Lecture 13: CSS3, Accessibility, and Twitter Bootstrap

CS-546 – WEB PROGRAMMING

Code for this Week

Accessibility and CSS3:

- https://github.com/Stevens-CS546/CS-546-WS-Summer-1/tree/master/Lecture%20Code/lecture_13/accessibility_and_css3

Bootstrap:

- https://github.com/Stevens-CS546/CS-546-WS-Summer-1/tree/master/Lecture%20Code/lecture_13/bootstrap

Accessibility through Keyboard Navigation

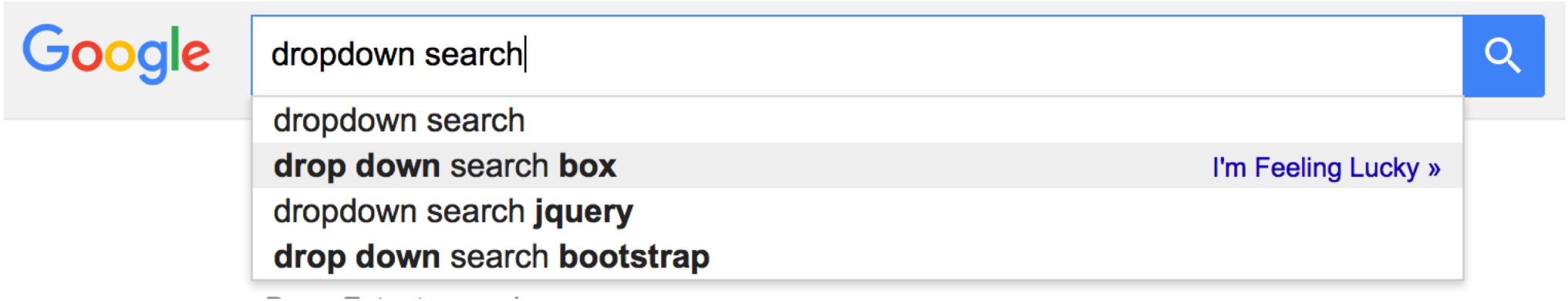
Complex frontends are hard to use

Many websites have search boxes that show results as you type in them.

Features like this are great; they save time, they are easy to use, and they simplify the process of using the web.

From an accessibility standpoint, to make this easy to use for *all* users, you need to do some work.

Features like this should be entirely usable via keyboard for accessibility reasons.



Case Study: live searches

The previous slide shows a ‘live search’, an adaptive search that shows results as they are found that you can select.

You can leverage the fact that your user is focused on an input to make the dropdown entirely keyboard navigable with ease!

- Add an event listener on that input for the *keydown* event
- Check which key was hit
  - if it was some sort of direction (up, down, left, right) then perform some logic to add a some indicator to a particular search result that is the selected
  - If it was `enter` then navigate to the selected search result, or a search page if none are selected
  - If it was `escape` then unselect any search results
  - If it was any other character, then simply re-query your search results and reset which result is the selected result

Using tabindex

The *tabindex* attribute allows you to focus on elements that can't normally be focused on, or change the order that you focus on elements

- This allows you to target an element using the :focus CSS selector

Your tabindex can be positive, negative, or 0:

- If it's positive, elements will be tabbed from the lowest to highest number of tabindex
- If it's 0, it will allow the element to be focusable but not change the order that you can tab into it; it would just be where it is in relation to the rest of the document (like static positioning)
- If negative, it will be ignored on tab but focusable
- Elements with an equal tabindex will be focused on in the order they appear in DOM

Case Study: reading comments on a website

Some websites are entirely devoted to reading comment chains, but they hide comments nested after a certain level.

While this will *probably* be entirely navigable through anchors, you would have to go through many anchors to get down to the first ‘load more comments’ factor.

Using a similar pattern as before, you can actually navigate through divs.

You would use the *tabindex* property, as well as some keyboard watching, to allow you to be focused on a particular post and navigate with the keyboard between posts.

Practical: Making a Rich Text Editor

Here's a fun fact – you can make certain elements *editable* using the *contenteditable* attribute.

We can use this to make a contenteditable div that is entirely stylable via the keyboard, rather than buttons!

To start off, we will make a simple Rich Text Editor that will allow you to open and close emphasized, strong, and underlined text.

Advanced CSS

Media Queries!

One of the most powerful things about modern CSS is that we can use media queries.

Media queries allow us to conditionally apply styles.

We can easily apply based on:

- width / max-width / min-width
- orientation (portrait vs landscape)
- media type (screen, print, all, speech)
- more!

The basic format is, in your stylesheet:

```
@media (min-width: 700px) {
    /* These rules will apply when the screen is 700px or wider */
   .nav { width: 650px; margin: auto;}
}
```

Print Layouts

You may use media queries to create styles that will apply to your page when it's printed only.

Often, you will use this to:

- Hide all but essential graphics
- Hide advertisements
- Create an easy to read version of your page that has little to no color
- Remove content that you would not want included, such as discussion comments

In your link tag, you can include a *media* attribute that describes which media type to apply those styles to.

- all
- print
- screen
- speech

Mobile First

You may be tempted at first to start changing your layout using media queries that change your layout as your screen size gets *smaller*.

Countless developers clocking in countless hours have come to the following conclusion: it's easier to define styles for your smallest screen, and add new styles that get applied as the screen gets larger.

An example, using a navigation that is a *nav > ul > li* setup:

- By default, your list items would display as blocks, each on their own line
- After a browser is 780px wide, they would float next to each other and display horizontally.

This is called *mobile first* development, in which you design for the smallest size first and support larger resolutions on top of that.

Transformations

In CSS, we can transform elements several ways:

- Translation
- Scaling
- Skewing
- Rotating

```
translated {  transform: translate(10px,-40px)}
.rotated {  transform: rotate(12deg);}
```

Animation

We can animate our elements rather easy, by defining and applying an animation:

```
.rotate_this_forever {
  -webkit-animation: infinite_rotation 2s infinite linear;
}

@-webkit-keyframes infinite_rotation {
  from {
    -webkit-transform: rotate(0deg);
  }

  to  {

    -webkit-transform: rotate(359deg);

  }

}
```

Hardware Accelerated CSS!

Many animations can be accomplished by using JavaScript to manually manipulate elements. You can make extremely complex animations that way by treating it like a video game script, where you manipulate things frame by frame.

This, however, is extremely poorly performant because you cannot take advantage of your computer's native hardware acceleration.

When you define an animation in CSS using certain CSS3 rules, your browser can understand exactly what you are trying to do and use hardware acceleration. This makes CSS only transformations extremely smooth.

By using certain CSS3 factors during animations, such as the translation rule rather than using left / right values we can force hardware acceleration.

Hardware-Accelerated Properties

Only the following rules can result in hardware acceleration being used

- opacity
- translate (or translateX, translateY, translateZ, translate3d)
- transform: scale;
- transform: rotate

Pseudo-Elements

Pseudo-elements allow you to target the content of elements in more complex ways.

- First line
- First letter
- Before
- After
- Selection
- Placeholder

Content

You can add content with CSS, as well!

This is useful for:

- Adding things to certain text; ie, quotation marks to block quotes.
- Creating an icon set
- Creating hovering tooltips

Combining Content and Pseudo-Elements

It is very common to combine the CSS *content* rule with pseudo-elements in order to add and style things before and after elements.

Commonly, you'll see:

- Adding quotation marks before blockquotes using the ::before pseudo-element that appear at the top left of an element, and one at the bottom right using the ::after pseudo-element
- Creating icon sets
- Creating clearfixes

Transitions

When CSS properties change, we can set transitions that target those properties to make them occur over a period of time.

For example, you can set the font-size of an object to change over time when a class is added:

```
.big-text {
  font-size: 60px;
  transition: font-size 2s ease-in-out;
}
.big-text.smaller {
  font-size:  30px;
}
```

Bootstrap

What is Bootstrap?

Bootstrap is a front-end framework.

- A front-end framework is any tool that is designed to make websites and applications easy to develop

Bootstrap was created for rapid development and internal standardization of internal tools at Twitter.

Bootstrap is now the most popular front-end framework and used all over the web.

What does Bootstrap provide?

Bootstrap provides an aesthetically consistent design that is easily customizable, along with many very important complex components

- Dropdowns
- Modals
- Tabs
- Tooltips
- Popovers
- Alerts
- Carousels
- Collapsible Content

Why is Bootstrap helpful?

Bootstrap allows developers to speed up a great deal of time in development by giving them a base of consistent styles and components that commonly appear in web applications

- No need to reinvent the wheel

It also is extremely customizable, so you can easily integrate your companies colors, fonts, etc into Bootstrap.

Why do so many websites use Bootstrap?

Many websites use Boostrap because having a unique design is not particularly necessary for every website.

Websites are, first and foremost, about transmitting information. Do the following websites really need to have a 100% unique design?

- Every project demonstration page to show off some cool JavaScript
- Every hair salon, restaurant, or small business in the world
- Every startup that is selling a non web-based-product

People are very familiar with the general look and feel of a Bootstrap landing page for a website, and familiarity gives users at comfort

- See, Human-Computer Interaction / User Experience course for more details

How can we use Bootstrap?

For the remainder of the course, you may use Bootstrap as the start of your styles to start off with a clean, consistent design.

For the final project, you will still have to demonstrate knowledge of CSS; this means that you must include enough custom CSS to demonstrate that you understand how it is used and how it works.

Setting up Bootstrap

For the sake of simplicity, we can use a CDN to setup Bootstrap

In your head:

- <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css" integrity="sha384-1q8mTJOASx8j1Au+a5WDVnPi2lkFfwwEAa8hDDdjZlpLegxhjVME1fgjWPGmkzs7" crossorigin="anonymous">

Before the body closing tag, and after jQuery is loaded:

- <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js" integrity="sha384-0mSbJDEHialfmuBBQP6A4Qrprq5OVfW37PRR3j5ELqxss1yVqOtnepnHVP9aJ7xS" crossorigin="anonymous"></script>

Bootstrap depends on jQuery and its JavaScript components will not run if jQuery is not loaded first!

Bootstrap Concepts

Branding

Bootstrap takes the concept of *branding* to a very high level; it makes sure that the following are consistent across the entire design.

By compiling Bootstrap, you can set what the colors related to each suffix will be.

Very often, you will see classes used that end with a branding related suffix such as:

Suffix	Use
`-primary`	A class that ends this suffix will generally follow the primary color for the brand, such as a call-to-action button for your product
`-success`	Used to represent a successful action; ie, *your changes have been saved.*
`-info`	Used to represent generic info; ie, *this section relates to your family history.*
`-warning`	Used to warn the user about an imminent action / potential issue; ie, *saving this is irreversible*
`-danger`	Used to represent that an error of some sort has occurred; ie, *you cannot divide by 0.*

Supported Sizes

	Extra small devices Phones (<768px)	Small devices Tablets (≥768px)	Medium devices Desktops (≥992px)	Large devices Desktops (≥1200px)
`.visible-xs-*`	Visible	Hidden	Hidden	Hidden
`.visible-sm-*`	Hidden	Visible	Hidden	Hidden
`.visible-md-*`	Hidden	Hidden	Visible	Hidden
`.visible-lg-*`	Hidden	Hidden	Hidden	Visible
`.hidden-xs`	Hidden	Visible	Visible	Visible
`.hidden-sm`	Visible	Hidden	Visible	Visible
`.hidden-md`	Visible	Visible	Hidden	Visible
`.hidden-lg`	Visible	Visible	Visible	Hidden

Print Styles!

Bootstrap comes with 4 classes that allow you to manipulate how something is rendered when printed

- These prevent your recipe website with comments from destroying an ink cartridge
  - http://coffeegrammer.com/cheesecake-printers-and-morality/

Classes	Browser	Print

`.visible-print-block`
`.visible-print-inline`
`.visible-print-inline-block`
	Hidden	Visible

`.hidden-print`	Visible	Hidden

Accessibility In Bootstrap

Bootstrap includes many classes that are used for accessibility reasons to perform simple tasks.

Many colors chosen by default in Bootstrap had accessibility in mind with respect to color contrast, however some things such as preformatted code have a low color contrast.

You will have to correct these issues.

Suffix	Use
`sr-only`	An element with this class will only be visible to a screen-reader.
`sr-only-focusable`	Will be focusable by a screen reader and visible to a screen-reader on focus, useful to make links to jump to content.
`text-hide`	Text inside an element with this class will not be shown visually, but will appear to screen readers.

Compiling Bootstrap

Bootstrap is actually written in a CSS-like language called LESS, which compiles into CSS.

As such, you can download Bootstrap's source, edit variables which define things like branding color, default text size, etc. and make a custom version of Bootstrap.

This also allows you to include Bootstrap's source into your custom code and only include styles relevant to your application.

The Grid

What is a grid?

In graphic design a grid is a series of guide lines used to structure content.

In web development, designs are often created against grids; this means that designers setup the content to fall within a certain number of columns.

Nowadays, it is common for grids to be a combination of fluid and fixed widths; the max size of a row will change with the resolution of a screen, whereas the rows will always be divided into columns of equal width.

Bootstrap's Grid

Bootstrap has a very simple grid that can adapt to the current screen size.

Bootstrap's grid allows you to setup rows with content, where each content element is set to take a certain number of columns in each row.

Most importantly, it allows you to setup how many columns your content will use at different resolutions.

For example, you could have a gallery where images take up:

- Hidden when the screen is < 768px
- The entire width of the row when the screen is between 768 and 991px
- Half the width of the row when the screen is between 992px and 1200px
- One third the width row when the screen is 1200px or wider.

This can all be accomplished by having a combination classes that dictate how many columns something should take up at particular sizes.

Using the Grid

You can use your grid at all levels of your content. For example, you could structure a blog as:

- div.container
  - div.row
    - div.col-sm-12.col-md-8.col-lg-6.blog-post-container
      - div.row (10 times repeated, for 10 posts)
        - div.hidden-sm.col-md-2.avatar
        - div.col-sm-12.col-md-6.description
          - h3.title
          - p
    - div.col-sm-12.col-md-4.col-lg-4.col-lg-offset-2.about-me

The **container** class is important, as it acts as a location for your grid to sit; rows need to be contained in an element with *container* or *container-fluid* class for proper alignment / padding.

Nesting rows and columns

Each row element has multiple columns inside of it, but you can nest rows inside of those columns.

Bootstrap's grid system is fairly fluid, so you can keep nesting, but at some point columns will get to be too small to be really useful.

Common Layout Components

What does Bootstrap do for normal HTML?

Bootstrap gives an aesthetically pleasing, default, and sensible style to most common elements. It sets up a series of rules for typography, padding, margins, and so on that make the layout proportional and related in many ways, so that even a barebones document will look logical and well laid out without any fancy components.

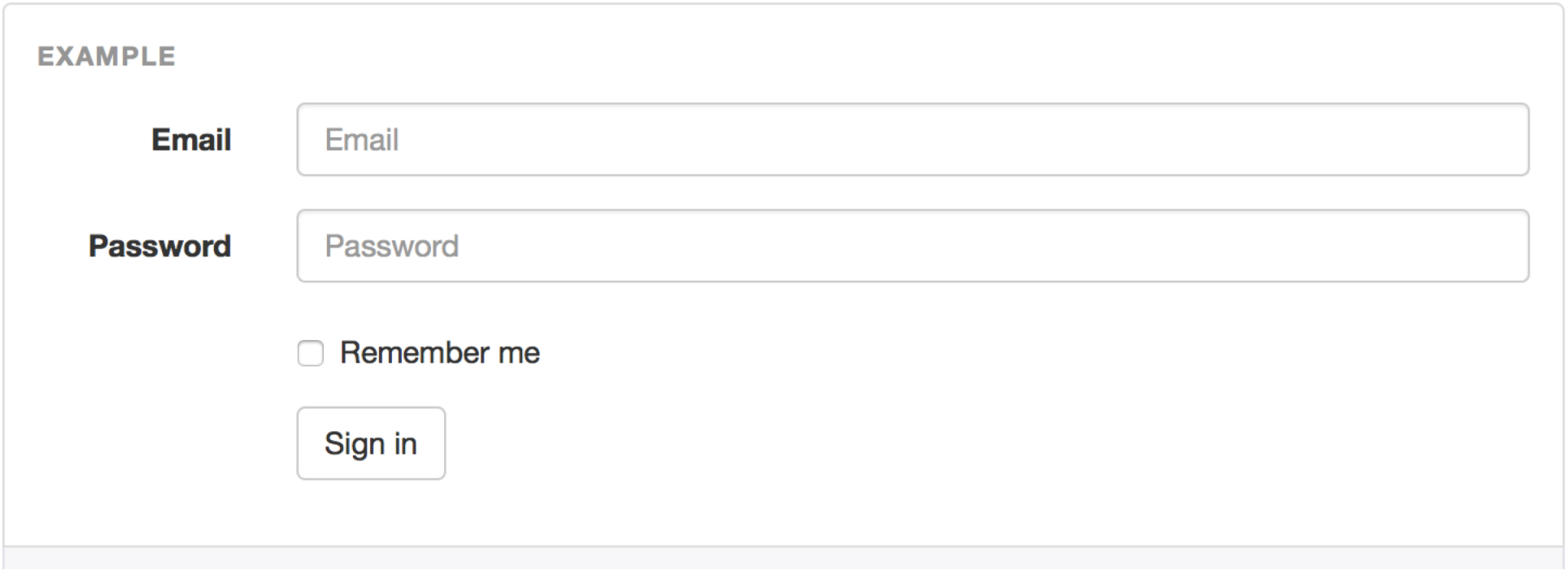
It also provides a series of classes to use in order to make it easy to build out basic and common layout, such as:

- Better looking forms
- Consistent Buttons
- Prettier tables
- Utilities for images
- General utilities

Forms in Bootstrap

Bootstrap provides several utility classes that can be used to group forms, their input, and their labels together to make horizontal or vertical forms.

- http://getbootstrap.com/css/#forms



Buttons

Bootstrap provides a number of classes to make anchors, buttons, or inputs appear as pretty buttons.

This is useful because while aesthetically it may make sense to have something that *looks* like a button, each element is used differently.

- You could have an anchor that looks like a button as a *call to action* to go view a different page
- You could need an input to submit a form that *looks* like a button that is consistent with all your other buttons
- You could want to use an actual *button* to interact with non-form content on your website.

With bootstrap, you can style all of these elements consistently and beautifully.

- http://getbootstrap.com/css/#buttons

Tables

Tables are particularly difficult to style, since they are very particular based on their content.

Bootstrap provides a number of classes to make tables look better, but most importantly, it provides a way to make a responsive table that will scroll horizontally on mobile browsers; normally, tables would squish themselves and break.

Simply wrap your table in an element with the class *table-responsive* to create a responsive table.

- http://getbootstrap.com/css/#tables

Image Utility Classes

There are several utility classes for images that cover common functions designers have developers implement for images:

- http://getbootstrap.com/css/#images

Class Name	Effect
`img-responsive`	Makes an image max its width at the width of its parent element and scale its height accordingly.
`img-rounded`	Rounds the corners of an image
`img-circle`	Makes the image appear inside of a circle, removing contents off the edges
`img-thumbnail`	Gives the image a border with rounded corners, some padding of empty space, and then shows the image as normal

Utility Classes

Bootstrap provides many, may utility classes to fulfill common needs:

- http://getbootstrap.com/css/#helper-classes

Class Name	Effect
`text-(primary muted success info danger warning)`	Formats the text to take on the color of the branding corresponding to the class; ie, `text-primary` will take on the primary color.
`bg-(primary muted success info danger warning)`	Formats the element to have the same background color as the respective branding.
`pull-left, pull-right`	Floats the content left or right
`center-block`	Centers the content block (not the text0
`show`	Displays an element
`hidden`	Hides an element
`text-hide`	Hides the text in an element, but allows the element to maintain is size

Bootstrap Components

Bootstrap Specific Components

Bootstrap implements several common and more complex use cases in web design and development and includes a JavaScript library to make these components interactive

These are extremely useful, since they allow you to jump right into application development rather than reinventing these components.

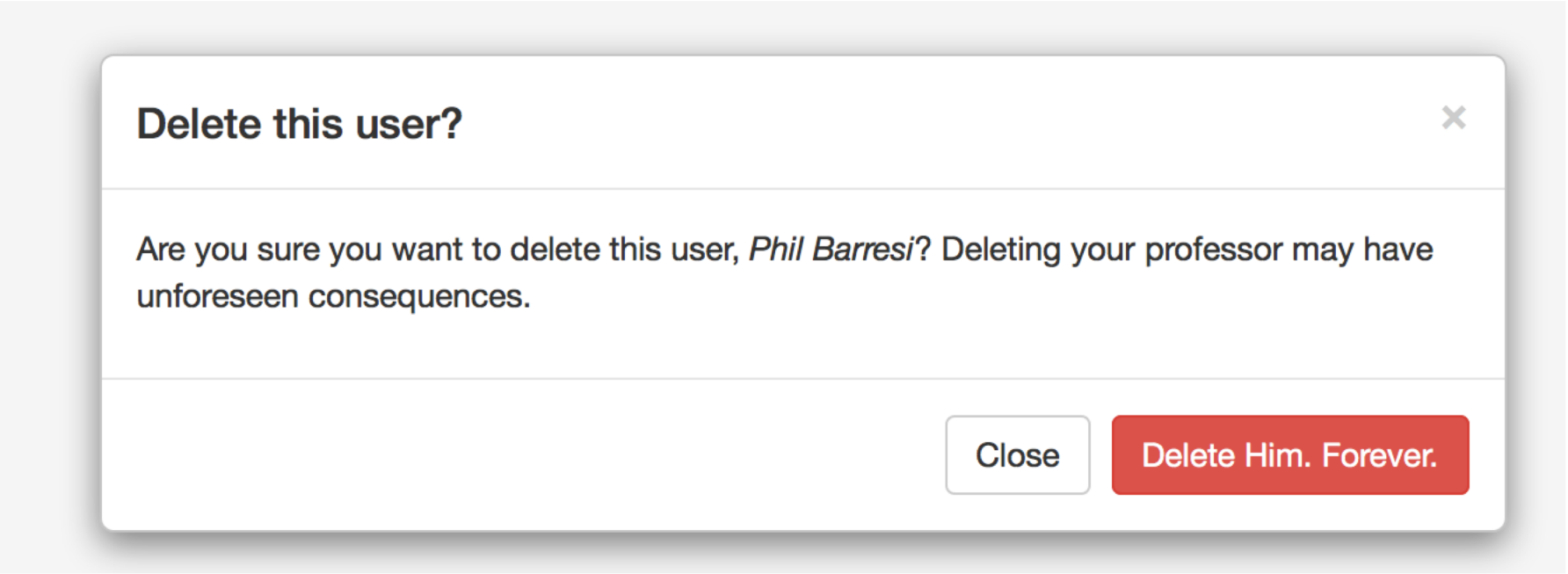
- Modals
- Alerts
- Tabbed Content
- Dropdowns
- Tooltips

Most of these are built with JavaScript to make them interactive

Modal Windows

A modal window (nowdays, just modal) is a window that pops up over the content of your website in order to provide some information or demand some action.

- http://getbootstrap.com/javascript/#modals

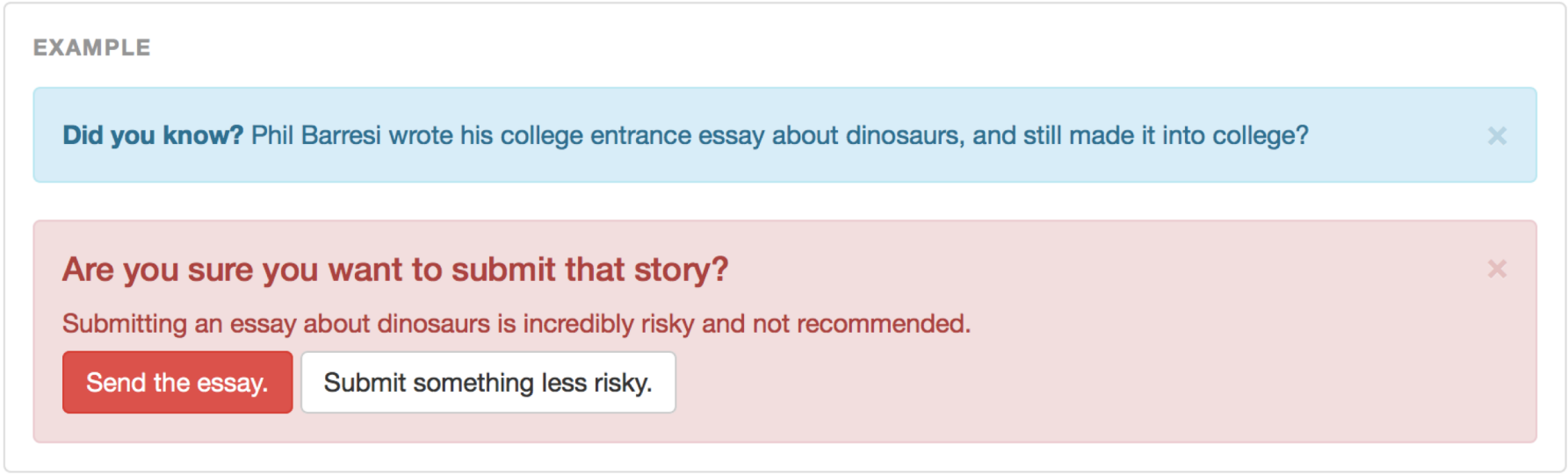


Alerts

It’s often useful to have a notification to appear that tells the user something went wrong, something went right, or just general information.

Bootstrap has classes to make style the alert messages, but also provides an API for closing them. You must place your elements in a specific order to achieve this.

- http://getbootstrap.com/javascript/#alerts

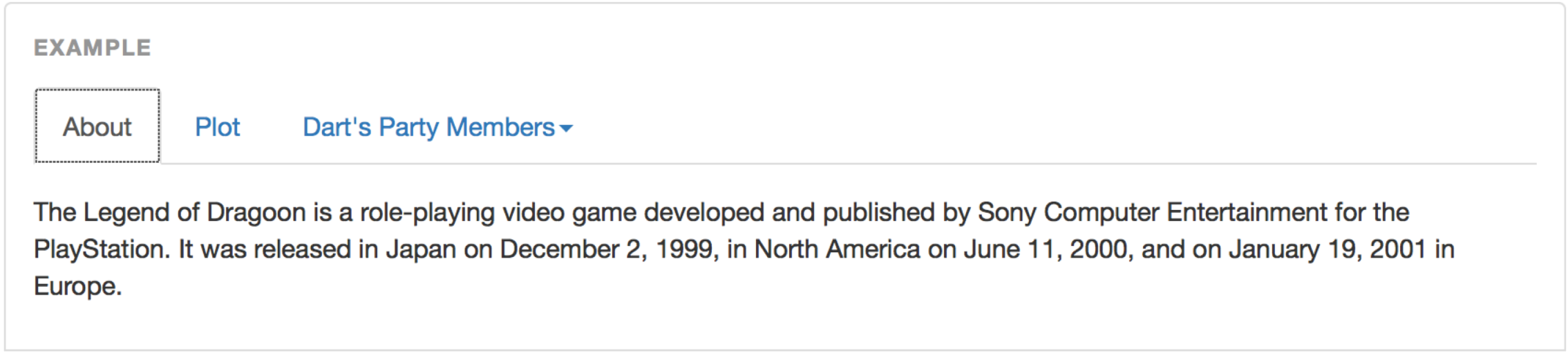


Tabs

Tabs allow you to setup content that is controlled by which tab is current active; you can view it as a way to navigate between a small amount of related content.

- http://getbootstrap.com/javascript/#tabs

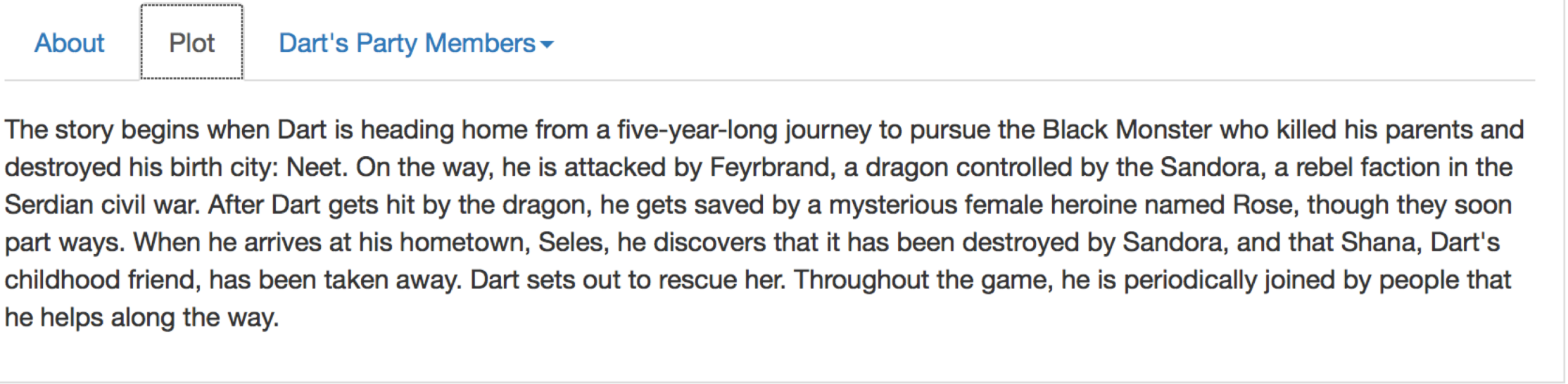
For example, if you were creating a website about video games, you may want to show the following on your page about *The Legend of Dragoon:*



Tabs

When you click on the ‘plot’ link, it would change to show the content related to the plot.

EXAMPLE



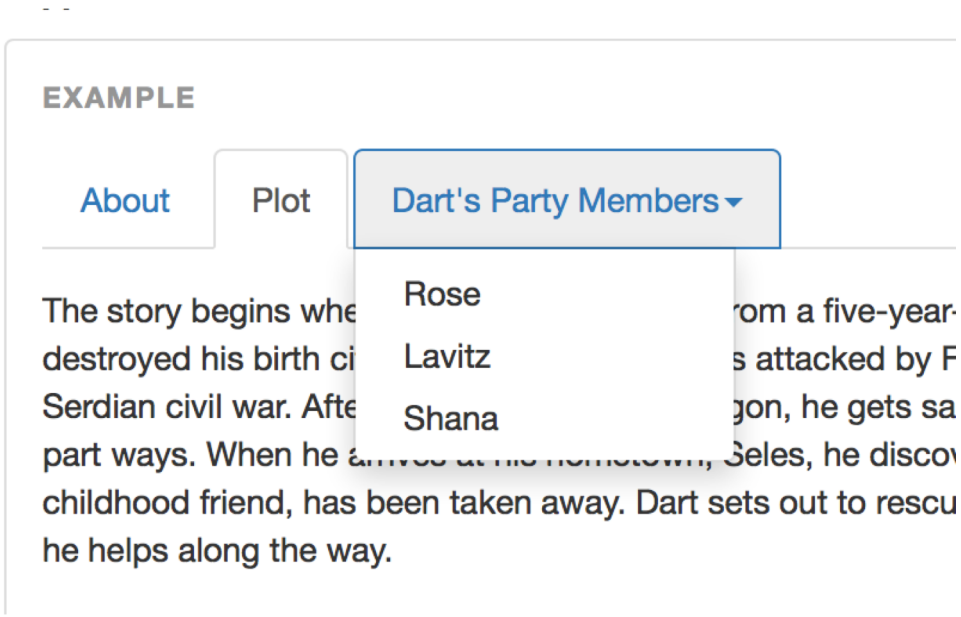
Dropdowns

Many elements can be turned into Dropdown menus by adding a few classes and data attributes.

To implement a dropdown, you need 3 elements:

- A parent element with the class *dropdown*
- An element that, on click, will trigger the dropdown to appear; this must have the attribute *data-toggle="dropdown"*
- An unordered list with the class *dropdown-menu* after the toggler.

http://getbootstrap.com/javascript/#dropdowns



Tooltips

Tooltips are easy to make, but hard to make *well*; there are many edge cases that have to be considered, and are actually performance intensive.

- **Because of this, Bootstrap requires you manually enable tooltips through the use of JavaScript**
- http://getbootstrap.com/javascript/#tooltips

You can add a tooltip to any element, and make it appear on the left / right / above / below the element.

EXAMPLE

Events

Many of these components are built out using a fair deal of JavaScript.

Because of these, there are a number of events you can watch for in JavaScript, such as:

- Modals opening or closing
- Tooltips being shown or hidden
- Tabs being hidden or made visible
- Dropdowns being opened or closed
- Alerts being closed, or after they are closed

What now?

Learn how to test!

Learning how to write frontend and backend unit tests are absolutely important to your future.

You can use PhantomJS to programmatically emulate a browser and write tests that way

- http://phantomjs.org/headless-testing.html

Mocha and Chai are commonly used for Node Testing

- http://chaijs.com/
- https://mochajs.org/

Research other databases!

There are literally entire courses filled with database information. There are even special *types* of databases, such as ElasticSearch, that fulfill limited sets of tasks such as text indexing.

You should familiarize yourself with tradition, SQL based databases such as MariaDB

- https://mariadb.org/

And it's always nice to know how to handle 'Big Data' with ElasticSearch

- https://www.elastic.co/

Experiment, and learn about everything!

You take a look at a topic or technology you may be interested in and build a dummy application

- Pick a frontend
- Pick a backend
- Look into unit tests for both, and learn how to write those!

In order to learn a new technology, I suggest simple tasks like the following:

- ToDo lists
- Comment Boxes / Blogs
- RSS Readers
- Shopping websites
- Anything!

Just keep building!