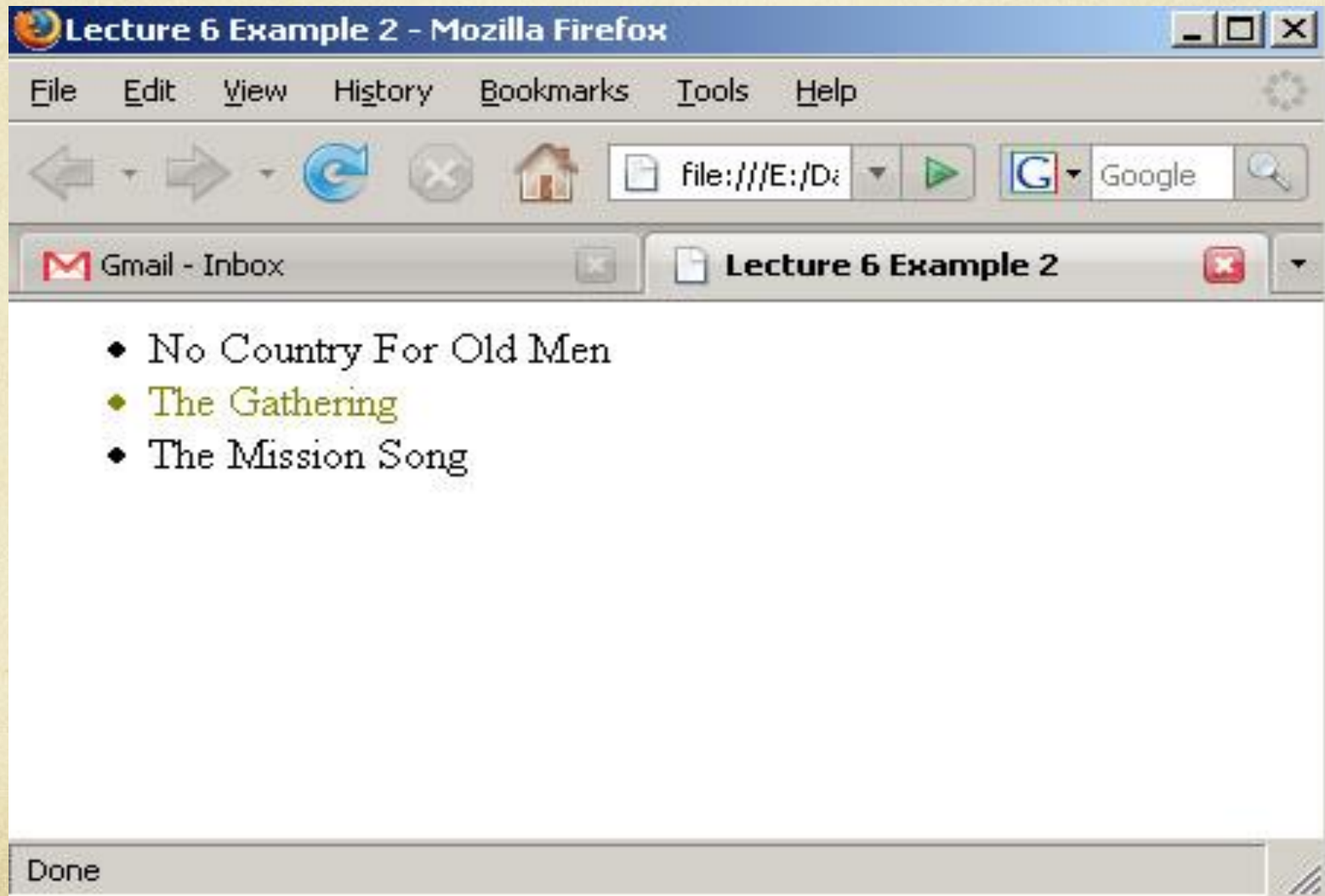
# ID Selectors - Example

```
<html>
<head>
<title>Lecture 6 Example 2</title>
<style type="text/css">
#isbn200453 {color: olive; }
</style>
</head>

<body>
<ul>
    <li id="isbn200452">No Country For Old Men</li>
    <li id="isbn200453">The Gathering</li>
    <li id="isbn200454">The Mission Song</li>
</ul>
</body>
</html>
```



# ID - Example





# Selector types and examples

1. Tags

2. ID

3. Class

# ID and an example

```
<!DOCTYPE html>
```

```
<html><head><style> @import "style.css"</style>
```

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

```
<h1>Welcome to My Homepage</h1>
```

```
<div class="intro">
```

```
<p id="firstname">My name is Marie.</p>
```

```
<p id="hometown">I live in New York.</p>
```

```
</div>
```

```
<p>My best friend is Donald.</p>
```

```
</body>
```

```
</html>
```

CSS: style.css

```
#firstname {  
    background-color: yellow;  
}
```

```
#hometown{  
    background-color: pink;  
}
```

HTML



# Welcome to My Homepage

My name is Marie.

I live in New York.

My best friend is Donald.

## ID #id

- used to specify a style for a single, unique element.
- The id selector uses the id attribute of the HTML element, and is defined with a "#".

```
<html>
  <head>
    <style type="text/css">
      #para1
      {text-align:center; color:red;}
    </style>
  </head>

  <body>
    <p id="para1">Hello World!</p>
    <p>This paragraph is not affected.</p>
  </body>
</html>
```



Hello World!

This paragraph is not affected.



# Selectors

- Select elements to apply a declared style.
- Selector types:
  - Element Selectors: selects **all** elements of a specific **type** (<body>, <h1>, <p>, *etc.*)
  - Class Selectors: selects **all** elements that belong to a given **class**.
  - ID Selectors: selects a **single** element that's been given a **unique id**.
  - Pseudo Selectors: combines a selector with a **user activated state** (:hover, :link, :visited)

# CSS Rule Structure

A CSS RULE is made up of a selector and a declaration. A declaration consists of property and value.

```
selector {property: value;}
```



declaration



# Selectors : RECAP

A selector, here in **green**, is often an element of HTML.

```
body { property: value; }  
h1 { property: value; }  
em { property: value; }  
p { property: value; }
```

## Selectors : RECAP

# Properties and Values

```
body {background: purple;}  
h1 {color: green; }  
h2 {font-size: large;}  
p {color: #ff0000;} /*hexadecimal for red*/
```

\*The CSS code can be written in a linear format (above) or in a block format (below)....

Properties and values tell an HTML element how to display.

```
body {  
background: purple;  
color: green;  
}
```



## Selectors : RECAP

# Grouping Selectors

Group **the same selector** with different declarations together on one line.

```
h1 {color: black;}  
h1 {font-weight: bold;}  
h1 {background: white;}
```

*Example of grouping selectors (both are correct):*

```
h1 {  
  color: black;  
  font-weight: bold;  
  background: white;  
}
```

## Selectors : RECAP

Group **different selectors** with the same declaration on one line.

# Grouping Selectors

```
h1 {color: yellow; }
```

```
h2 {color: yellow; }
```

```
h3 {color: yellow; }
```

*Example of grouping selectors (both are correct):*

```
h1, h2, h3 {color: yellow; }
```



# NB - Comments in CSS (recap)

- Explain the purpose of the coding
- Help others read and understand the code
- Serve as a reminder to you for what it all means
- Starts with `/*` and ends with `*/`

```
p {color: #ff0000;} /*Company Branding*/
```

Lets just quickly go back to CSS  
and our links again.....



```
<!DOCTYPE html>
<html>
<head>
<style>
a:link, a:visited {
    background-color: blue;
    color: white;
    padding: 14px 25px;
    text-align: center;
    text-decoration: none;
    display: inline-block;
}

a:hover, a:active {
    background-color: red;
}
</style>
</head>
<body>

<a href="default.asp" target="_blank">This is a link</a>

</body>
</html>
```



Document... x

cssExamples.html

cssExamples.html

```
<!DOCTYPE html>
<html>
<head>
<style>
a.ex1:hover, a.ex1:active {color: red;}
a.ex2:hover, a.ex2:active {font-size: 150%;}
a.ex3:hover, a.ex3:active {background: red;}
a.ex4:hover, a.ex4:active {font-family: monospace;}
a.ex5:visited, a.ex5:link {text-decoration: none;}
a.ex5:hover, a.ex5:active {text-decoration: underline;}
</style>
</head>
<body>

<p>Mouse over the links to see them change layout.</p>

<p><a class="ex1" href="page1.html">This link changes color</a></p>
<p><a class="ex2" href="page1.html">This link changes font-size</a></p>
<p><a class="ex3" href="page1.html">This link changes background-color</a></p>
<p><a class="ex4" href="page1.html">This link changes font-family</a></p>
<p><a class="ex5" href="page1.html">This link changes text-decoration</a></p>

</body>
</html>
```

[This link changes color](#)

[This link changes font-size](#)

[This link changes background-color](#)

[This link changes font-family](#)

[This link changes text-decoration](#)

Mouse over the links to see them change layout.

[his link changes color](#)

[This link changes font-size](#)

[his link changes background-color](#)

[his link changes font-family](#)

[his link changes text-decoration](#)



# Span

- A `<span>` element used to color a part of a text:

```
<!DOCTYPE html>
<html>
<head>
<style>
@import "span.css";
</style>
</head>
<body>
<p>My mother has <span class = "blue">blue</span> eyes and my father has <span class ="green">dark green</span> eyes.</p>
</body>
</html>
```

HTML

CSS

```
.blue
{
color:blue;
font-weight:bold;
}

.green
{
color:darkolivegreen;
font-weight:bold;
}
```

browser

My mother has **blue** eyes and my father has **dark green** eyes.

W3c examples of all the selector types ...

<http://www.w3.org/TR/CSS2/selector.html#pattern-matching>



# Recap.....Separating Content

# <div> Element tag

- Useful for dividing parts of the page into sections.
- Creates a “box” with the following attributes:

- margin
- padding
- border
- height
- width
- (..and lots more)

We will be covering these in more detail in the box model

- Primary element used for CSS Layouts  
(more information in CSS Layouts tutorial)



# HTML semantic page elements

## The magnificent 7 elements

- Main 5 useful new elements:
  - header
  - nav
  - footer
  - aside
  - section
- Also (less useful), remaining 2 new elements:
  - article
  - address

<header>

<nav>

<section>

<article>

<aside>

<footer>



header	Defines a header for a document or a section
nav	Defines a container for navigation links
section	Defines a section in a document
article	Defines an independent self-contained article
aside	Defines content aside from the content (like a sidebar)
footer	Defines a footer for a document or a section
details	Defines additional details
summary	Defines a heading for the details element

# HTML semantic page elements

- These are BLOCK LEVEL elements
- They have no inherent presentational qualities
  - browser will not do anything special with a section of code marked up with such elements
  - unless you tell it to using a style sheet



# HTML semantic page elements

- If these not are enough, you also have generic
  - **div**
- In fact the page blocks (header, footer, nav etc.) took the most used DIV IDs and turned them into new elements in HTML 5
  - So instead of  
`<div id="nav">`      `<div id="header">` etc.  
we now just write  
`<nav>`                      `<header>`                      etc.

## **Div Tags Defined:**

The `<div>` tag defines a division or a section in an HTML document.

The `<div>` tag is used to group block-elements to format them with styles.

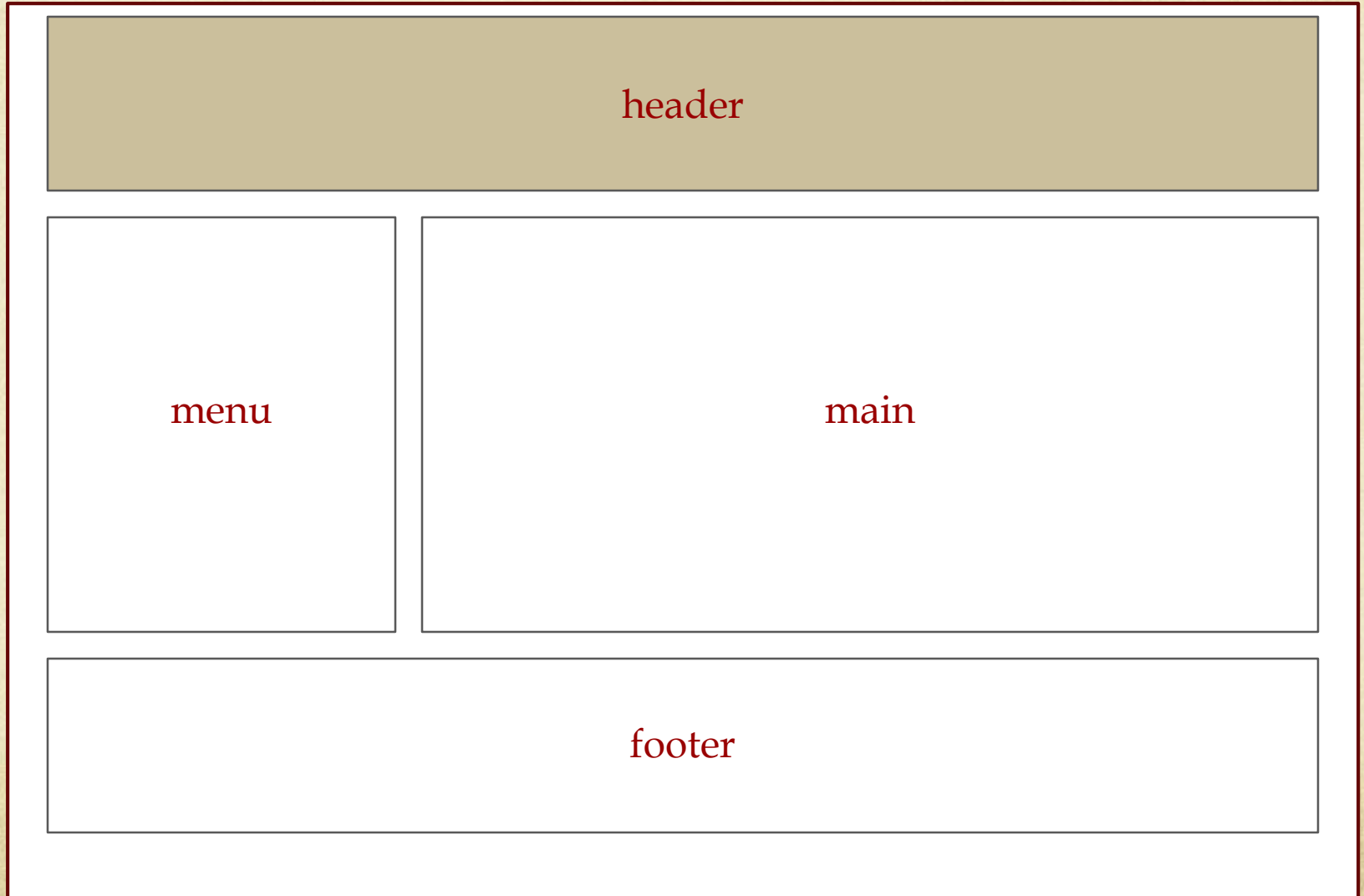
### **Tip:**

The `<div>` element is very often used together with CSS, to layout a web page.



# Typical Web Page (Browser)

Container



# Typical Web Page (HTML)

Typical HTML Web page is made up of containers (boxes) or DIVs.

Each DIV is assigned an ID or a Class, ID if they are unique and only used once...when you run out of html5 sections!

```
<div id="container">
```

```
  <header>Insert Title</header>
```

```
  <aside></aside>
```

```
  <article></article>
```

```
  <content></content>
```

```
  <footer></footer>
```

```
</div>
```



# HTML Creating your own divs

```
</div>
```

```
<div id = "menu">
```

```
</div>
```

You can create another section and give it an id and format that in css

## CSS

```
#menu  
{  
  
}
```

# Typical Web Page (CSS)

The CSS file uses the same DIV/ID/Class names as the HTML and uses them to style the elements.

```
#container {property: value;}  
#menu {property: value;}  
#main {property: value;}  
footer {property: value;}
```



# IDs and Classes – recap

- IDs (#) are unique and can only be used once on the page
- Classes (.) can be used as many times as needed

## *HTML Code:*

```
<h1 id="mainHeading">Names</h1>
```

```
<p class="name">Joe</p>
```

## *CSS Code:*

```
#mainHeading {color: green}
```

```
.name {color: red}
```



# HTML semantic page elements

- These are BLOCK LEVEL elements
- They have no inherent presentational qualities
  - browser will not do anything special with a section of code marked up with such elements
  - unless you tell it to using a style sheet

# HTML semantic page elements

- If these not are enough, you also have generic
  - **div**
- In fact the page blocks (header, footer, nav etc.) took the most used DIV IDs and turned them into new elements in HTML 5
  - So instead of  
`<div id="nav">`      `<div id="header">` etc.  
we now just write  
`<nav>`                      `<header>`                      etc.



## Generic Element – **div**

- There are two generic elements, both mark up sections of HTML code
  - **div** : for BLOCK LEVEL elements
  - **span** : for IN-LINE elements
- DIV has no inherent presentational qualities
  - unless you tell it to do so using a style sheet
  - Just like header, footer, nav etc.



# Headers and footers

## <header>

- Headers
  - For introductory material about
    - Page
    - Section/article in a page

## <footer>

- Footers
  - For information typically at the end of
    - Page
    - Section/article in a page
  - E.g. author details / copyright etc.

**<header>**

```
<h1>More about WOFF</h1>
```

```
<p>by Jennifer Robbins, <time datetime="11-11-2011"
pubdate>November 11, 2011</time></p>
```

**</header>**

<section>

<p>...content goes here...</p>

</section>

**<footer>**

```
<p><small>Copyright &copy;2012 Jennifer
Robbins.</small></p>
```

```
<nav>
```

```
<ul>
```

```
<li><a href="">Previous</a></li>
```

```
<li><a href="">Next</a></li>
```

```
</ul>
```

```
</nav>
```

**</footer>**

# <div>

- Tip: The <div> element is very often used together with CSS, to layout a web page.
- By default, browsers always place a line break before and after the <div> element.
- However, this can be changed with CSS.



# **Summary and Conclusions**

## Conclusion: Page content

- HTML 5 page content – the most useful 5 block elements:
  - header
  - footer
  - nav
  - section
  - aside
  
- And DIV when the others not enough ...we will cover more divs and spans later...but before all of this we need to understand.....



## Introduction – to FLOATs

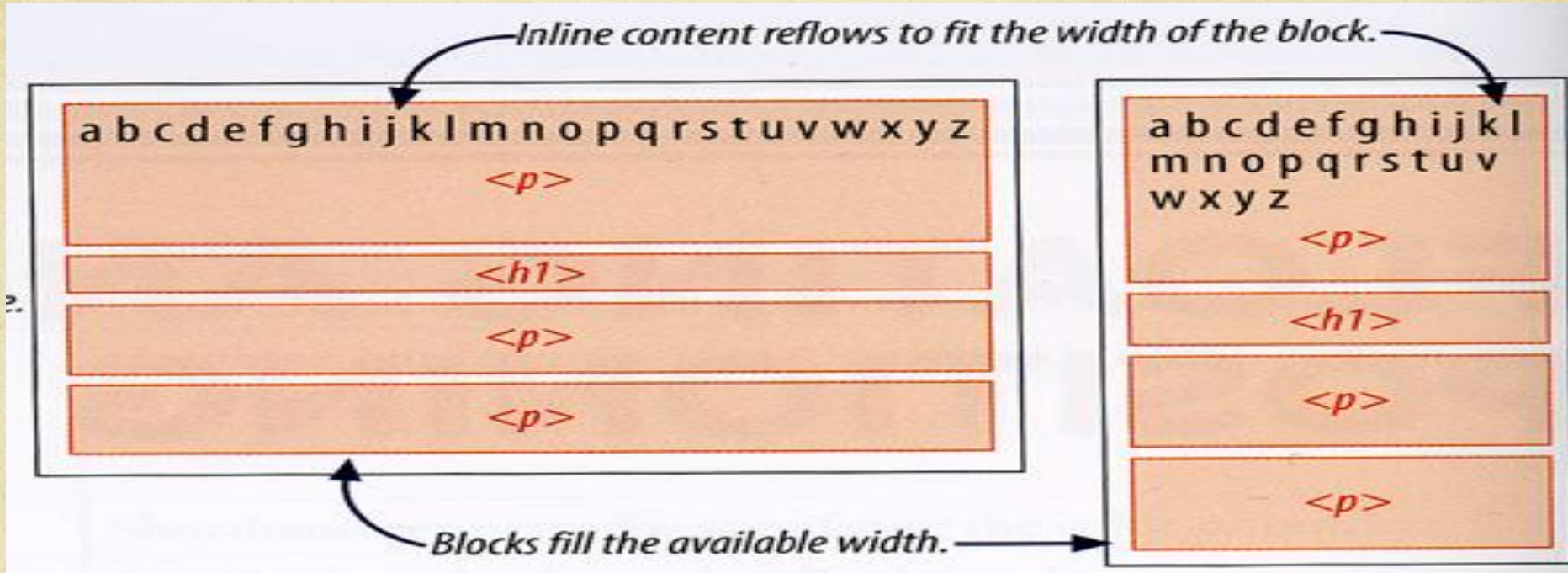
- In this week's class we are going to learn about how to use “floats” to control where boxes appear within the browser window...for now
- Floating – introduction
- Clearing floats
- Multi-column page layouts using Floats
- We'll start by reviewing how elements behave with normal flow.



## “Normal” Flow

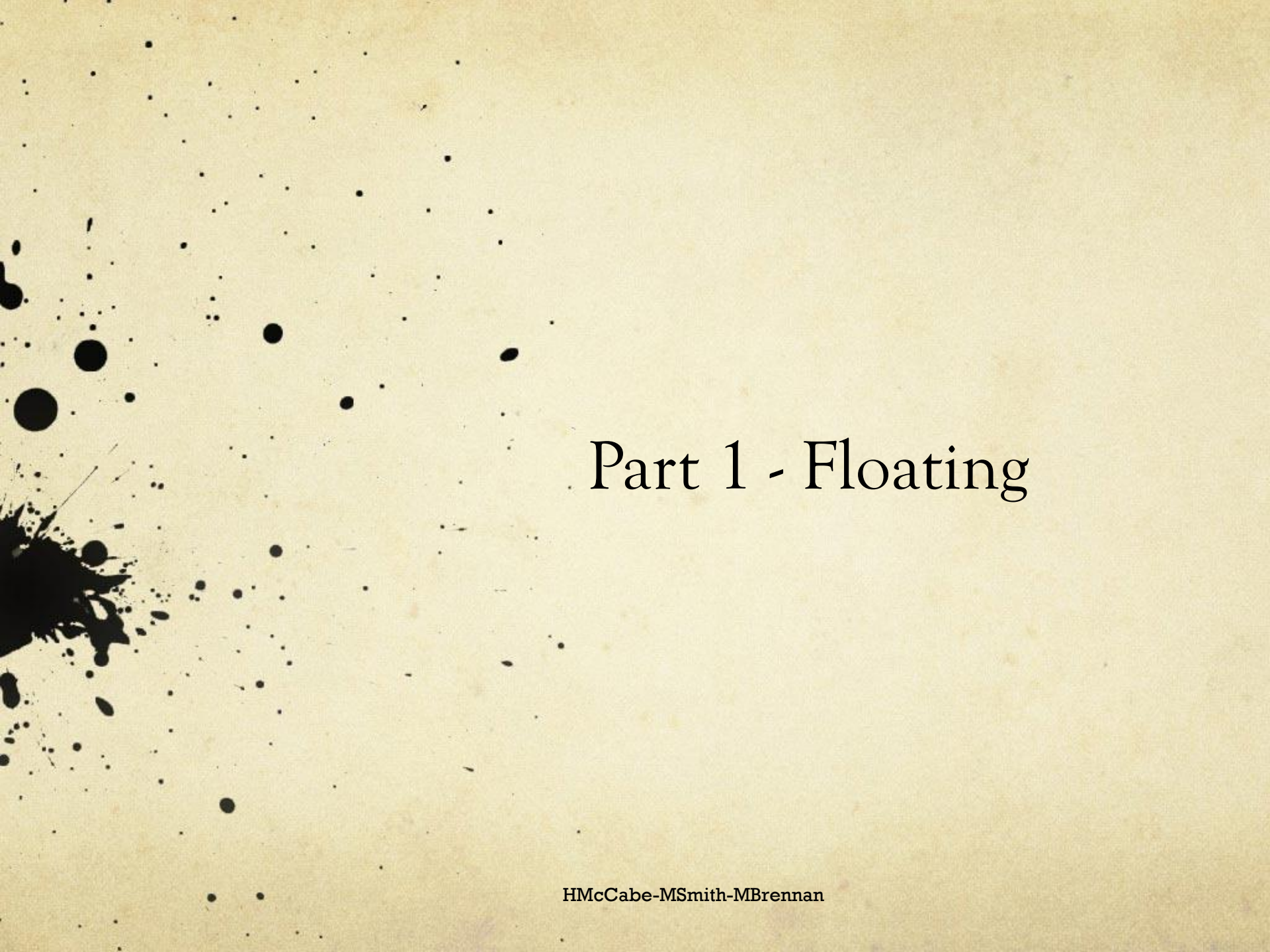
- In the CSS layout model, text elements are laid out from top to bottom in the order in which they appear in the HTML source.
- *BLOCK elements*: stack up TOP-to-BOTTOM vertically
  - They fill the available width of the browser window (or other containing elements if inside one)
- *INLINE elements* line up to each other LEFT-to-RIGHT
  - they line horizontally to fill block elements.
  - When the browser window is resized the block elements expand or contract and the inline elements reflow to fit

# Normal Flow



- Blocks are laid out in the order they appear in HTML source.
- Each block starts on a new line.
- Each block expands horizontally to fill as much space as it can.





# Part 1 - Floating



# Floating

- *Floated elements (or floats)* can be used to
  - Move images or text blocks to the LEFT or RIGHT and have main text ‘flow’ around the side box
  - create navigation toolbars from lists
  - create table-like alignment without tables
  - create multicolumn layouts
- We start by looking at the **float** property itself.

## **float**

*values:* left | right | none | inherit

*default:* none

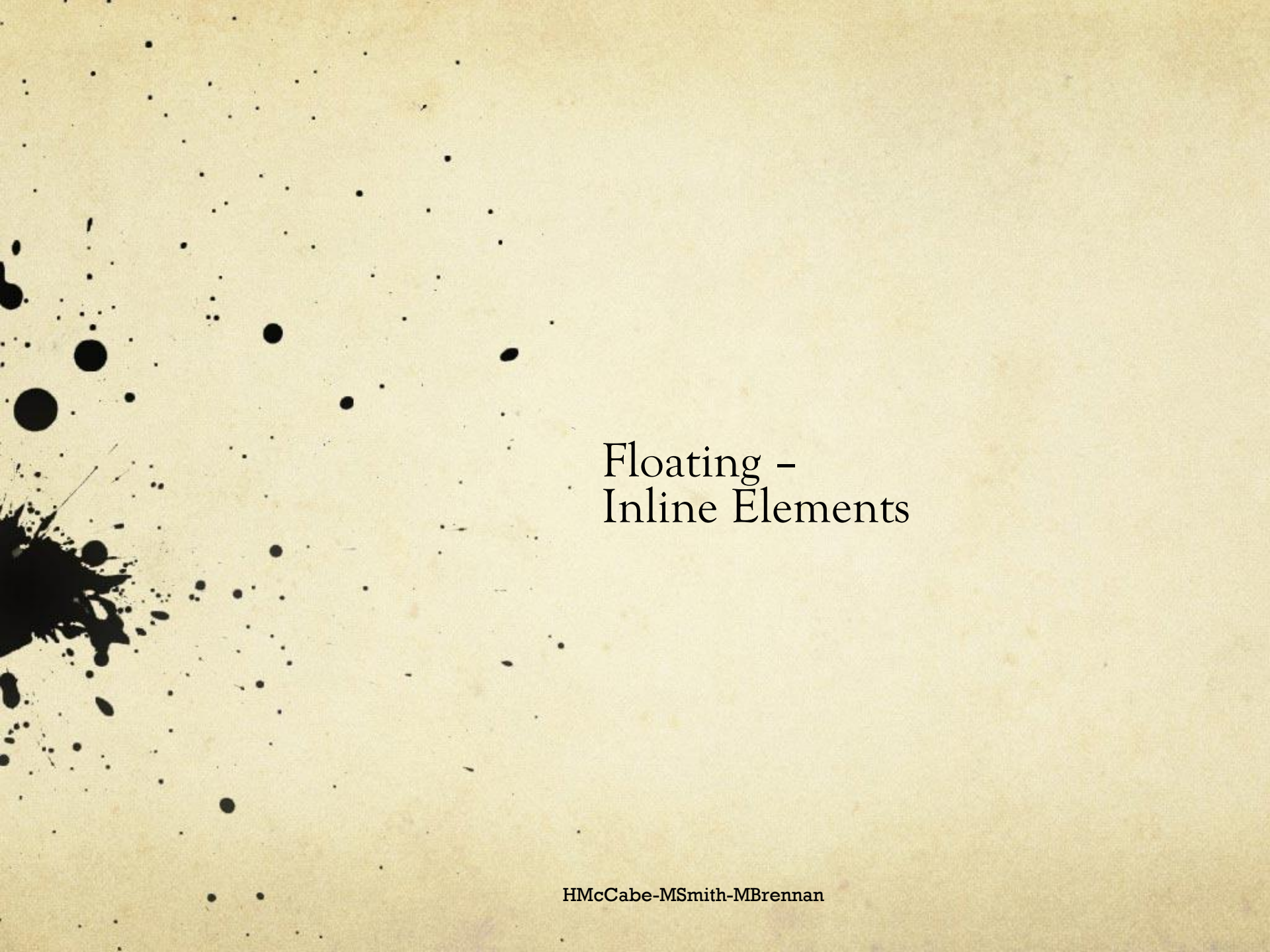
*applies to:* all elements

*inherits:* no

# Floating

- The **float** property moves an element as far as possible to the left or right
  - Elements are moved as far as possible left/right in their '**container**'...whatever that may be
    - which may be the <body> element, or it may be inside a <div> or <p> etc.
- The content following the FLOATed element is allowed to wrap around it
  - i.e. the HTML elements that appear AFTER the FLOATed element will be the ones that attempt to flow into the space 'vacated' by the FLOATed element





# Floating – Inline Elements



# Floating

- Let's look at a simple FLOAT example.
- We are going to apply the **float** property to an `<img>` element to float it to the right.
- First we will see how things look with no float applied
- Then we'll see the mark-up and results of 'floating' the image to the right...

# No float applied to IMG

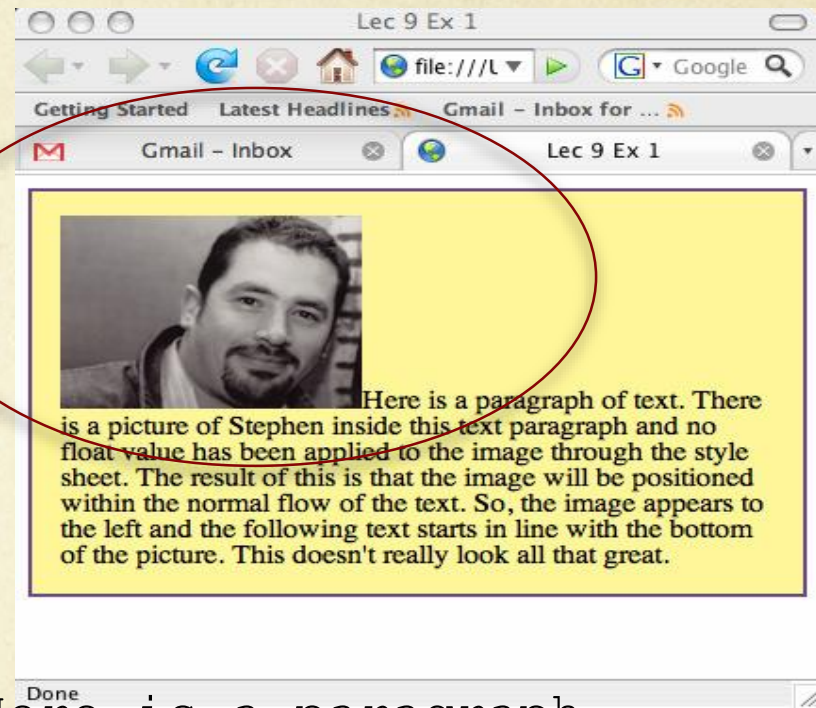
- The style sheet

```
p {  
    padding: 15px;  
    background-color: #FFF799;  
    border: 2px solid #6C4788;  
}
```

- The markup

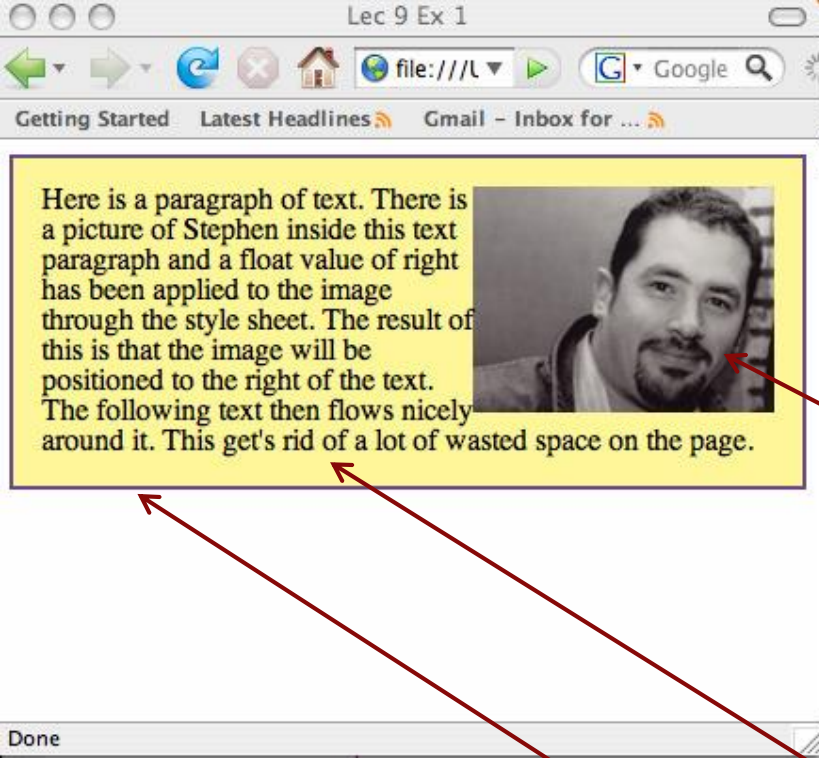
```
<p>  
 Here is a paragraph ...  
</p>
```

- The image appears in-line just like a large character ...





# The Floated image



- The style sheet:

**img**

{

**float: right;**

}

*p*

{

*padding: 15px;*

*background-color: #FFF799;*

*border: 2px solid #6C4788;*

}

- The image has been FLOATed to the right, other text 'flows' around it



# Margin around image

- We can create a bit of space around the image and stop the text bumping into it by adding a margin.

img

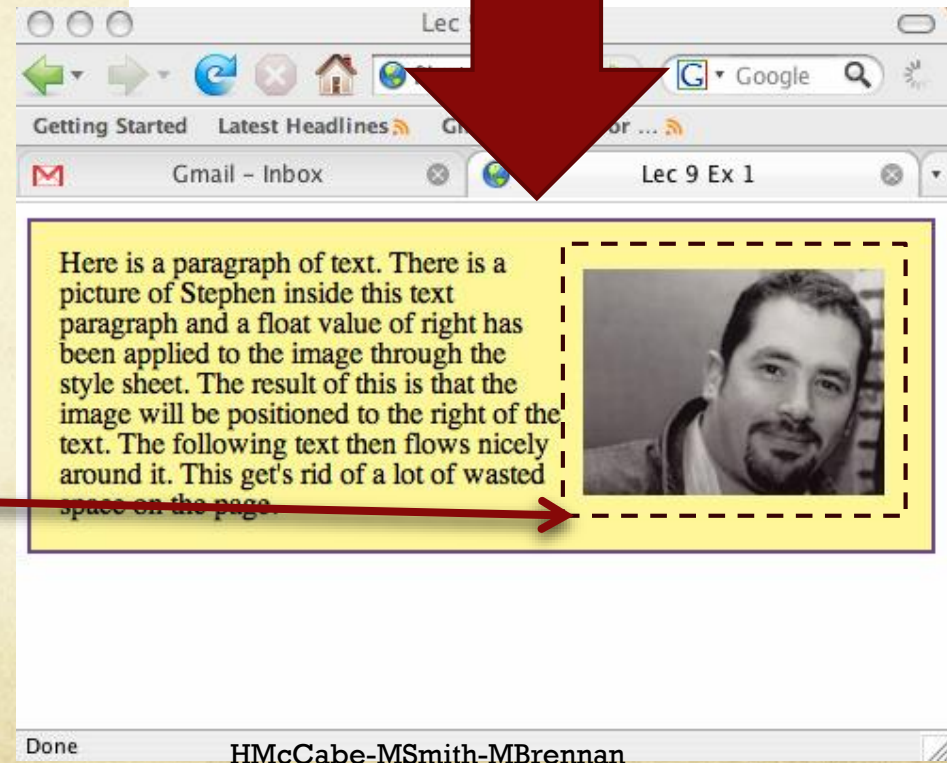
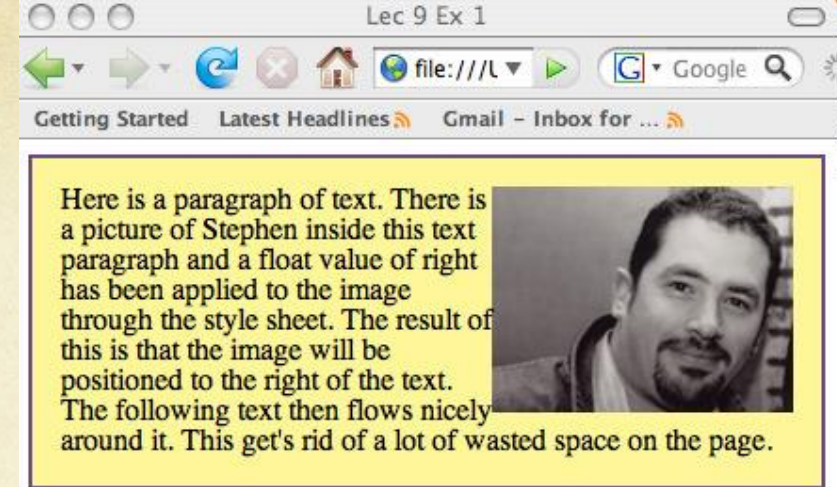
{

float: right;

margin: 10px;

}

- NOTE: Images have an inherent width and height



## Floats: 4 IMPORTANT NOTES

### 1. *A floated element is like an island in a stream.*

- The image was removed from its position in the normal flow but it still affects the position of the other elements.
- A common analogy is that a float is like an island in a stream - everything else has to flow around it.

### 2. *Floats stay in content area of the containing element.*

- In this case the paragraph.

### 3. *Margins are maintained.*

- The entire element is floated!

### 4. *Margins on floated elements **do not** collapse however. NB*

### 5. *Floated in-line elements behave like blocks with margins on all 4 sides ...*



# Floating Inline Elements

- Let's look at some more examples.
- We'll start by looking at what happens when we float an inline text element.

- The markup

**<p>**

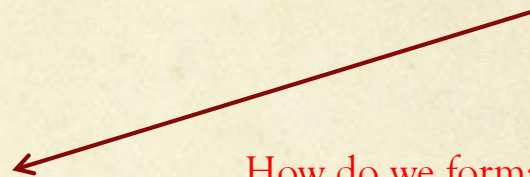
**<span class="disclaimer">**

Disclaimer: The existence of silver, gold and diamond trees is not confirmed.

**</span>**

They went down, down, till at last they came to a passage ...

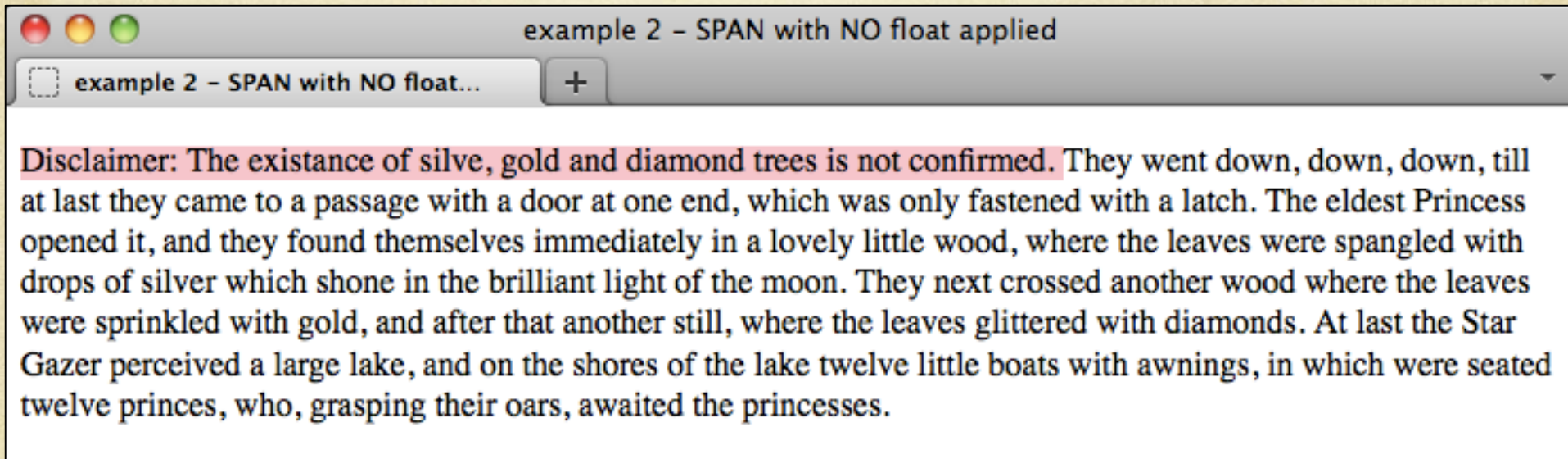
**</p>**



How do we format a class in css?



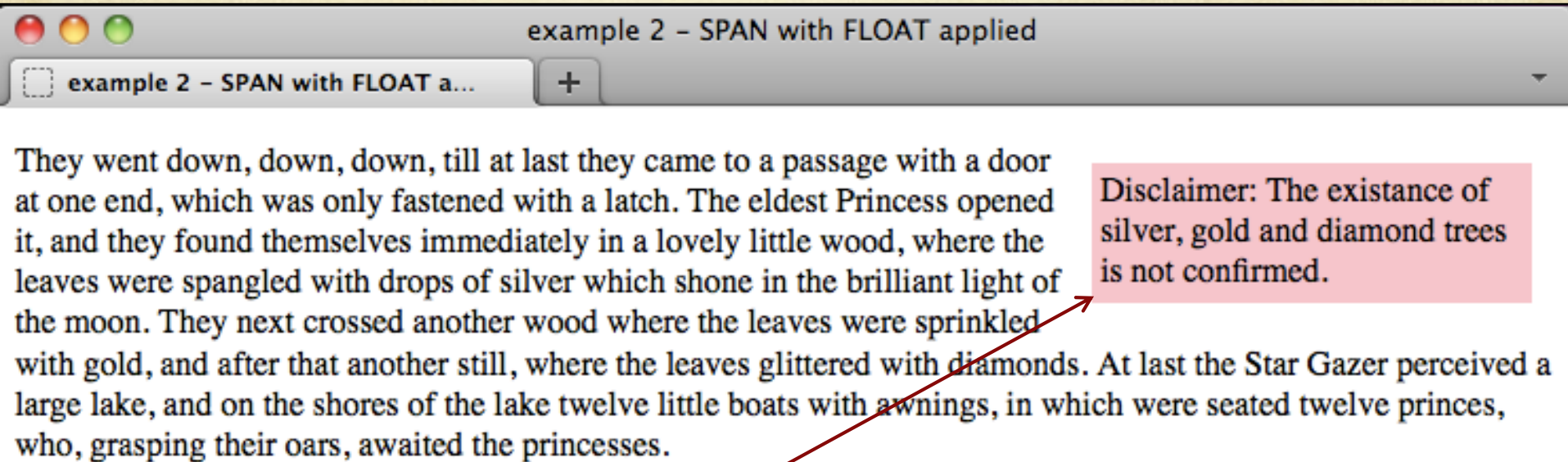
# No float applied to SPAN



- Some CSS markup to give a PINK background to the span

```
.disclaimer  
{  
    background-color: pink;  
}
```

# FLOAT applied to SPAN



```
.disclaimer {  
    float: right;  
    width: 200px;  
    background-color: pink;  
    margin: 10px;  
    padding: 4px;}
```



## More CSS to Para and SPAN for pretty effect

They went down, down, down, till at last they came to a passage with a door at one end, which was only fastened with a latch. The eldest Princess opened it, and they found themselves immediately in a lovely little wood, where the leaves were spangled with drops of silver which shone in the brilliant light of the moon. They next crossed another wood where the leaves were sprinkled with gold, and after that another still, where the leaves glittered with diamonds. At last the Star Gazer perceived a large lake, and on the shores of the lake twelve little boats with awnings, in which were seated twelve princes, who, grasping their oars, awaited the princesses.

**Disclaimer: The existence of silver, gold and diamond trees is not confirmed.**

```
.disclaimer {  
    float: right;  
    width: 200px;  
    margin: 10px;  
    padding: 4px;  
  
    color: white;  
    background-color: darkred;  
}
```

```
/* yellow background,  
grayish border with some  
padding space */  
  
p {  
    padding: 15px;  
    background-color: #FFF799;  
    border: 2px solid #6C4788;  
}
```

## Floating Inline Elements

- The FLOATed <span> behaves mostly like the floated image, but with a few subtle differences...
  1. **Always provide a width for floated text elements.**
    - You don't have to do this for images as they have an inherent width already.
  2. **Floated inline elements behave like block elements.**
    - For example, margins are rendered on all four sides.

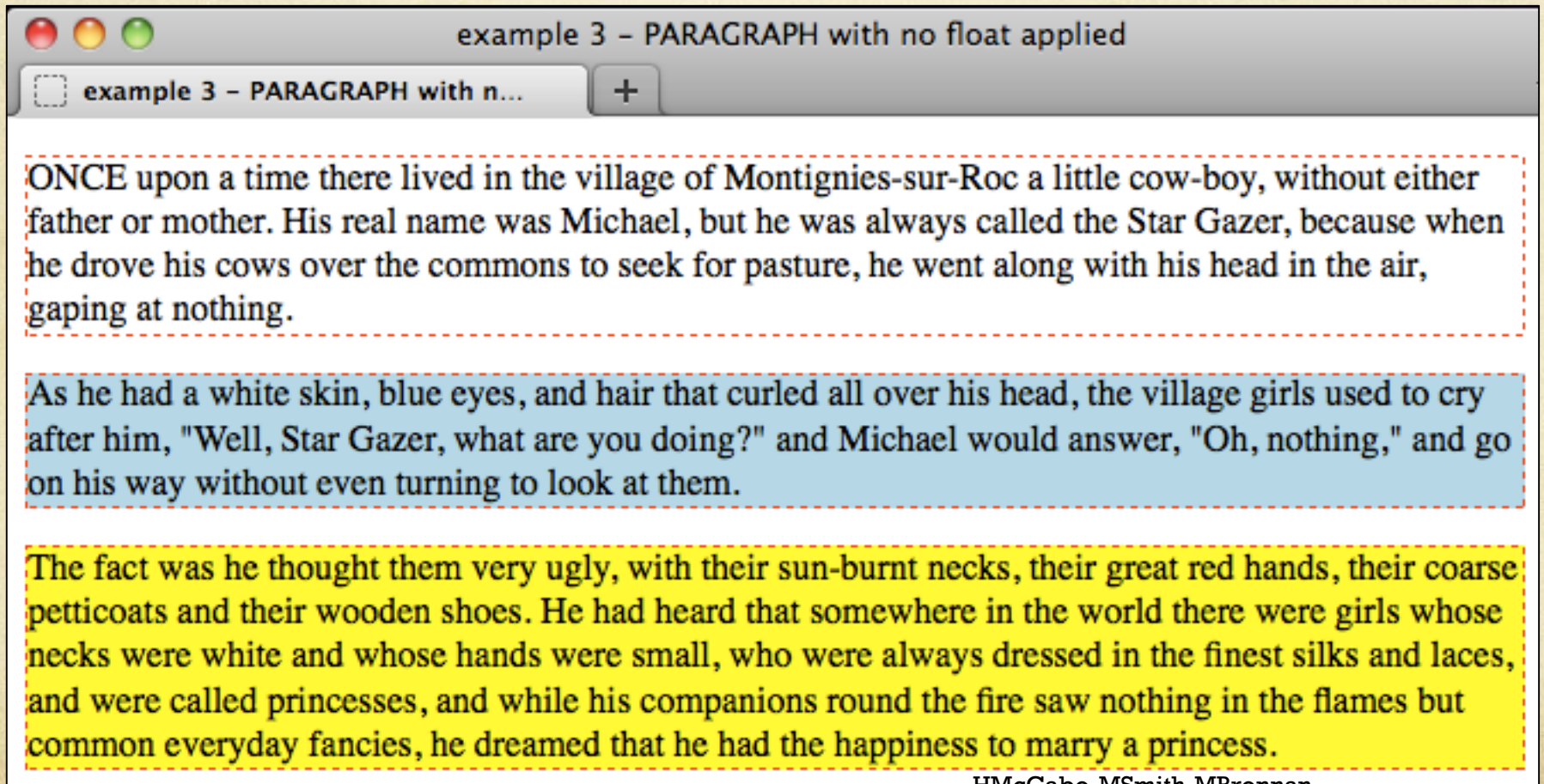




# Floating – Block Elements

## Floating Block Elements

- We will now try to float an entire paragraph to left.
- Here is how the 3 paragraphs look with NO FLOAT applied:





## Floating Block Elements

- We have 3 paragraphs, two have ID attributes:
- The HTML markup:

```
<p>ONCE upon a time ... </p>
```

```
<p id="para2">
```

```
  As he had a white skin, blue eyes ...
```

```
</p>
```

```
<p id="para3">
```

```
The fact was he thought them very ugly...
```

```
</p>
```

## The initial CSS markup

- All paragraphs have dashed red border.

```
p{  
    border: 1px red dashed;  
}
```

- Para2 has (light)blue background.  
Para3 has yellow background.

```
#para2{  
    background-color: lightblue;  
}
```

```
#para3{  
    background-color: yellow;  
}
```



## FLOAT para2 to the left (width of 30%)

– but PROBLEM with margins ...

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat predefined chunks as necessary, making this the first true generator on the Internet. It uses a dictionary of over 200 Latin words, combined with a handful of model sentence structures, to generate Lorem Ipsum which looks reasonable.

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat predefined chunks as necessary, making this the first true generator on the Internet. It uses a dictionary of over 200 Latin words, combined with a handful of model sentence structures, to generate Lorem Ipsum which looks reasonable.

```
#para2 {  
    float: left;  
    width: 30% }
```

ONCE upon a time there lived in the village of Montignies-sur-Roc a little cow-boy, without either father or mother. His real name was Michael, but he was always called the Star Gazer, because when he drove his cows over the commons to seek for pasture, he went along with his head in the air, gaping at nothing.

As he had a white skin, blue eyes, and hair that curled all over his head, the village girls used to cry after him, "Well, Star Gazer, what are you doing?" and Michael would answer, "Oh, nothing," and go on his way without even

The fact was he thought them very ugly, with their sun-burnt necks, their great red hands, their coarse petticoats and their wooden shoes. He had heard that somewhere in the world there were girls whose necks were white and whose hands were small, who were always dressed in the finest silks and laces, and were called princesses, and while his companions round the fire saw nothing in the flames but common everyday fancies, he dreamed that he had the happiness to marry a princess.

- FLOATED elements do NOT collapse margins
  - So the margin-bottom of "para1" (white background) touches, but does not overlap, the margin-top of FLOATED "para2"
- This is usually NOT the layout we desire



# SOLUTION 1 – remove margin-top for the FLOATed paragraph

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat predefined chunks as necessary, making this the first true generator on the Internet. It uses a dictionary of over 200 Latin words, combined with a handful of model sentence structures, to generate Lorem Ipsum which looks reasonable.

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat predefined chunks as necessary, making this the first true generator on the Internet. It uses a dictionary of over 200 Latin words, combined with a handful of model sentence structures, to generate Lorem Ipsum which looks reasonable.

```
#paraz {  
    float: left;  
    width: 30%;  
    margin-top: 0px;  
}
```

That's better!

## SOLUTION 2 - FLOAT both paragraphs to the LEFT (so have same non-collapsed margins)

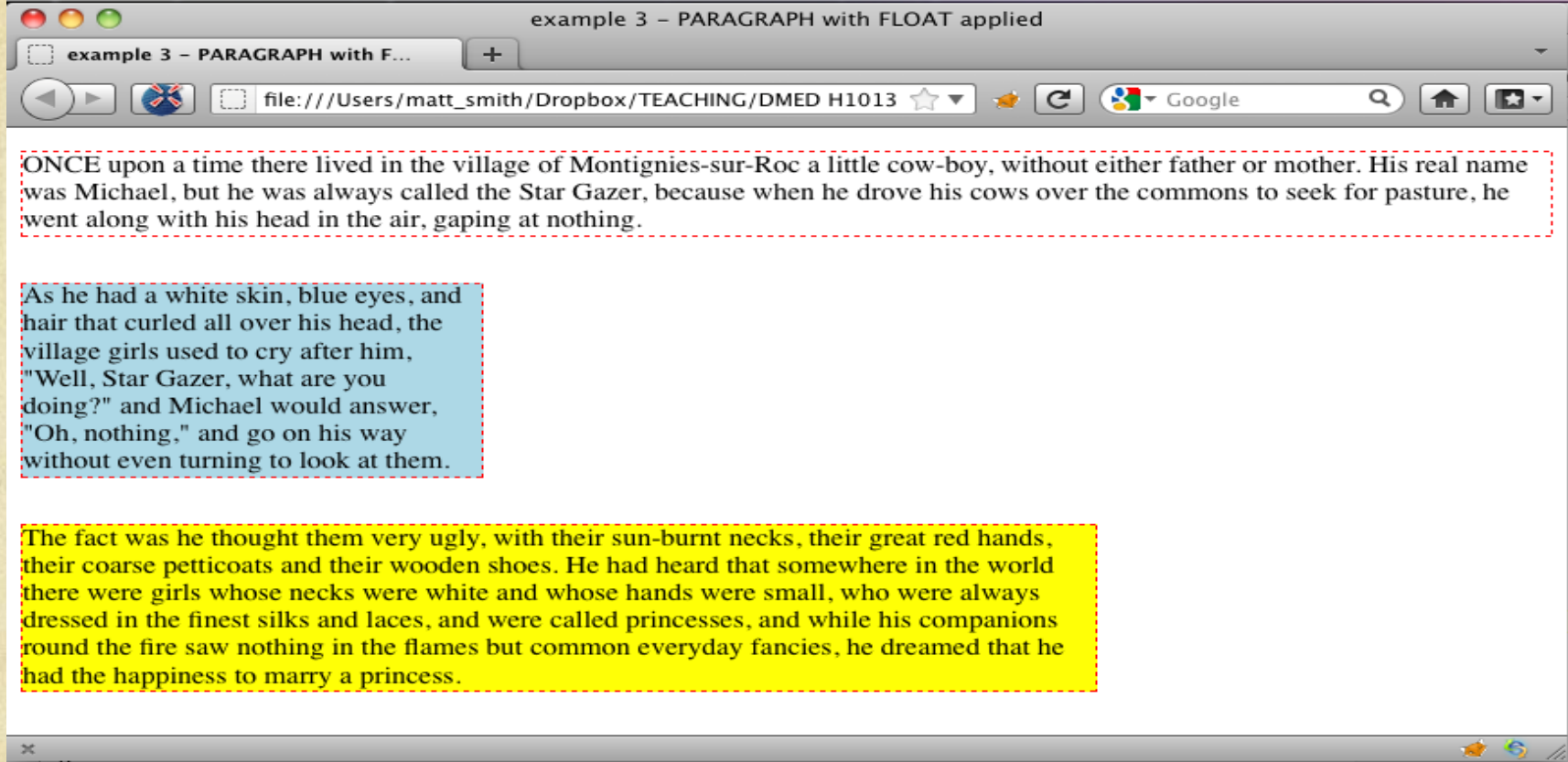
```
#para2
```

```
{  
    float: left;  
    width: 30%;  
}
```

```
#para3
```

```
{  
    float: left;  
    width: 70%;  
}
```





- But for some reason,  $30\% + 70\%$  doesn't seem to fit
  - Why doesn't  $30 + 70 = 100$ ???
- The problem relates to margins/border/padding of paragraphs
  - The "box model" of what makes up TOTAL width of an element ...

### ***SOLUTION 3*** – float 2 DIVs, created around each paragraph (&no spacing/border for DIVs)

- The new HTML markup for the 2 paragraphs to be FLOATed:

```
<div id="column_left">
```

```
  <p id="para2">
```

```
    As he had a white skin, blue eyes, ...
```

```
  </p>
```

```
</div>
```

```
<div id="column_right">
```

```
  <p id="para3">
```

```
    The fact was he thought them very ugly, ...
```

```
  </p>
```

```
</div>
```



## SOLUTION 3 – float 2 DIVs, created around each paragraph (&no spacing/border for DIVs)

- The new CSS markup for the 2 paragraphs to be FLOATed:

```
#column_left
```

```
{
```

```
    float: left;
```

```
    width: 30%;
```

```
}
```

```
#column_right
```

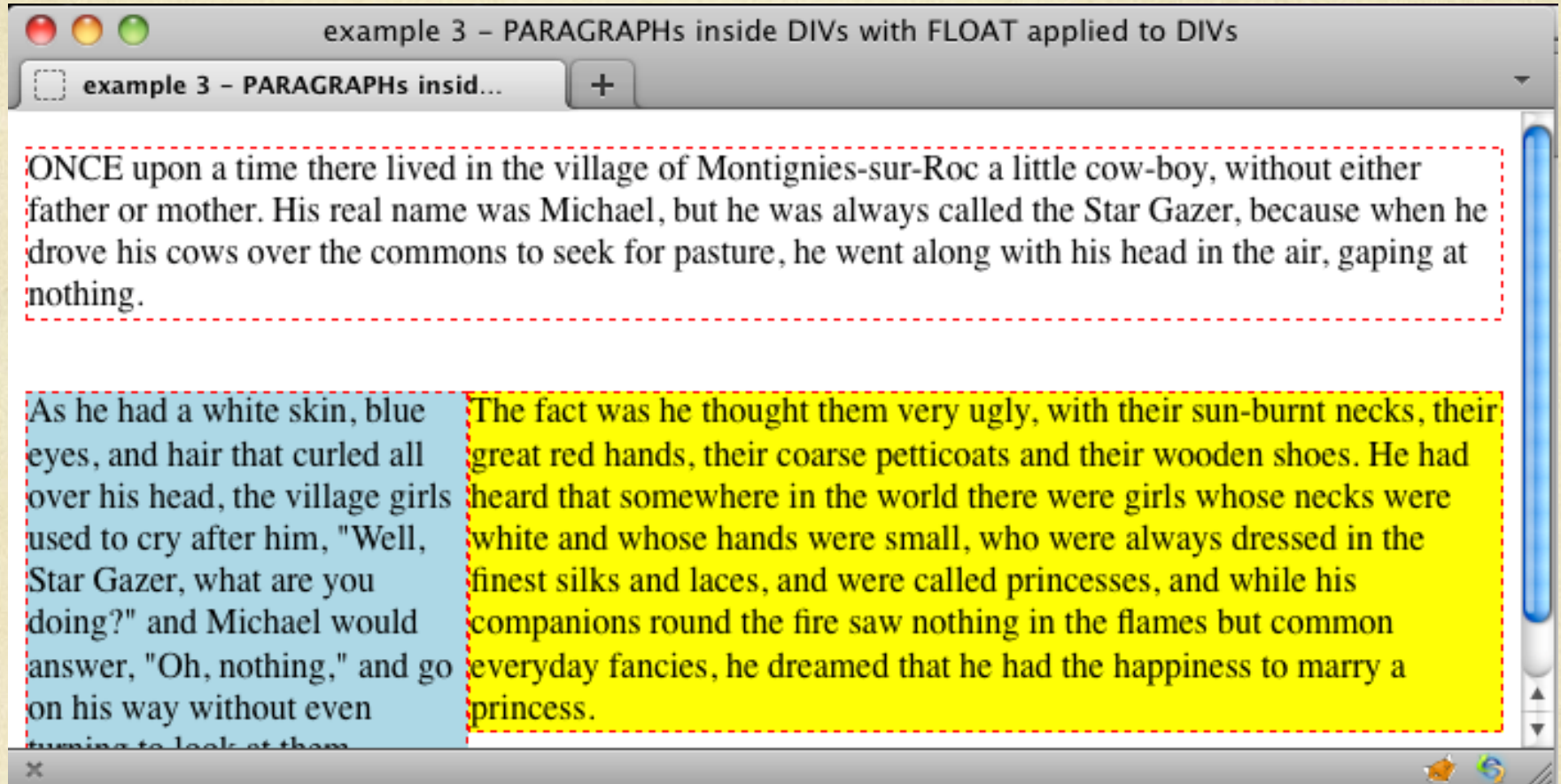
```
{
```

```
    float: left; /* or right! The result is the  
same */
```

```
    width: 70%;
```


```
}
```

## SOLUTION 3 – float 2 DIVs, created around each paragraph (& no spacing/border for DIVs)



- It works! Why? Because.....
  - The DIVs have no associated margin/border/padding
  - And so 30% + 70% really does add up to 100%





Forcing content to start BELOW a  
FLOAT  
- “CLEARing” floats ...

## Clearing Floated Elements

- Suppose you want to turn text wrapping off and get back to layout as normal.
- Apply the clear property to the element you want to start *below* the floated element.

### **clear**

*values:* left | right | both | none | inherit

*default:* none

*applies to:* block level elements only

*inherits:* no



Consider page header and footer (gray) padded paragraphs added to our story page ...**PROBLEM** – the (gray) page footer block gets mixed up with the 2 floated paragraphs ...

## Title here

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat predefined chunks as necessary, making this the first true generator on the Internet. It uses a dictionary of over 200 Latin words, combined with a handful of model sentence structures, to generate Lorem Ipsum which looks reasonable.

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat predefined chunks as necessary, making this the first true generator on the Internet. It uses a dictionary of over 200 Latin words, combined with a handful of model sentence structures, to generate Lorem Ipsum which looks reasonable.

**Created by WebFoot Designs**

SOLUTION – the page footer must CLEAR floats before it is displayed (so on new line ...)

```
#header, #footer
{
    background-color: #CCCCCC;
    padding: 15px;
}
```

The layout CSS markup for the page footer:

```
#page_footer
{
    clear: both;
    /* that's it - floats are stopped
       and footer starts on new line */
}
```




# The CLEAR solution for the page footer ...

## Title here

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat predefined chunks as necessary, making this the first true generator on the Internet. It uses a dictionary of over 200 Latin words, combined with a handful of model sentence structures, to generate Lorem Ipsum which looks reasonable.

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat predefined chunks as necessary, making this the first true generator on the Internet. It uses a dictionary of over 200 Latin words, combined with a handful of model sentence structures, to generate Lorem Ipsum which looks reasonable.

## Created by WebFoot Designs

The background of the slide is a light beige or cream color, resembling aged paper. It is decorated with numerous black ink splatters and dots of varying sizes. A large, dense cluster of splatters is located on the left side, while smaller, more isolated dots are scattered across the rest of the page.

Setting the Content  
Dimensions within the box  
model



## Setting the element dimensions

- Use the **width** and **height** properties to specify the width and height of the element.
- You can specify the width and height only of block-level elements or non-text inline elements such as images.

### **width or height**

*values:* length measurement | percentage | auto | inherit

*default:* auto

*applies to:* block level elements and non-text inline elements

*inherits:* no

- By default the width and height of an element is calculated automatically by the browser.
- Hence the default setting of **auto**.

## Setting the element dimensions

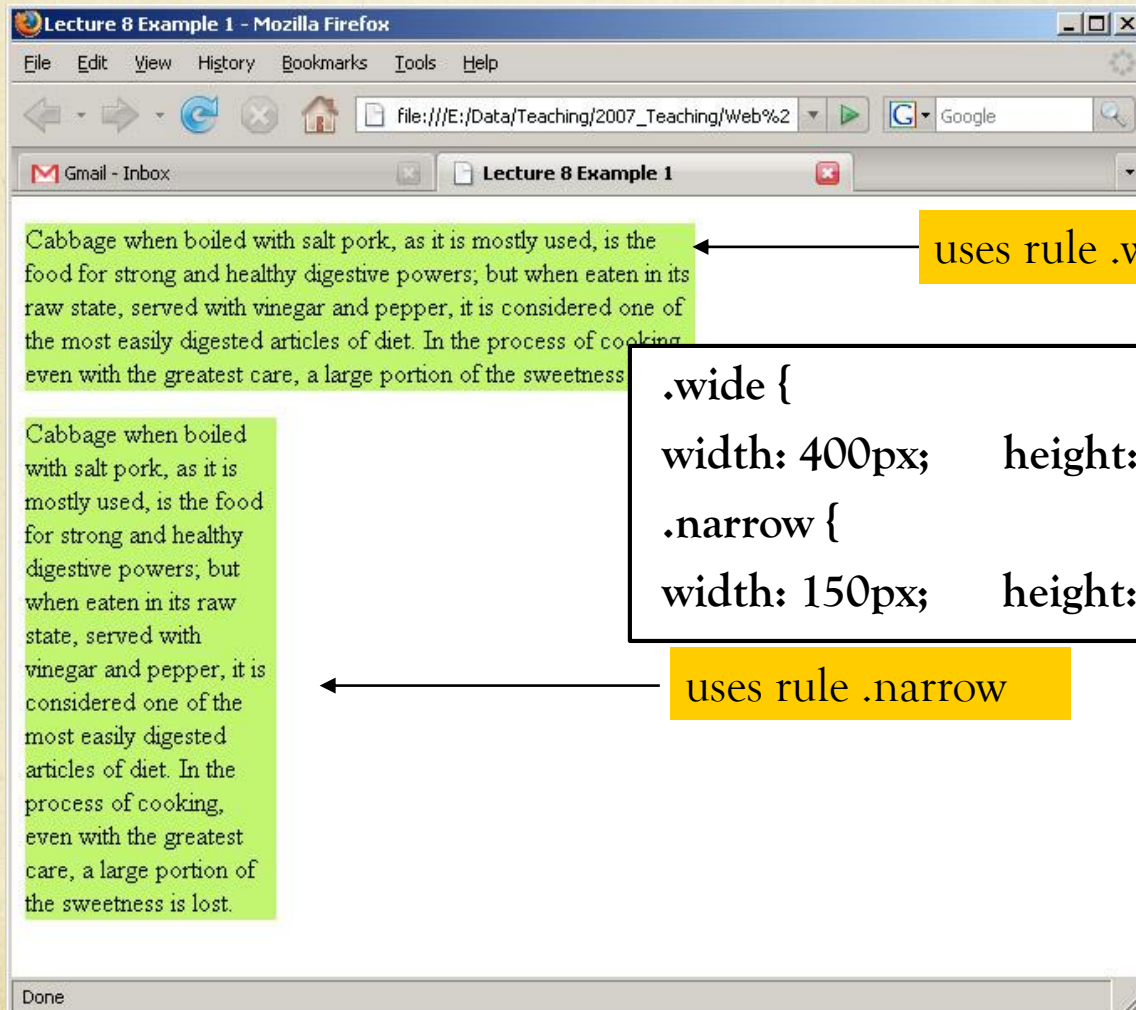
- However you can use the **width** and **height** properties to set this yourself – usual values to give are em, pixel or percentage values.
- Here are some examples of using them:

```
.wide { width: 400px;    height: 100px; }  
.narrow { width: 150px;    height: 300px;  
}
```

- Slide overleaf shows the results of formatting the same paragraph twice with the two different style rules above. A green background has been added for clarity.



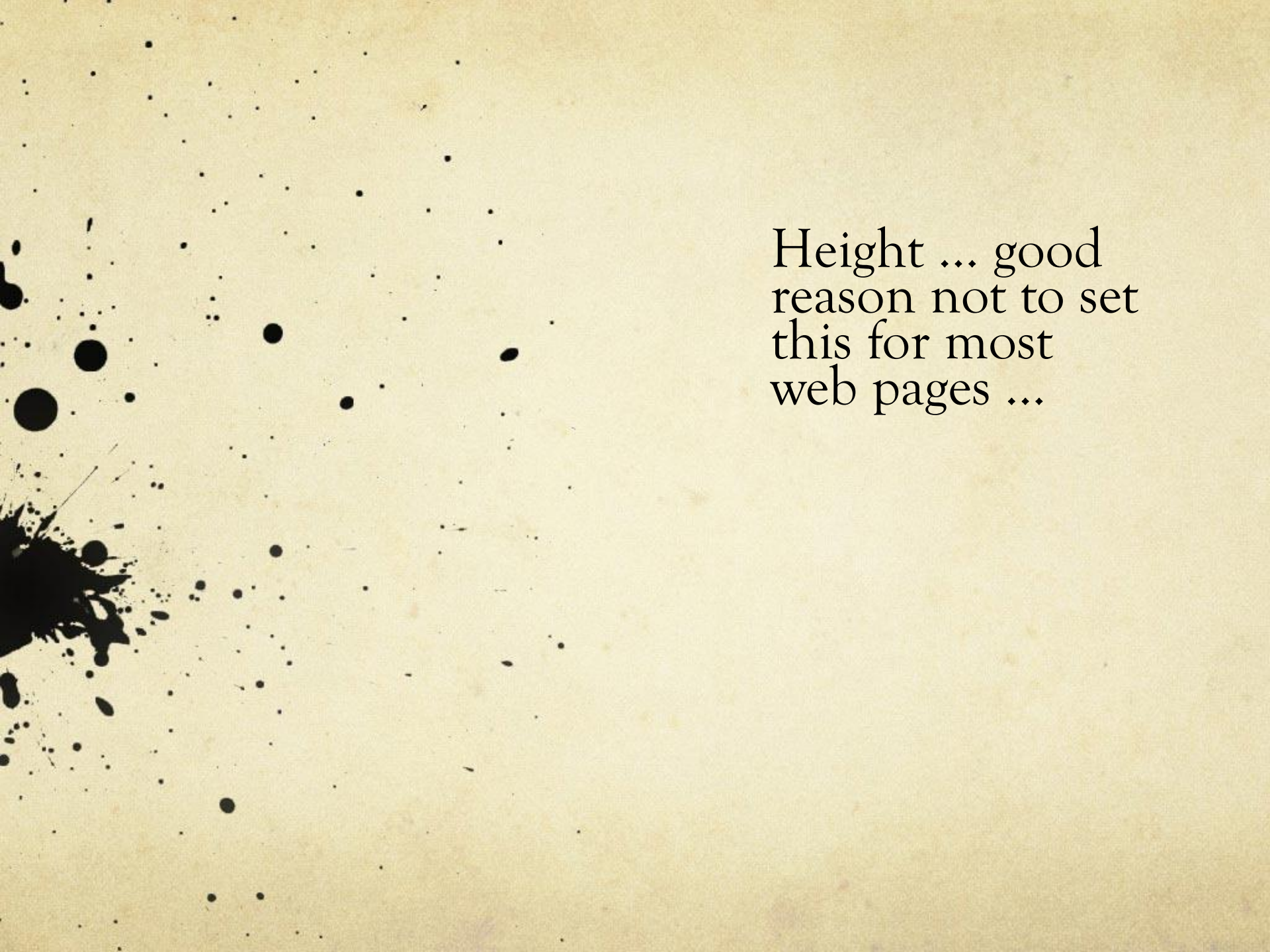
# Setting the Content Dimensions



uses rule .wide

```
.wide {  
width: 400px;    height: 100px; }  
.narrow {  
width: 150px;    height: 300px; }
```

uses rule .narrow



Height ... good  
reason not to set  
this for most  
web pages ...



## Setting the Content Dimensions

- In general you don't specify the height of elements.
- It's usually better to let the browser figure this out based on how much content there is.
- If you do specify height though, you must consider what happens if the content does not fit. This situation is called *overflow*.
- There is a property called **overflow** which can be used to specify this. There are five predefined values it can take
  - visible, hidden, scroll, auto, inherit
- Next slide shows examples ...

## Setting the Content Dimensions

visible

Cabbage when  
boiled with salt  
pork, as it is  
mostly used, is  
the food for  
strong and  
healthy digestive  
powers; but  
when eaten in its  
raw state,  
served with  
vinegar and

hidden

Cabbage when  
boiled with salt  
pork, as it is  
mostly used, is  
the food for

scroll

Cabbage  
when boiled  
with salt  
pork, as it is

The auto option just lets the browser decide what to do ... i.e. it will add Scroll-bars when overflow begins ...



# Summary and Conclusions

- We have now dealt with how to *float* elements so they break out of normal flow and text can wrap around them.
- FLOATs can be effective to create complex and visually interesting layouts
  - However, the issue of non-collapsing margins means that either templates must be used (and carefully adjusted) or that page layouts created from scratch must pay careful attention to the addition of top border/padding to ensure that non-floated columns line up with FLOATed ones

## The 5 rules for effective use of Floats

1. A width must be defined for any Floated item
1. Margins do not collapse for Floated items
2. Floated in-line elements behave as block-level elements
3. Floated items should appear first in the HTML file
4. Non-floated elements need right/left margins to prevent flowing underneath Floated elements



## 2- and 3- column FLOAT templates

1. All content inside a 960px 'wrapper' DIV
2. Sum of all 2 or 3 columns = 960px
3. Internals of each column can be individually styled, without 'breaking' the container DIVs
4. We will avoid all of this hassle by using flex boxes for layout.

# Next

- Navigation bar
- Flex boxes
- No lecture next week
- **Lab06 to attend lab session on Wednesday evening from 4pm-6pm in F202**
- Next weeks plan: Lecture/lab session to create a navigation bar that is flexed