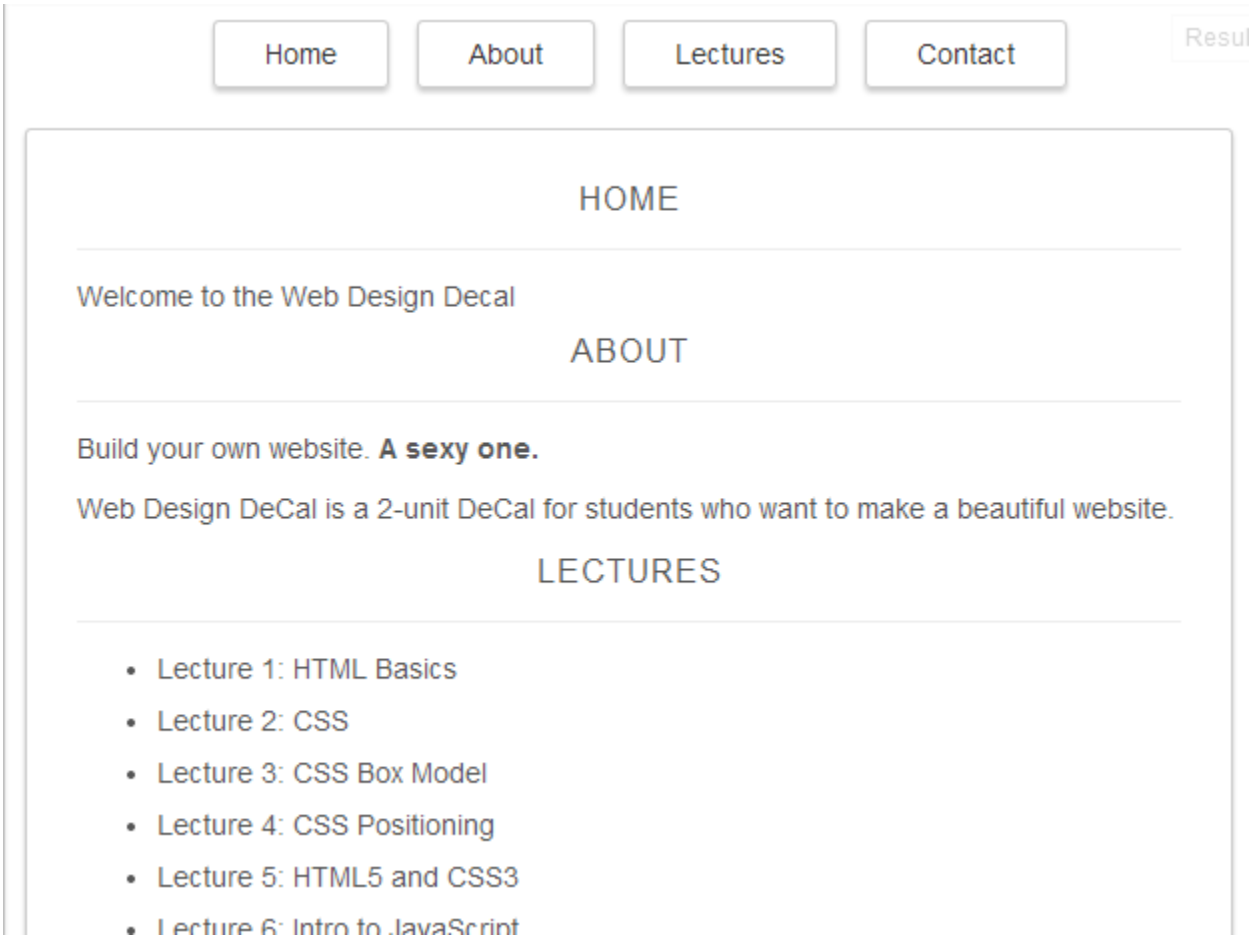
"The Web Design Workshop" Decal – Spring 2014
Programming Handout 8

Goals:

1. Displaying a Panel
2. Changing the Button CSS

1 – Displaying a Panel

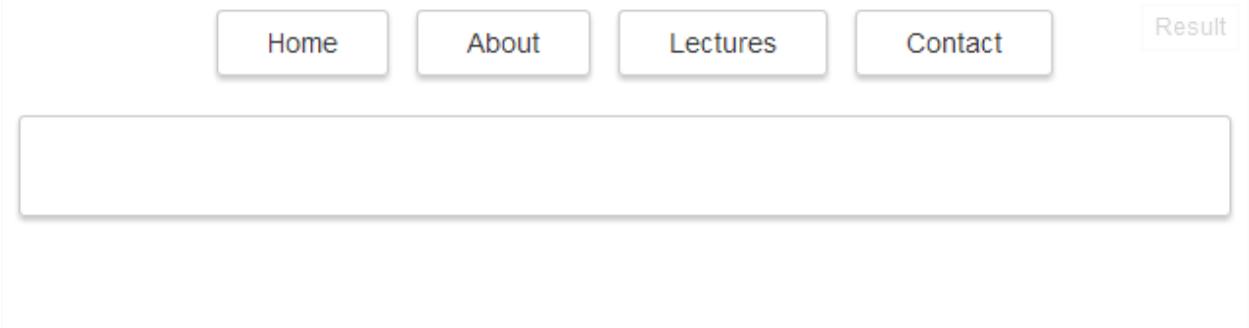
1. Go to http://jsfiddle.net/8xKn8/1/ . You will see this result in the bottom-right corner:



Notice how all the sections are clumped together in one giant container, **#panels.**

Our goal will be so that when you click one of the 4 buttons (Home, About, Lectures, or Contact), the corresponding panel will be shown and none of the others will.

2. First, let's modify the CSS. Find the style for **#panels .panel** and add the CSS style **display: none;** This will make all the elements with class **.panel** to be hidden (and we have 4 such **.panel** elements). We get:



3. Now that our panels are hidden, we can start using jQuery to display them when the corresponding button is pressed.

   What should we apply our click event to?

   We will apply a click event to our **.button** element.

```javascript
$(document).ready(function() {
    $('.button').click(function() {

    });
});
```

4. Next, notice that the parent element for **.button** is wrapped in a *<a>* element. This is an anchor tag, or link. Notice in the **href**, instead of putting a URL, we replaced it with an ID for a panel.

```html
<div id="buttons">
    <a href="#home">
        <div class="button">Home</div>
    </a>
    <a href="#about">
        <div class="button">About</div>
    </a>
    <a href="#lectures">
        <div class="button">Lectures</div>
    </a>
    <a href="#contact">
        <div class="button">Contact</div>
```

   This will be used for a jQuery trick so that when we click a **.button**, we checked the current button's parent element (<a>) and grab the text in href (which will be an ID). We will then put a jQuery selector on this ID and fade in that element.

First, let's grab the href value from the current element **.button** using DOM traversal and **.attr(..)**. How would we do this?

```html
<div id="buttons">
    <a href="#home">
        <div class="button">Home</div>
    </a>
    <a href="#about">
        <div class="button">About</div>
    </a>
    <a href="#lectures">
        <div class="button">Lectures</div>
    </a>
    <a href="#contact">
        <div class="button">Contact</div>
    </a>
```

```javascript
$(document).ready(function() {
    $('.button').click(function() {
        $('.button').parent().attr('href');
    });
});
```

Notice what we just did. For a **.button** element, we go find its parent (<a>) and grab the parent's href value. Just 2 problems: (1) We did **.parent().attr('href')** on *all* **.button** elements when clicked when it should just be the *current* .button element. (2) We grabbed the href, but that's it! We haven't used it or saved it, and that means our line is useless.

5. Let's modify this line by replacing **$('.button')** with **$(this)** and storing this line in a variable, say **var panel**.

```javascript
$(document).ready(function() {
    $('.button').click(function() {
        var panel = $(this).parent().attr('href');
    });
});
```

Much better. Thus if we click on the About button:

```html
<a href="#about">
    <div class="button">About</div>
</a>
```

Our jQuery will trigger the **.button** click event, find the element's parent's href value (#about) and store it in var panel. So at this moment, our JavaScript variable **panel** contains the String **#about**.

If you look further in the HTML portion of the JSFiddle, you can find an element with ID **#about**.

```
    </div>
    <div id="about" class="panel">
        <div class="title">About</div>
        <div class="content">
            <p>Build your own website. <strong>A sexy one.
</strong></p>
            <p>Web Design DeCal is a 2-unit DeCal for
students who want to make a beautiful website.</p>
        </div>
    </div>
    <div id="lectures" class="panel">
        <div class="title">Lectures</div>
```

Right now, it is hidden because we applied a display: none CSS style to the elements with **.panel** class. However, because we now have the ID of the About panel when clicking on the About button, we can simply make **#about** show!

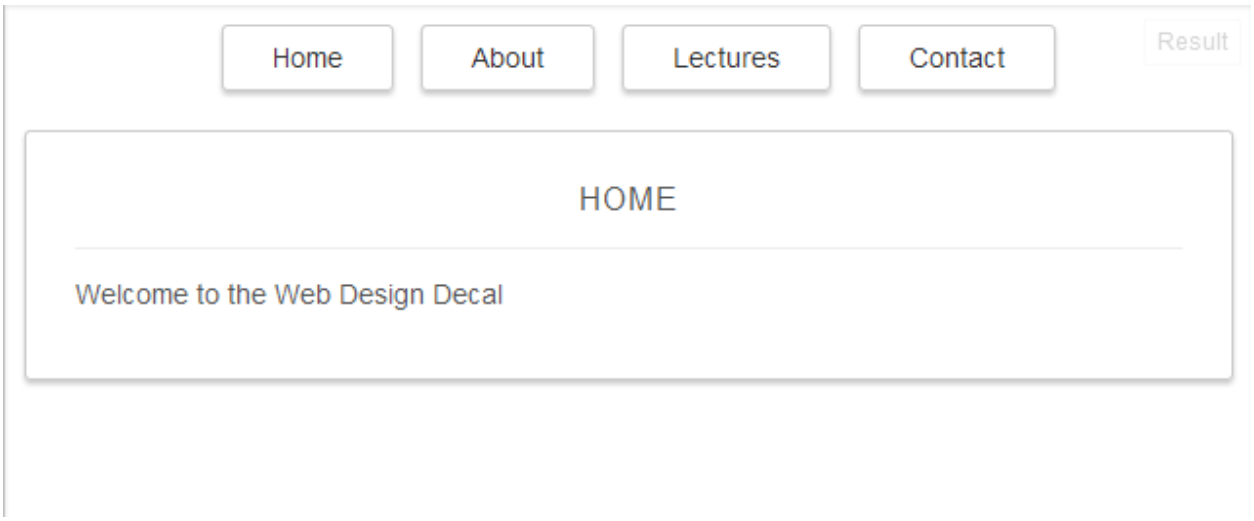
6. How will make a panel show?

We have the panel ID stored in our variable **panel**, and we can simply show it with:

JavaScript

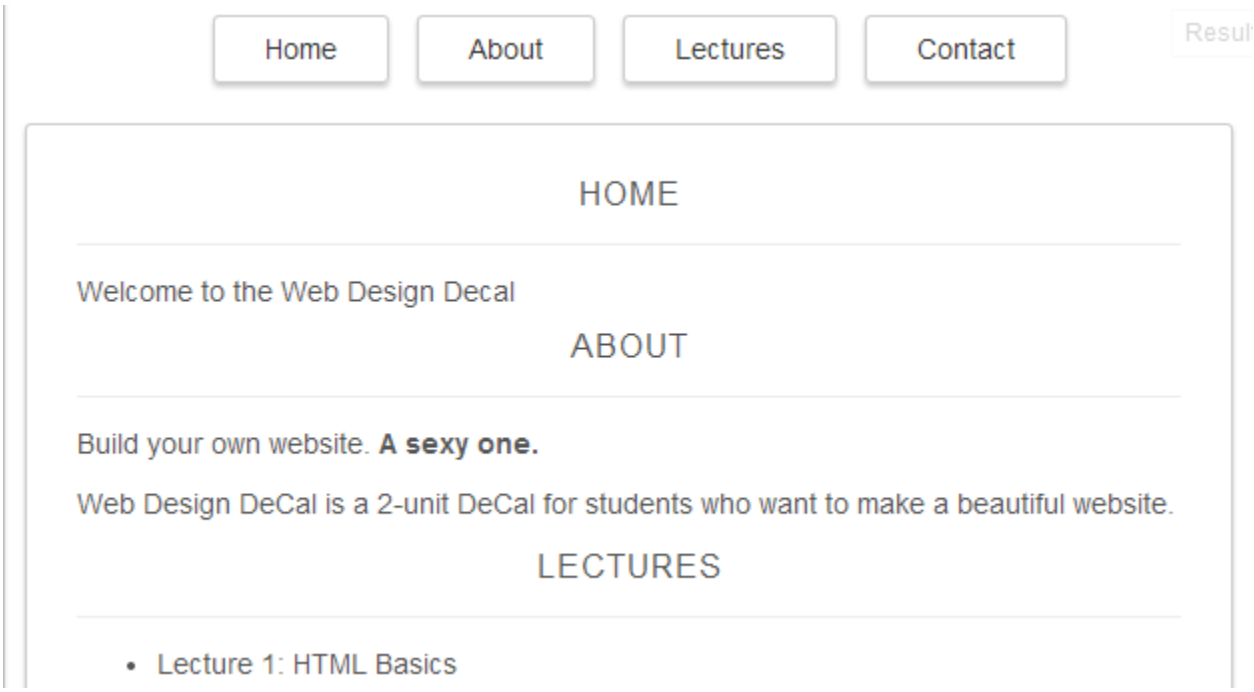
```
$(document).ready(function() {
    $('.button').click(function() {
        var panel = $(this).parent().attr('href');
        $(panel).fadeIn();
    });
});
```

Try clicking a button now and testing it out!



Great!

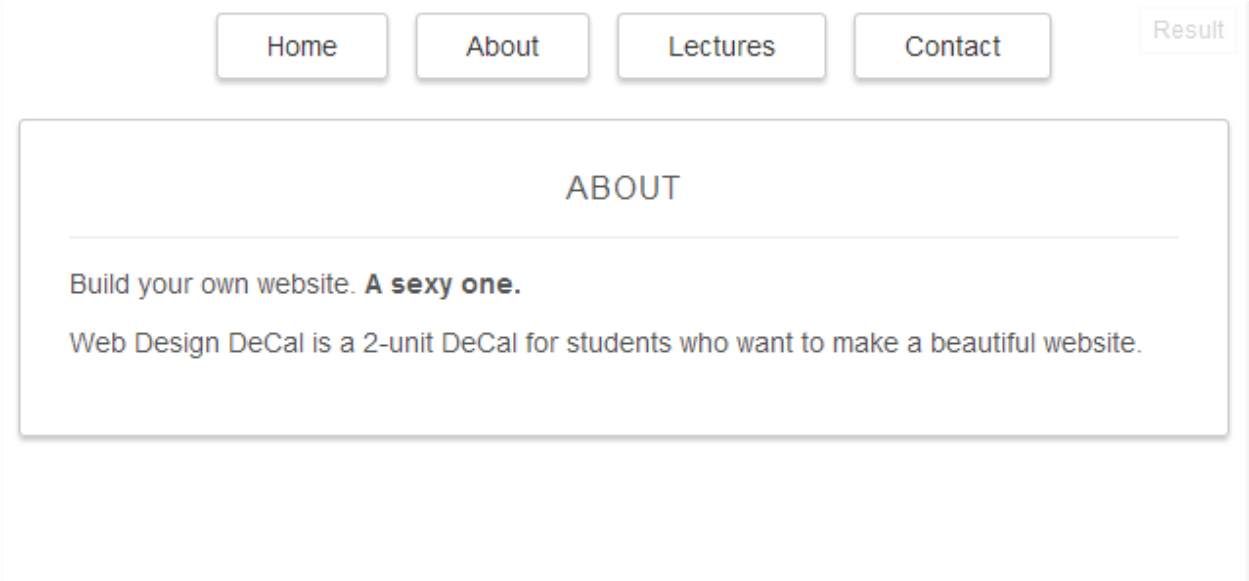
7. One problem! Clicking one button is fine, but click another and you will see a second panel fade in, but the first is still there!



8. To remedy this, make all elements with class **.panel** disappear before you fade in the current **panel**. Like this:

```
$(document).ready(function() {
    $('.button').click(function() {
        var panel = $(this).parent().attr('href');
        $('.panel').hide();
        $(panel).fadeIn();
    });
});
```

9. Try clicking the buttons again and you should notice that one button will correspond to only one panel showing up, and the others will be hidden. ☺



2 – Changing the Button CSS

1. In the CSS, find the style for **.button.active**

```
    -webkit-transition: 0.2s;
}
.button:hover, .button.active {
    background-color: #333;
    border: 1px solid #555;
    color: white;
}
```

This is the style for hovering over a button and a style for **.button** elements that also have the class **.active.** That is what **.button.active** means. When two classes or IDs or a class/ID are connected together with no space, it simply refers to the element that has those corresponding classes, ids, or class/ID. You can even do this with element names (a.active, img.scaled, etc.)

At the moment, none of **.button** elements have the class active. But that will change shortly.

```html
<div id="buttons">
    <a href="#home">
        <div class="button">Home</div>
    </a>
    <a href="#about">
        <div class="button">About</div>
    </a>
    <a href="#lectures">
        <div class="button">Lectures</div>
    </a>
    <a href="#contact">
        <div class="button">Contact</div>
    </a>
```

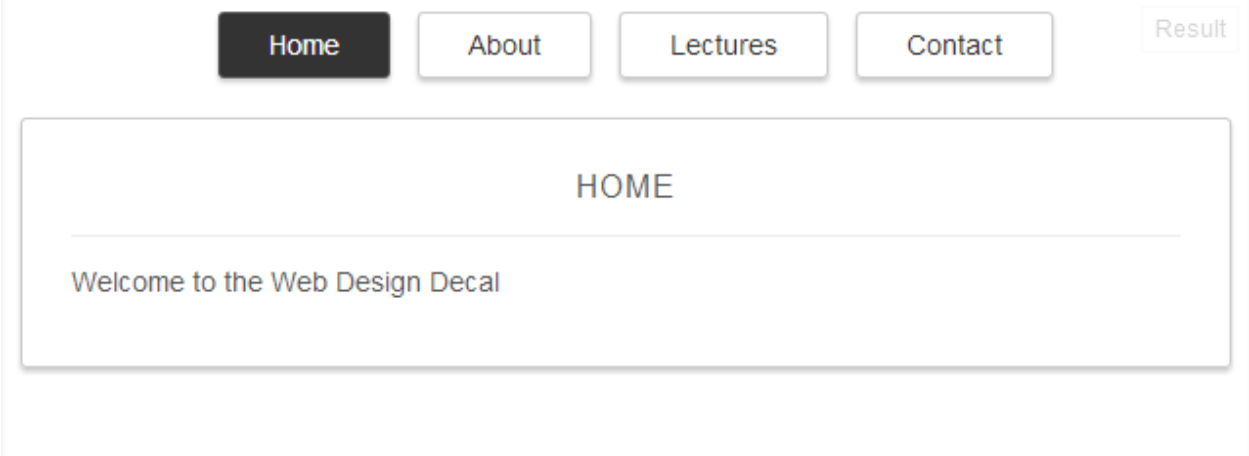
2. When you click on a button, we want to make it so that its CSS will change to the same one as **.button:hover**. Thus, when you click on a **.button**, we want to add the class **.active** to it, to signify that the button was clicked and we are showing the corresponding panel for that button.

   We do this like so (in the line above var panel):

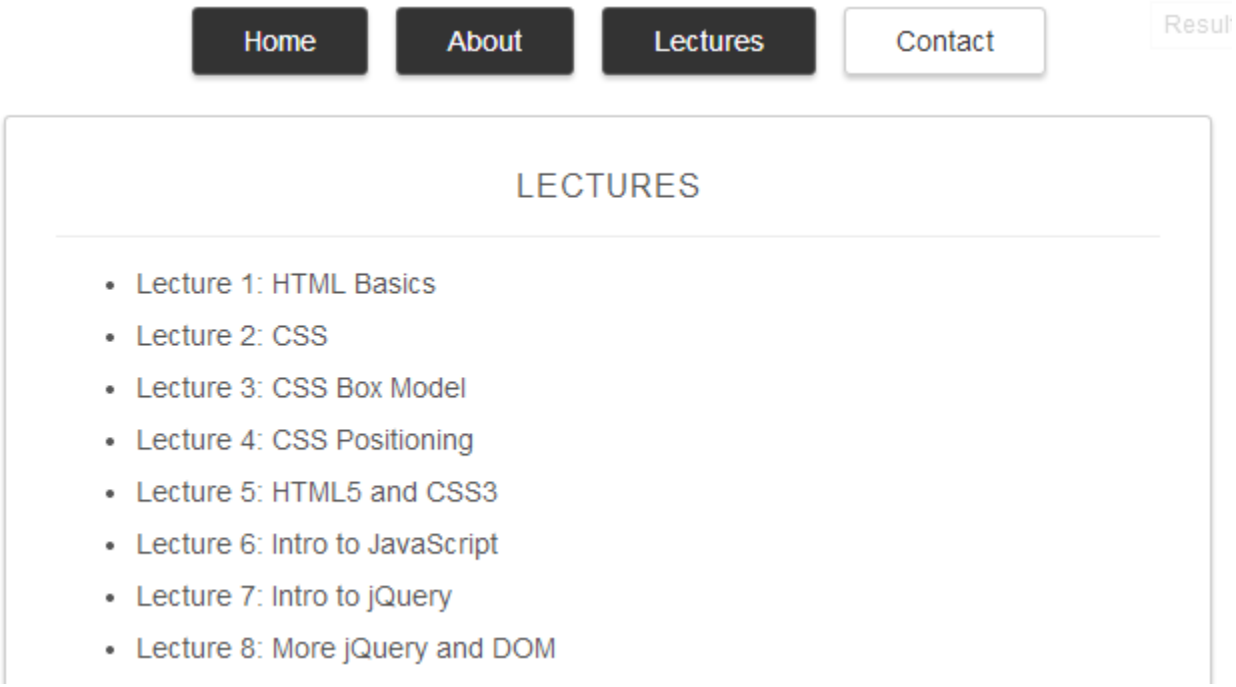
```javascript
$(document).ready(function() {
    $('.button').click(function() {
        $(this).addClass('active');
        var panel = $(this).parent().attr('href');
        $('.panel').hide();
        $(panel).fadeIn();
    });
});
```

   Again, **$(this)** is referring to just this **.button** element (the one we clicked) and not the other **.button** elements.

   Try clicking the buttons now!



3. Again, we have another issue, and that is when you click multiple buttons, the CSS changes but the other buttons never go back to their original style.

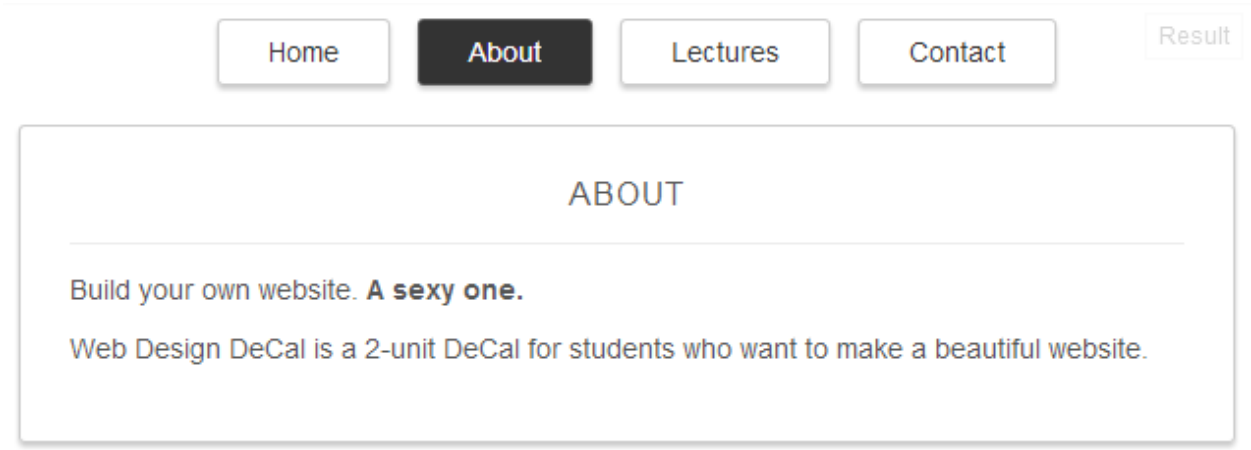


This is because we keep adding the class **.active** to buttons without ever removing them for the other buttons.

4. Before our line **$(this).addClass('active');** we have to select all elements with the **.button** class and remove **.active**, whether or not they have the class. This is literally just resetting all the elements with **.button** class to their original state (no second class **.active**).

```
$(document).ready(function() {
    $('.button').click(function() {
        $('.button').removeClass('active');
        $(this).addClass('active');
        var panel = $(this).parent().attr('href');
        $('.panel').hide();
        $(panel).fadeIn();
    });
});
```

5. Great! We have fixed our buttons!

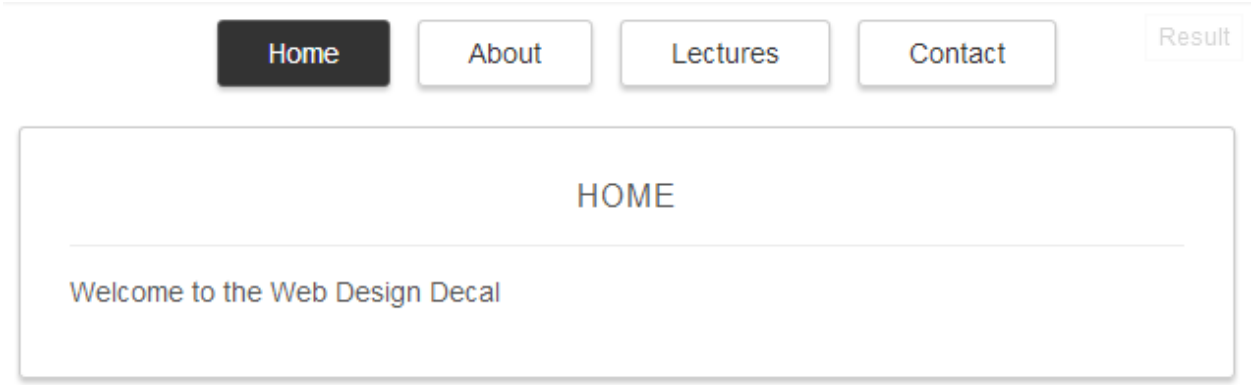


Just one last thing. Initially when we click run and load the page, no panels are selected because we haven't clicked on any button.

We can change this by adding a line right after our click event, so that we can click the **first .button element** on our page when it loads.

```
$(document).ready(function() {
    $('.button').click(function() {
        $('.button').removeClass('active');
        $(this).addClass('active');
        var panel = $(this).parent().attr('href');
        $('.panel').hide();
        $(panel).fadeIn();
    });
    $('.button:first').click();
});
```

Now if you run the page, you will notice that our **Home** button is selected, and the **Home** panel is displayed, thanks to this new line that we added.



Note that this line is executed immediately as opposed to the the **.click(function() { … })** above it. Because the above click event defines a **function** inside, it doesn't do anything when the page first runs. Instead, it acts as an event listener waiting for events to occur and executing everything in the function when the event happens.

```
$('.button').click(function() {
    $('.button').removeClass('active');
    $(this).addClass('active');
    var panel = $(this).parent().attr('href');
    $('.panel').hide();
    $(panel).fadeIn();
});
```

Compare this to the second click event that has no function, so it executes right away. If there is an event listener that was defined before it with a **function** inside, it will call that **function** (like what just happened).

```
$('.button:first').click();
```

Great job! You have just learned how to do panels, or essentially simulating tabbed content on a website. ☺ This is an extremely useful hands-on activity that you may want to reference in the future, as it will allow you to hide and unhide content on a page without navigating to other pages.

Congratulations! Let's review what we learned:

- Reviewed **.click()** events
- Utilized **$(this)**
- Accessed the DOM with **.parent()**
- Grabbed attribute values with **.attr()**
- Adding and Removing classes with **.addClass()** and **.removeClass()**
- Clicking the first element with **:first**
- Creating panels