HTML & CSS: FUNDAMENTALS OF DEVELOPMENT

Instructor: Beck Johnson

Week 2

SESSION OVERVIEW

- Week One review and questions
- File organization
- Version control and code sharing with Git
- Background images and gradients with CSS
- How to FTP (put your website online)

REVIEW!

REVIEW: WEBPAGE COMPONENTS

HTML

Structures and organizes content

CSS

Styles the markup and creates layout

JAVASCRIPT

Brings content and design to life

REVIEW: HTML DOCUMENTS

- `<!DOCTYPE html>` tells the browser it's serving an HTML file using HTML5 standards
- `<html>` wraps the whole document
- `<head>` wraps the metadata and styles
- `<body>` wraps the visible content
- Most HTML elements have **opening** and **closing tags** and some have **attributes**

REVIEW: LAYOUT ELEMENTS

- `<header>` wraps header content
- `<footer>` wraps footer content
- `<nav>` indicates that everything inside is related to navigation
- `<section>` is used to define content sections

REVIEW: HTML CONTENT

- **Headings** create an header/outline

  `<h1>...<h6>`

- **Paragraphs** and **lists** structure text

  ```
  <p>
  <ul>
  <ol>
  ```

- **Images** and **links** both require **attributes** to work

REVIEW: IMAGES

```
<img src="kitten.jpg" alt="Cute kitten" />
```

- Does not have a closing tag ("self-closing")
- Two required `attributes`:
  - **src** is where the file lives (local or external)
  - **alt** is a description of the image (used for screen readers, search engines, etc.)

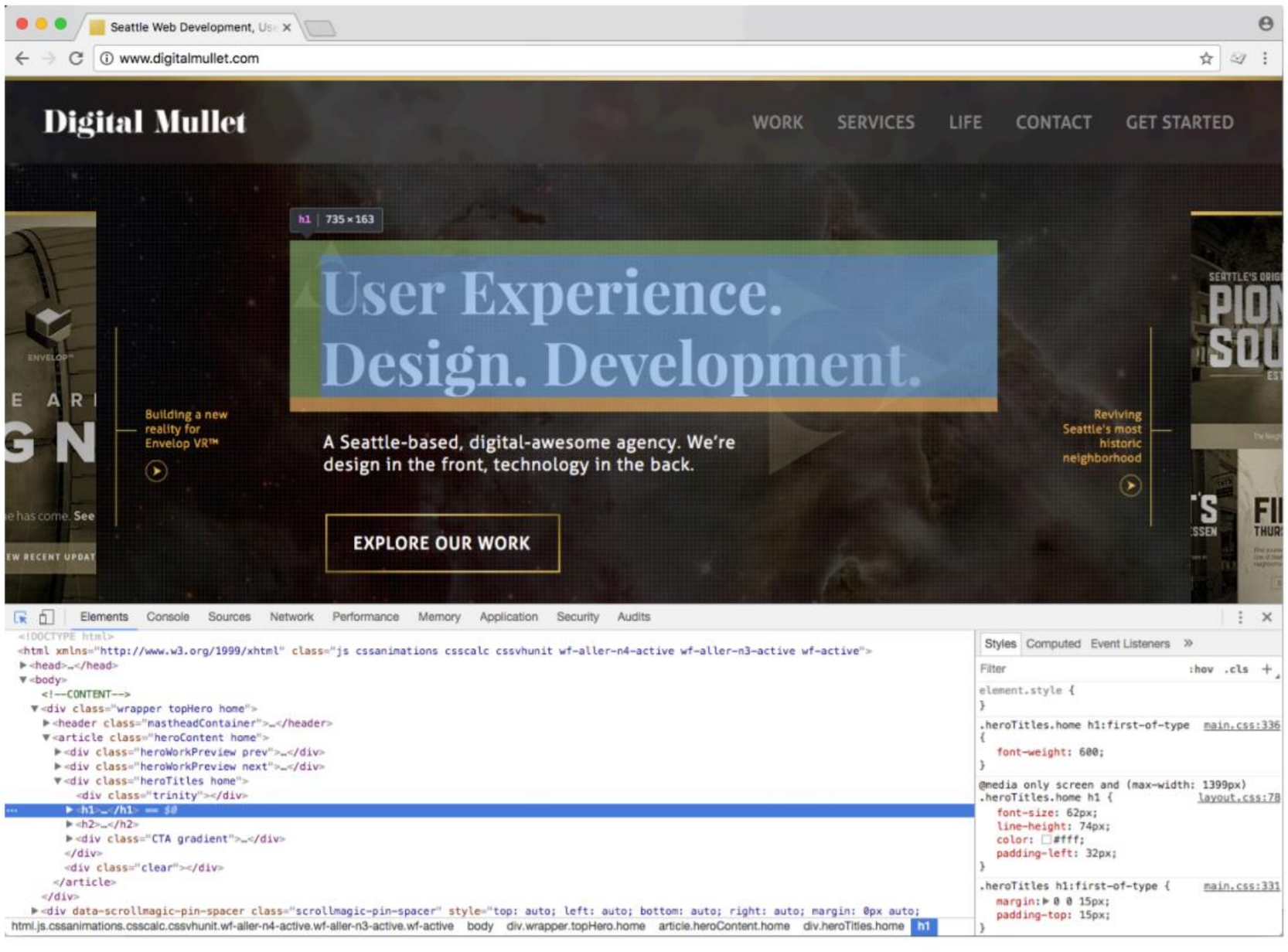
REVIEW: LINKS

```
<a href="http://google.com">Google</a>
```

- Creates a link to other pages or websites
- The `href` attribute says where the link should go
- Anything inside `<a>` tags is clickable

REVIEW: DEV TOOLS

Right-click > Inspect, or hit the F12 key



REVIEW: CSS

CSS (Cascading Style Sheet) is a different type of language than HTML, and has its own syntax

- CSS can go directly in your HTML file, inside a `<style></style>` element

- CSS can be added in a separate .css file that can be linked to your HTML page

REVIEW: EXAMPLE CSS RULE

```
p { color: blue; }
```

- `selector` is `p` (all `<p>` tags in the HTML)
- `property` is `color`
- `value` is `blue` (many color names are supported, or use the hex code `#0000ff`)

QUESTIONS?

FILE ORGANIZATION

FILE ORGANIZATION

- If you structure your site correctly, you are one step closer to faster updates
    - Structure should be not just for you, but for anyone who might use, need or want any of your files (images, scripts, stylesheets, etc)
- The next person to work on or look at your code will be able to understand what you've done and where to find things

FILE ORGANIZATION

Typical files in a website include:

HTML files (.html)
CSS files (.css)
Javascript files (.js)
Images (.png, .jpg, .gif)

- HTML should usually go in the **main** (root) directory

- Make **subdirectories** for media, CSS, and Javascript files

FILE NAMING RULES

- Use a consistent naming convention when naming files and folders
    - For example, always all lowercase, or words always separated by dashes, etc
- Capitalization matters
    - INDEX.html is not the same as index.html
- Use only letters, numbers, hyphens (-) or underscores (_)
- No spaces in file names

CODE ORGANIZATION

- Comment your files – especially if you have unfinished development code, or if you think you may forget *why* you made the decision you did

```
.viewmore {
    max-height: 2.85714286em; /* line-height of the paragraph x 2 */
}
```

- Indent your code (trust me)

CSS INDENTATION

This is one popular way to indent CSS

- Starting bracket is on the same line as the selector
- Each property is on its own line, tabbed once
- Ending bracket is on its own line

```
8  html, body {
9      background-color: #fff;
10     color: #4a3c31;
11     font-size: 16px;
12     height: 100%;
13 }
14
15 body, p {
16     font-family: "Lato", Arial, Helvetica, Lucida Grande, Sa
17     line-height: 1.5;
18 }
19
20 a {
21     color: #4a3c31;
22     text-decoration: none;
23 }
24
25 img {
26     max-width: 100%;
27 }
28
29 strong {
30     font-family: "Lato-Bold", Arial, Helvetica, Lucida Grand
31 }
```

PRACTICE TIME!

ASSIGNMENT

Create a folder for your images and move all images there

- Fix the paths in all your `<img src>` tags so that images show like before
- Remember the different between **relative** and **absolute** paths

Prettify your CSS and HTML so that it's easy to read

- Use indentation and whitespace


CSS

{ } STATES IN CSS

CSS also allows you to apply styles based on the **state** of an element

- Being **hovered** over with a mouse
- Gaining **focus** via tabbing or clicking

This is known as a CSS pseudo-class (because it doesn't really exist)

Whenever you see a `:` in a selector, that style will only apply to elements that are in that state

{ } PSEUDO EX-PSAMPLE

```
p:hover { background-color: #999; }
```

This paragraph gets fancy when you hover over it

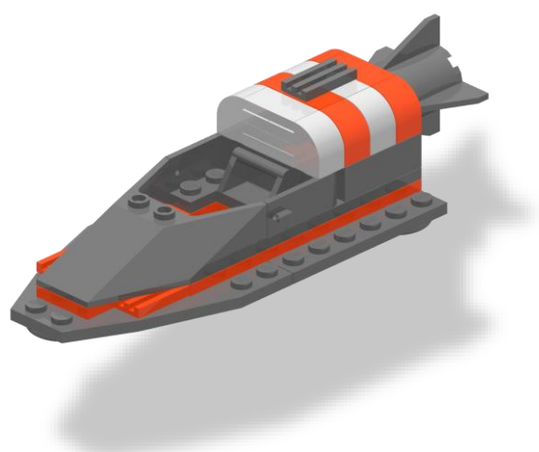
{ } LINK STATES

Links have two additional states

- `:visited` only applies once the user has viewed the page that is being linked to
- `:link` only applies to unvisited pages

`a:link`

`a:visited`



`a:hover`

`a:active`

{ } FOUR LINK STATES

```
a { color: blue; }

a:visited { color: gray; }

a:hover {color: purple; }

a:active { color: yellow; }
```

Let’s inspect a live demo of how this looks

{} CSS IN MULTIPLE PLACES

So far, we've been making CSS changes directly on a single webpage, in the `<head>` element.

- These **internal styles** only apply to that page (but affect every element on that page that is styled)

{} CSS IN MULTIPLE PLACES

You can also add **inline styles** to a single element by using the `style` attribute in HTML markup

```
<p style="color: red">This paragraph is
special.</p>
```

- Inside the `style` attribute, use the same syntax as CSS (selector: value)
- Typically discouraged, because it can be hard to maintain

{} CSS IN MULTIPLE PLACES

The most common way to use CSS in the “real world” is to use an **external stylesheet**.

- CSS lives in a separate .css file
- The **same** stylesheet can be included on multiple pages
- A single page can include **multiple** stylesheets

{} LINKING TO EXTERNAL STYLESHEET

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

- Tells the browser to find and load the styles.css file from the css directory
- The `rel` attribute stands for "relation" - in this case, this link's relationship to the document is "stylesheet"
- This tag goes inside the `<head>` element
- Should be on every page that needs the styles

{} THE "CASCADING" PART

The beauty of CSS is being able to create styles and then **override** them when you want to customize the look of your pages.

There are **3 rules** for determining how styles get applied:

- Styles are applied from **far** to **near**
- Styles are applied from **top** to **bottom**
- **Children** elements are more specific than **parents**

{} FAR TO NEAR

Styles that are “closer” to the elements they style take precedence.

Browser default

External style (from a .css file)

Internal style (from styles in the `<head>`)

Inline styles (directly on the element)

{} TOP TO BOTTOM

CSS rules are applied sequentially

If the same property is styled multiple times for the same selector, **the last one wins**

```
p { color: #2f4251; }

p { color: #daa645; } /* this one wins */
```

{} HTML CHILDREN

In an HTML document, an element that is nested inside another element is referred to as a “child” of that element

- `<h2>` is a child of `<header>`
- `<a>` are children of `<nav>`

```
<html lang="en">
▶ <head>…</head>
▼ <body>
  ▼ <section>
    ▼ <header>
        <h2>Kangaroo Valley Safari</h2>
      </header>
    ▼ <p>
        "Located two hours south of Sydney in
        the Southern Highland of New South
        Wales..."
      </p>
    ▼ <nav>
        <a href="#">SHARE</a>
        <a href="#">LEARN MORE</a>
      </nav>
    </section>
  </body>
</html>
```

Both `<a>` and `<h2>` are also children of `<section>`

{} HTML CHILDREN

In CSS, to style only elements that are inside another element, use this syntax:

```
parent child { property: value; }
```

```
nav a { color: #c4fe46; }
```

“Change the color of links that are contained within a nav”

```
<html lang="en">
▶ <head>…</head>
▼ <body>
  ▼ <section>
    ▼ <header>
        <h2>Kangaroo Valley Safari</h2>
      </header>
    ▼ <p>
        "Located two hours south of Sydney in
        the Southern Highland of New South
        Wales..."
      </p>
    ▼ <nav>
        <a href="#">SHARE</a>
        <a href="#">LEARN MORE</a>
      </nav>
    </section>
  </body>
</html>
```

{} CHILDREN ARE SPECIFIC

Children elements **inherit** styles from their parents, but can **override** with their own style

```
p { color: #daa645; } /* all paragraphs */

b { color: #e7c0c8; } /* bold text in general */

p b { color: #c4fe46; } /* bold text in paragraphs */
```

PRACTICE TIME!

{ } EXTERNAL STYLESHEETS

Create a folder for css, then create a new file in that folder called **styles.css**

- Copy and paste the styles from inside `<style></style>` into that .css file, then delete the `<style></style>` tag

Create a link to your new stylesheet on all of your webpages:

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

- Does everything still look the same?

Make a style change that only affects a child element, for example:

- Only links inside a footer, header, or navigation element
- A list that's inside another list

VERSION CONTROL

VERSION CONTROL

In modern development, most websites are a team effort.

Version control systems (VCS) allow multiple people to work on the same file with less risk of overriding changes.

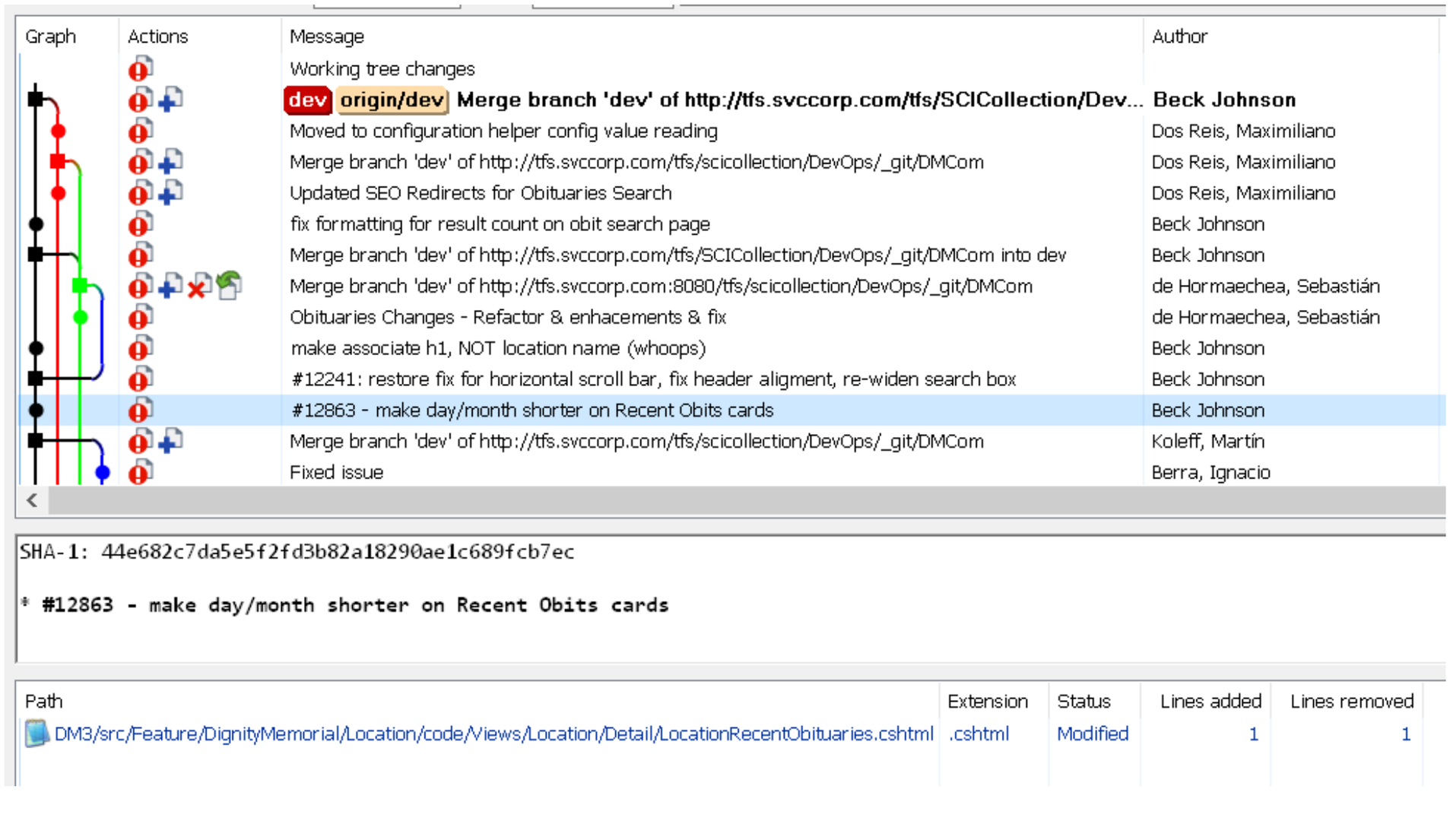
GIT

is a free, open-source version-control system

Git is software that tracks any changes that you (or anyone else) make to a project

- A project contains every file and subdirectory inside a folder
- The “master copy” of the project is stored on a remote server
- You can download a copy of the project from the server, create new versions of the project, and also have access to every version (of every file!) saved along the way

When you upload changes to the remove server, you can add a friendly message indicating what you changed:



Graph	Actions	Message	Author
		Working tree changes	
		dev origin/dev **Merge branch 'dev' of http://tfs.svccorp.com/tfs/SCICollection/Dev...**	**Beck Johnson**
		Moved to configuration helper config value reading	Dos Reis, Maximiliano
		Merge branch 'dev' of http://tfs.svccorp.com/tfs/scicollection/DevOps/_git/DMCom	Dos Reis, Maximiliano
		Updated SEO Redirects for Obituaries Search	Dos Reis, Maximiliano
		fix formatting for result count on obit search page	Beck Johnson
		Merge branch 'dev' of http://tfs.svccorp.com/tfs/SCICollection/DevOps/_git/DMCom into dev	Beck Johnson
		Merge branch 'dev' of http://tfs.svccorp.com:8080/tfs/scicollection/DevOps/_git/DMCom	de Hormaechea, Sebastián
		Obituaries Changes - Refactor & enhacements & fix	de Hormaechea, Sebastián
		make associate h1, NOT location name (whoops)	Beck Johnson
		#12241: restore fix for horizontal scroll bar, fix header aligment, re-widen search box	Beck Johnson
		#12863 - make day/month shorter on Recent Obits cards	Beck Johnson
		Merge branch 'dev' of http://tfs.svccorp.com/tfs/scicollection/DevOps/_git/DMCom	Koleff, Martín
		Fixed issue	Berra, Ignacio

```
SHA-1: 44e682c7da5e5f2fd3b82a18290ae1c689fcb7ec

* #12863 - make day/month shorter on Recent Obits cards
```

Path	Extension	Status	Lines added	Lines removed
DM3/src/Feature/DignityMemorial/Location/code/Views/Location/Detail/LocationRecentObituaries.cshtml	.cshtml	Modified	1	1

GITHUB

GitHub.com is a free online hosting provider for code that is tracked using Git

- Code is stored publicly (unless you want to pay extra)
- Very popular storage place for open-source projects
- The website for this class is hosted by GitHub!

BACKGROUND IMAGES

BACKGROUND COLOR REVIEW

```
p {
    background-color: gray;
    color: white;
}
```

This is a paragraph with the background color set to gray.

BACKGROUND IMAGES

Can set background of an element as an **image** (instead of a color) with the property `background-image`

The `value` is `url("path")`, where `path` is the **relative** or **absolute** path to where the image lives, like this:

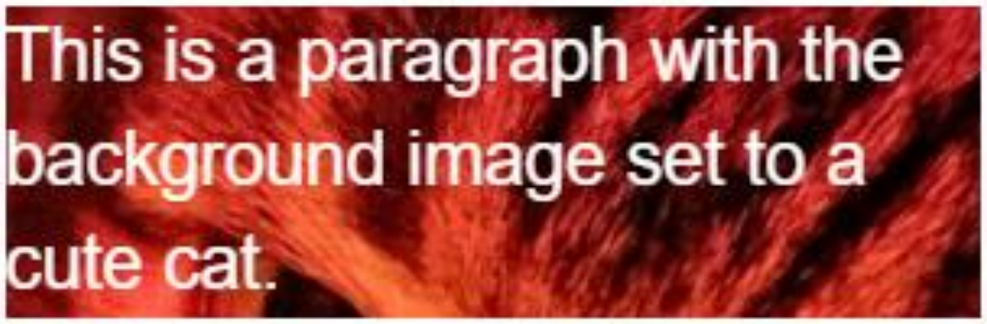
```
p {
    background-image: url("images/kitten.jpg");
    color: white;
}
```

BACKGROUND IMAGES

```
p {
    background-image: url("images/kitten.jpg");
    color: white;
}
```

The amount of image that displays in the background is calculated based on image size and container size.

- Make sure to resize images so that the part you want visible is within the “view window”
- Or…



BACKGROUND POSITION EXAMPLES

`background-position:` allows you to move a background image around within its container

- By default, an image is positioned at the top left side of the container

```
section {
        background-image: url("octopus.jpg");
        background-position: top left;
}
```

Image width: 600px by 800px

BACKGROUND POSITION EXAMPLES

Container width: 600px by 200px

```
background-position: top left;
```

```
background-position: center center;
```



```
background-position: bottom right;
```

BACKGROUND REPEAT

`background-repeat:` defines if (and how) the background image will repeat

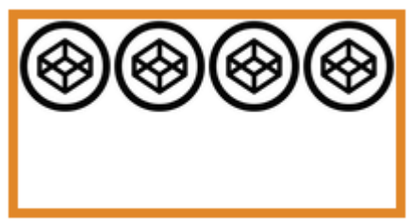
- By default, background images are repeated until they fill the entire container

```
p {
    background-image: url("codepen.gif");
    background-repeat: repeat;
}
```

BACKGROUND REPEAT

`repeat`: tile the image in **both** directions

`repeat-x`: tile the image **horizontally**



`repeat-y`: tile the image **vertically**

`no-repeat`: don't repeat, just show the image **once**

BACKGROUND ATTACHMENT

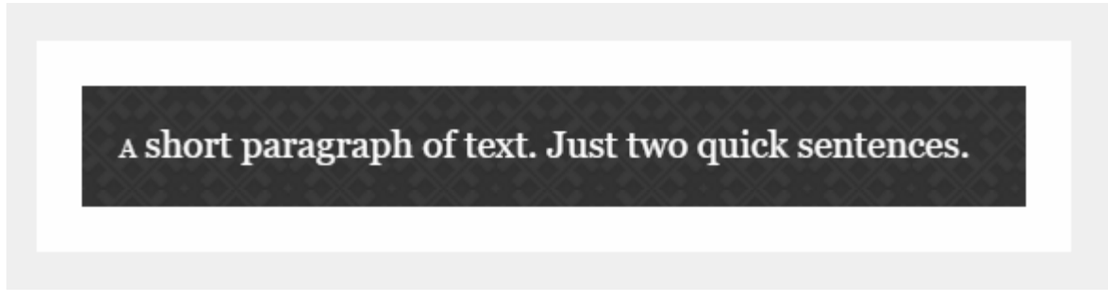
`background-attachment:` images usually scroll with the main view, but setting to `fixed` means the image stays in place when the user scrolls the page

- Difficult to describe, so check out this demo or this demo

```
section {
	background-image: url("pattern.png");
	background-attachment: fixed;
}
```

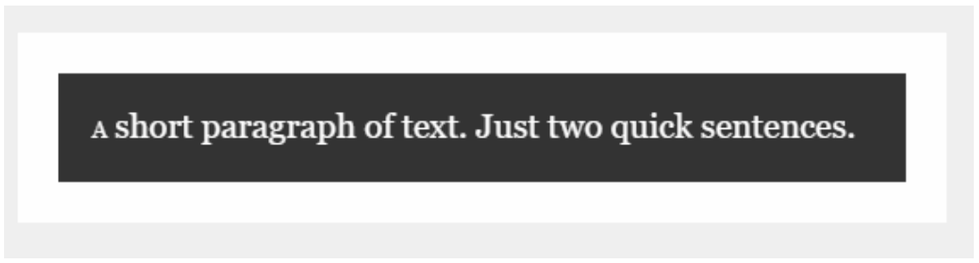
FALLBACK BACKGROUND COLOR

If your background image is dark and your text is light



You may want to specify a `background-color` in **addition** to a `background-image` so that content is visible while the image is loading

So instead of a “blank” area…

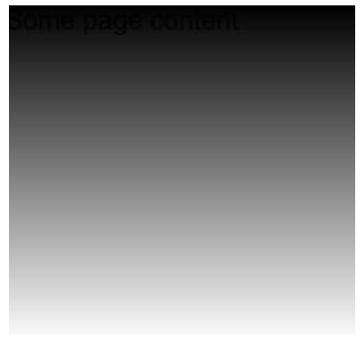


…the user can see content while the image downloads

BACKGROUND GRADIENTS

You can set `background-image` to `linear-gradient`, which is a gradient that the browser draws for you:

```
section { background: linear-gradient(black, white); }
```



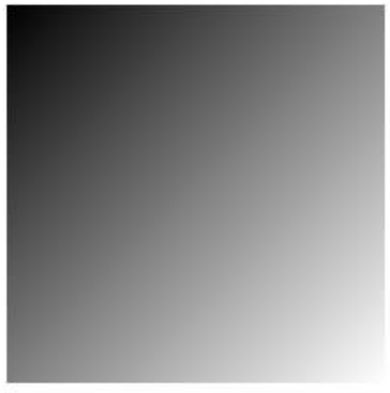
As many colors as you want can be blended, separated by commas:

```
section {
      background: linear-gradient(#ea992e, red, #9e5308);
}
```

BACKGROUND GRADIENTS

By default `linear-gradient` draws from top to bottom, but you can set the gradient to draw at an angle instead by starting with `to`

```
section { background: linear-gradient(to bottom right, black, white); }
```

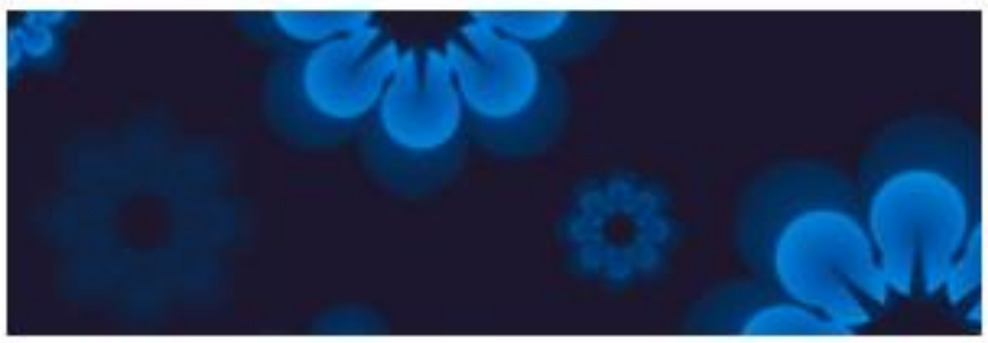


```
section {
        background: linear-gradient(to right, red, #f06d06, yellow, green);
}
```

BACKGROUND GRADIENTS

Background gradients can use rgba colors, meaning you can create a gradient that fades to transparent:

```
body {
        background-image: url("flowers.png");
}
```



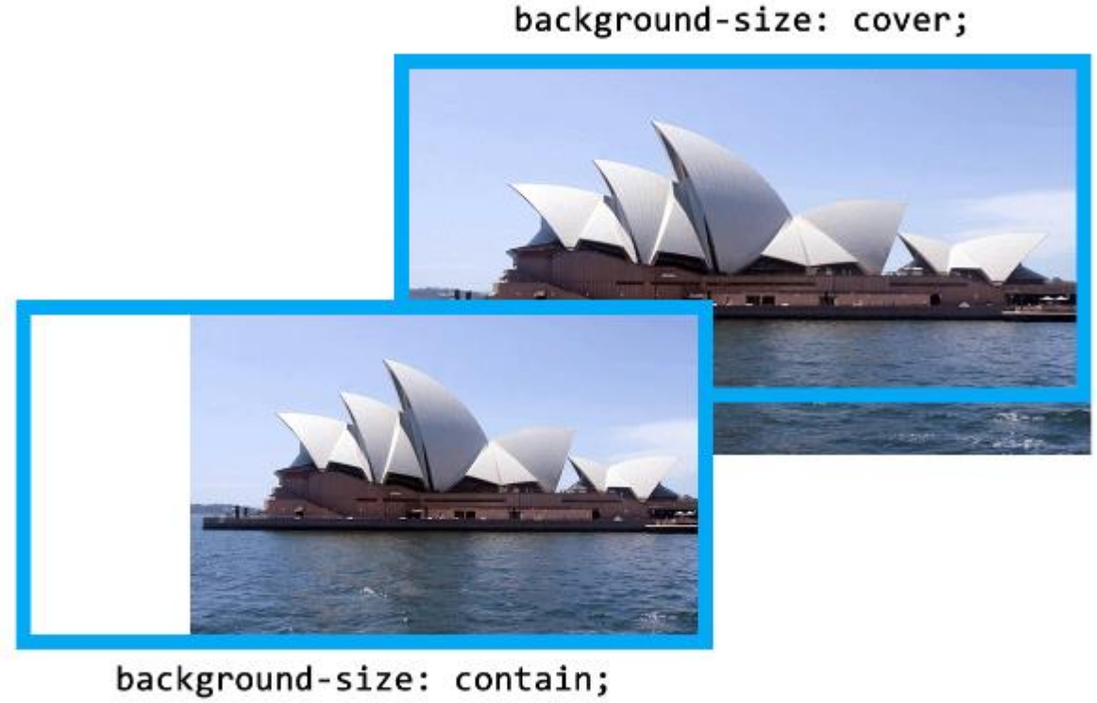
```
header {
        background-image: linear-gradient(to
right, rgba(255,255,255,0),
rgba(255,255,255,1));
}
```

BACKGROUND SIZE

`background-size:` specifies how much of the container that the image covers

`cover`: always cover the entire container (even if that means cropping an edge, or making the image bigger)

`contain`: always show the whole image (even if that means there is space on the sides or bottom)



HEIGHT AND WIDTH

To ensure that a background image fully displays, you can set the `height` (and/or `width`) attribute on the element using CSS:

```
header {
    background-image: url("images/hero.png");
    height: 600px;
}
```

HEIGHT AND WIDTH

`height` and `width` can be set on (most) elements to change how much room they take up on the page.

- We'll discuss later why elements like `<a>` and `<em>` don't change when you set their `height` or `width`

The `value` of this property must be a positive number.

- Units are either px or em
- Or you can specify a percentage

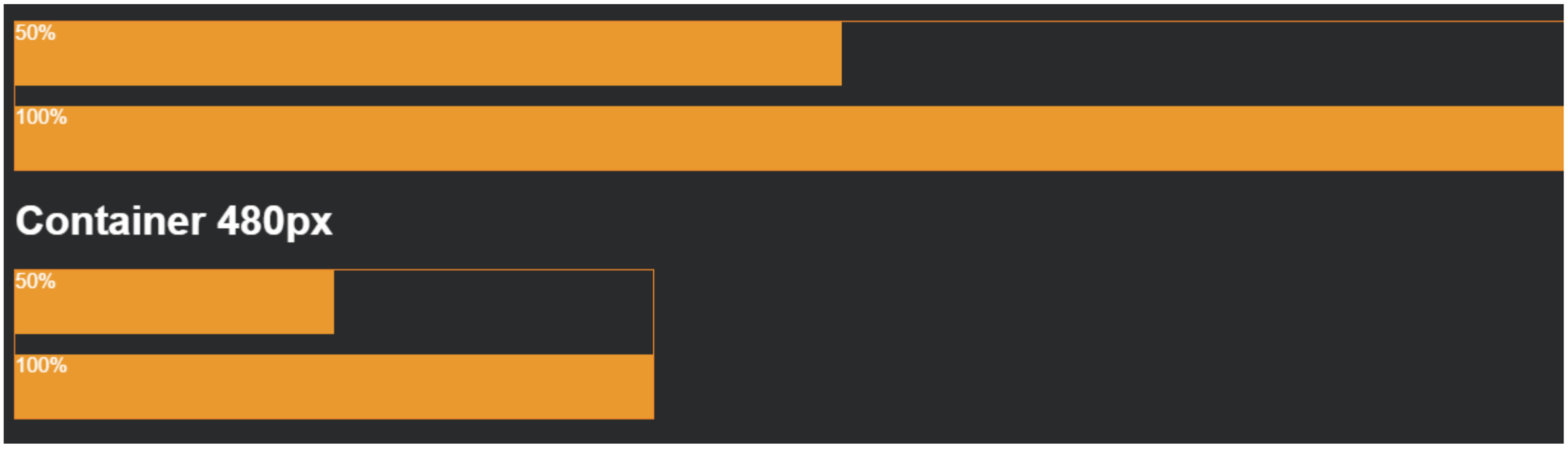
```
header { height: 6em; }
```

HEIGHT AND WIDTH %

Percentage is based on the element's **parent**'s width or height

```
section { width: 50%; }
```

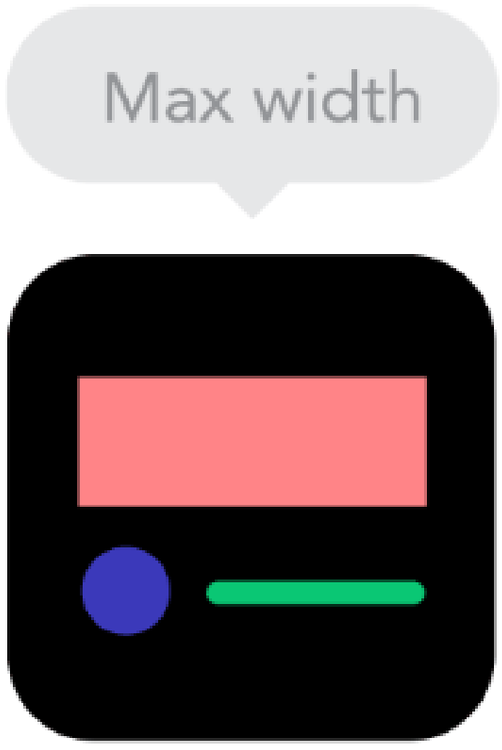
If that `section` were inside a 480 px wide container, it would end up being 240 px wide.



MAX-HEIGHT AND MAX-WIDTH

To ensure an element is **never larger** than a certain value, use `max-height` or `max-width`

- Typically used to make sure content (particularly text) doesn't spread too far out on large monitors

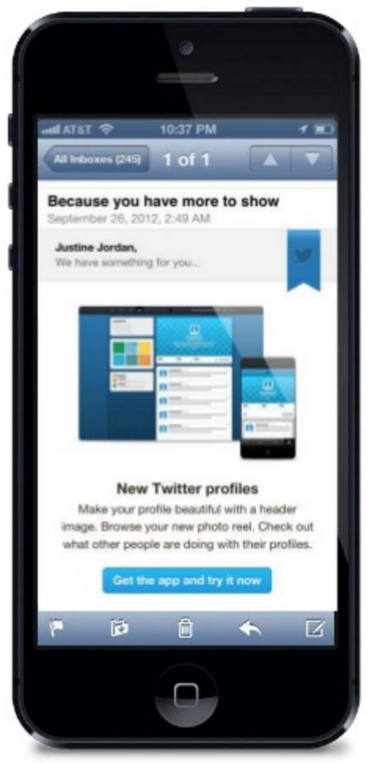


MIN-HEIGHT AND MIN-WIDTH

Specify `min-height` or `min-width` if you want to ensure an element is **never smaller** than a certain value.

- This is especially helpful if your size is “dynamic” (based on percentage) and will vary depending on device

```
img {
      width: 50%;
      min-width: 350px;
}
```



MIN-MAXING

`height` and `width` fix an element to a specific size regardless of display size

- If `width` is wider than the display – scroll bars
- If `width` is smaller than the display – content may wrap even if there is room

`min-height`, `min-width`, `min-height`, and `min-width` allow elements to change when the display size changes, but still allow some control over presentation.

MIN-MAXING

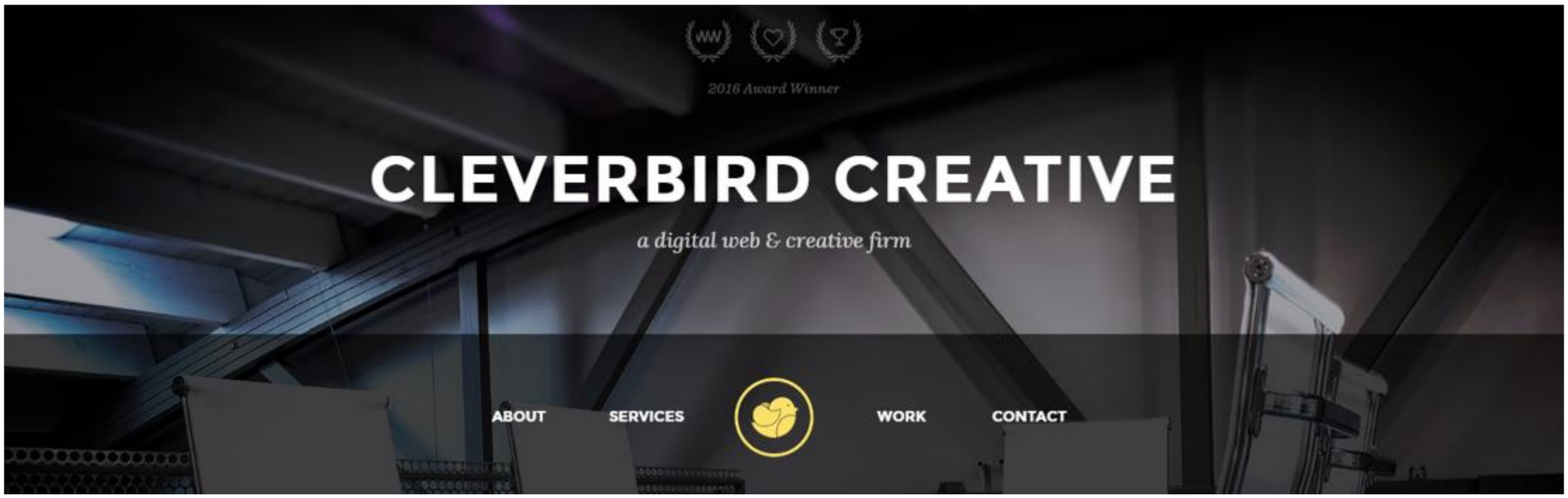
You can choose to set only `width` and/or `height`, only `min-width`/`min-height`, and only `max-width`/`max-height` – or any or all of them, depending on your design.

For example, this `section` will expand up to 500 px wide, and then get no bigger. If you shrink your browser, it will shrink until its 100 px wide, and then get no smaller.

```
section {
      min-width: 100px;
      max-width: 500px;
}
```

NOT ALL HEROES WEAR CAPES

A common use of `background-image` is to create a "hero" image with text overlaying it



PRACTICE TIME!

MAKE A HERO

Add a “hero image” to your site.

- Play around with a bunch of the background properties we learned to make your hero look pretty
- Try setting a `width` and `height`. What happens when you resize your browser window? Change to `min-width` – what changes?

DOMAINS & WEB HOSTING

What is a domain name?

The Domain Name System, or **DNS**, is like a phone book for the internet

It’s essentially a list that maps the location of files on a server (identified by a series of unique numbers called an IP Address) to a friendly name, like Wikipedia.org

How do I buy a domain name?

Companies called registrars manage the reservation of domain names

- GoDaddy is one of the largest registrars, but many smaller companies also provide this service

- ICANN is the agency responsible for regulating and accrediting registrars

DOMAINS & HOSTING

I bought a domain name… now what?

Registering a domain name maps that name to a location where the files will be hosted, but does NOT necessarily provide server space for your files

- GoDaddy will both register and host your website, as will many other companies

- You can buy a domain name from one company and host your files at another (or, host from a computer you own!)

DOMAINS & HOSTING

A common way to upload your website is using FTP (File Transfer Protocol).

You don't have to understand it, just find an FTP client you like and copy your files using the program.

Filezilla

Cyberduck

“HOMEWORK”

- Practice!

- Optional: read chapters 10-12 and chapter 16 of HTML and CSS: Design and Build Websites

- Check out the CSS Zen Garden for inspiration on how simply changing CSS can change the entire look and feel of a page