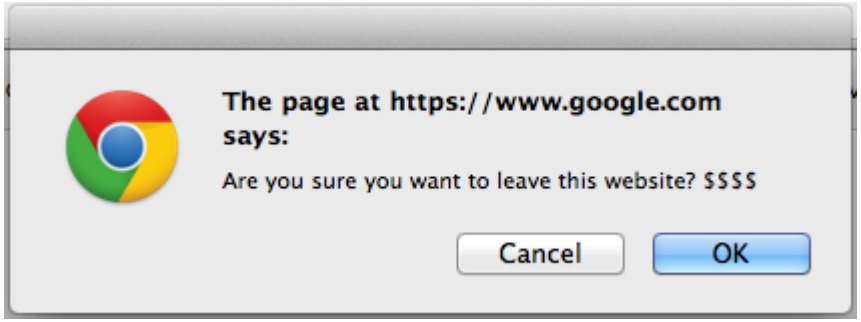
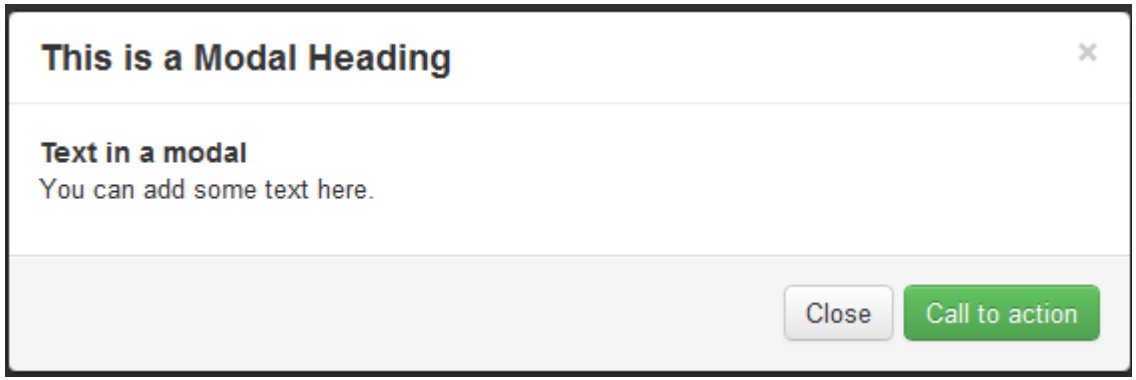
FE101

Building a Modal, One Page Websites

What is a modal?

An HTML/CSS modal is a much nicer way of representing the default JavaScript confirm box





First Up: HTML

- We'll need the following:
  - A wrapper for the modal to make the rest of the page seem "grayed out"
  - A `<div>` for the modal box itself so we can give it some padding and style
  - A button or link within the modal box to close the modal and two buttons or links to represent choices

HTML Syntax

```
<div class="modal-wrapper">
  <div class="modal">
    <a href="#" class="exit">X</a>
    <h3>Are you sure you want to quit?</h3>
    <a href="#" class="yes">Yes</a>
    <a href="#" class="no">No</a>
  </div>
</div>
```

Next Stop: CSS

for the modal wrapper

- In order to make our modal take over the whole screen wherever we are on it while being over layed across the current content, we'll use `position: fixed;`
- We'll want this overlay to take up the whole length and width of the page, hence `width: 100%;` and `height: 100%;`
- To achieve the "grayed-out" look, we can use an `rgba` color value and set a low alpha value or the `opacity` CSS attribute

Necessary CSS

```
.modal-wrapper{
  position: fixed;
  top:0;
  left:0;

  background-color: rgba(155,155,155,.6);

  width:100%;
  height:100%;
  padding: 0;
  margin: 0;

  display: none;
}
```

Next Stop: CSS

for the modal itself

- If we use `position: relative;` and `top: 50%;` we can vertically center our modal
- Set a fixed width so we can use `margin: 0 auto;` to center the modal horizontally
- Make sure to make it look pretty with `padding`, `border-radius`, and `background-color`!

Necessary CSS

```
.modal{
  position: relative;
  top: 50%;
  width: 500px;

  padding: 30px;
  margin: 0 auto;

  background-color: white;

  border-radius: 10px;
}
```

Next Stop: CSS

for javascript things

- `.modal-on` is simply used to set the body element's overflow to hidden when the modal is activated

- `.exit` is floated right so our "x out" is on the right side of the modal

Necessary CSS

```
.modal-on{
  overflow: hidden;
}

.exit{
  float: right;
}
```

JavaScript/JQuery

```
$(document).ready(function(){
  $(".activate-modal").click(function(e){
    e.preventDefault() #dont follow the link
    $(".modal-wrapper").show() #show our modal
    #add the .modal-on class to <body>
    $("body").addClass("modal-on")
  })
  $(".exit").click(function(e){
    e.preventDefault()
   #basically the opposite of activate
    $(".modal-wrapper").hide()
    $("body").removeClass("modal-on")
  })
})
```

Exercise

Now that we've built a modal together, try building one on your own!

15 Minute Break

What is a “one page website”?

Loosely defined, a site that has all of the content on one page, with the links only to scroll you to different parts of the page.

Some examples from onepagelove.com:

http://remysclippa.com/

http://www.visage.co/

http://www.nerdlab.be/

Page Elements

- Fixed navigation bar, typically after the first page only
- Separate “pages”
- Footer with contact information, copyright notice

Navigation Bar - HTML

```
<div class="nav-wrapper">
  <nav>
    <img src="logo.png" width="200">
    <ul>
      <li>
        <a href="#about">about</a>
        <a href="#contact">contact</a>
      </li>
    </ul>
  </nav>
</div>
```

Navigation Bar - CSS

```
.nav-wrapper{
  border-bottom: 4px solid black;
  padding: 10px;
  background-color: #fff;
  width: 100%;
}

nav{
  display: inline;
}

nav ul{
  display: inline;
  float: right;
}

nav ul li{
  display: inline;
}

nav ul li a{
  text-decoration: none;
  font-size: 20px;
  margin-right: 20px;
  font-weight: bold;
}
```

Navigation Bar - JavaScript

- Let’s create some JavaScript that will make our navigation bar fixed if the user scrolls past a certain point on the page

- This way, the navigation bar follows them around regardless of where they are on our one-page website!

Navigation Bar - JavaScript

```
$(document).ready(function(){
  $(window).scroll(function(){
    if($(window).scrollTop() > 900 &&
      $(".nav-wrapper").css("position") != "fixed"){
      $(".nav-wrapper").hide(function(){
        $(".nav-wrapper").css("position", "fixed")
        $(".nav-wrapper").slideDown('1000')
      })
    }else if($(window).scrollTop() <= 900 ){
      $(".nav-wrapper").css("position", "initial")
    }
  })
})
```

Separate “Pages”

To achieve the “separate pages” effect, simply wrap pages with different content in separate sections with a different background-color!

Separate “Pages” - Navigation

- There are two different approaches to a one-page website navigation:
  - Use anchor links to link to each separate “page” of the website.
  - Use a JavaScript library to fluidly scroll to your “pages”

Anchor Link Approach

Give your different “pages” an id, like so:

```
<div id=”page-two”></div>
```

Then, link to those “pages” from the nav bar:

```
<a href=”#page-two”>Page Two</a>
```

JQuery Library - $.scrollTo

https://github.com/flesler/jquery.scrollTo

Once this library is included in your HTML file, you can scroll to anchors on the page fluidly like so:

```
$.scrollTo( '#about', 800);
```

```
$.scrollTo( <anchor name>, <scroll time>);
```

JQuery Library - $.scrollTo

You'll probably want to enclose the code to scroll to different parts of the page inside of a click event for your link:

```
$(".some-link").click(function(){
   $.scrollTo( '#about', 800);
})
```

Hybrid Approach

You could use the anchor link approach and this simple JavaScript snippet to have a smooth scrolling experience:

http://css-tricks.com/snippets/jquery/smooth-scrolling/

Exercise

Build your own one-page website or attempt to emulate one of the sites on onepagelove.com

Have fun!