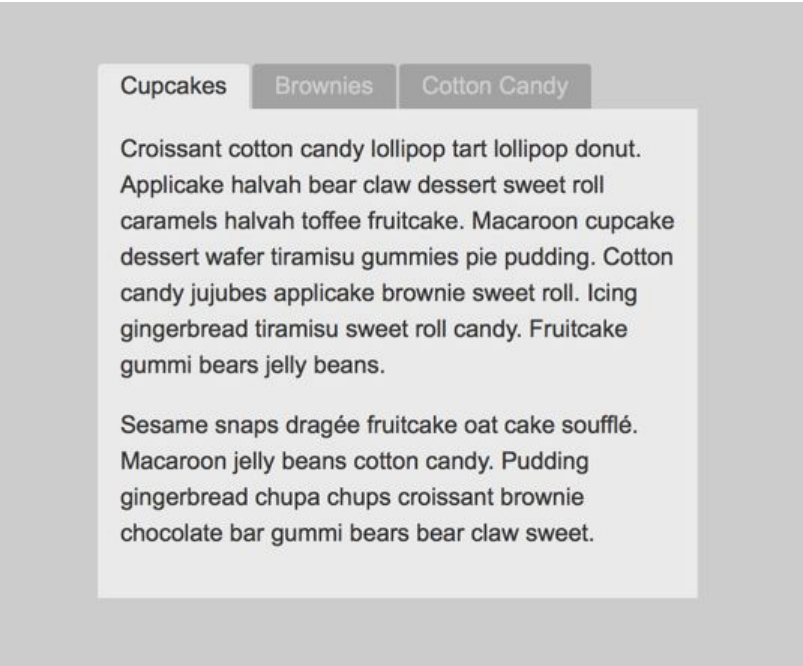
Tabbed Interface

Using jQuery

End Result



The end result for this project will be a nice simple tabbed interface that could be included in any website.

The Start File

```
<div id="tabs">
    <ul>
        <li><a href="#tabs-1">Cupcakes</a></li>
        <li><a href="#tabs-2">Brownies</a></li>
        <li><a href="#tabs-3">Cotton Candy</a></li>
    </ul>

    <div id="tabs-1">
        <p>Croissant cotton candy lollipop tart lollipo
fruitcake. Macaroon cupcake dessert wafer tiramisu gumm
gingerbread tiramisu sweet roll candy. Fruitcake gummi
        <p>Sesame snaps dragée fruitcake oat cake souff
brownie chocolate bar gummi bears bear claw sweet.</p>
    </div>
```

The start file has the markup we need on it. This includes a div with the id set to “tabs”.

Then there is an unordered list with list items and links. These will make the actual tabs.

Then there is a div for each tab. Notice the href matches the ID of each div with and ID of tab. You can add more tabs or change their href, just make sure the match the ID of the content that appears in the tab. You can put in whatever content you like. I just have some text in each tab.

CSS

```
#tabs {
      width:400px;
      margin:auto;
}

#tabs > ul {
      list-style-type:none;
      display: flex;
}
```

There are a few basic styles already at the top of the page. Let's add to these to get the basic setup for the site in place. For this demo, the entire tab interface will be 400px wide and centered on the screen.

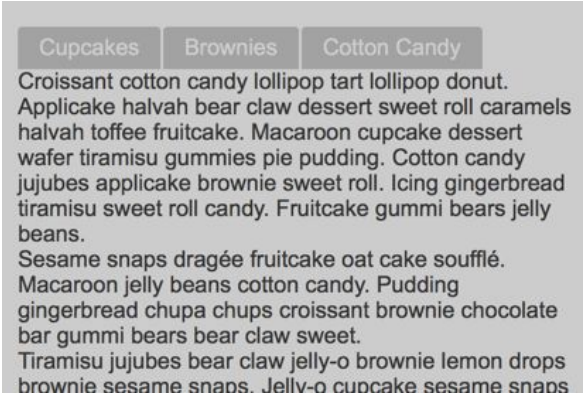
The bullets are removed from the list and `display: flex;` puts the list items all in a line.

Styling the Anchor Tags

```
#tabs > ul > li > a {
      display: block;
      height: 30px;
      line-height: 30px;
      margin-right: 2px;
      background: #A2A2A2;
      color: #CECECE;
      text-decoration: none;
      padding: 0 15px;
      border-radius: 3px 3px 0 0;
}
```

Most of the styling happens on the actual anchor tags.

It should look like this, but with more text...



Final CSS Needed

Initially, all of the content areas for the tabs are hidden. Set them to `display: none`.

Then you want the first div to display. That will be your first tab and will come up displaying by default when the page loads.

The anchor tag inside the first list item should have the styling of the current active tab.

Finally, add some styling for the paragraphs, so that they look a little better.

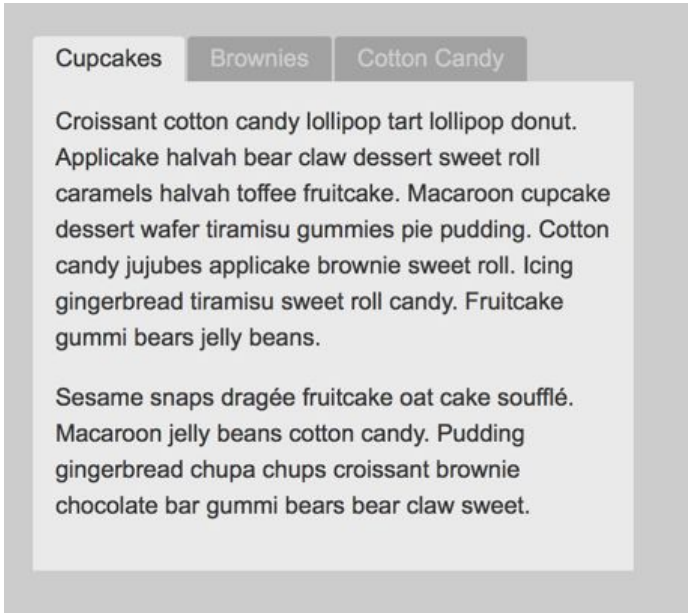
```
#tabs > div {
      display: none;
      padding: 15px;
      background: #EAEAEA;
}

#tabs > div:first-of-type {
      display: block;
      background: #EAEAEA;
}

#tabs > ul > li:first-of-type a {
      color: #333;
      background: #EAEAEA;
}

p {
      line-height: 1.5em;
      margin-bottom: 1em;
}
```

Tabs with Styling



Now it looks like our final product, but it does not function yet. You will do that with jQuery.

Starting the Script

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

<script>
      $("#tabs > ul > li > a").click();
</script>
```

Start the jQuery by adding a click handler to the anchor tags. Use the child combinator selector angle brackets because it is possible that someone might put an unordered list with list items and anchor tags in the content of the tab, and you don't want clicking those to trigger the change of the tab.

```
$("#tabs > ul > li > a").click( function(){/* code here */} );
```

Add the anonymous function that will run when one of these links is clicked.

Changing the Tab Colors

```
$("#tabs > ul > li > a").click( function(){
      $("#tabs > ul > li > a").css( { "background" : "#A2A2A2", "color" : "#CECECE" } );

      $(this).css( { "background" : "#EAEAEA", "color" : "#333" } );
} );
```

The first thing to do in the function, when a tab is clicked, is change the styling of all the tabs to the inactive tab styling. Then change the one we clicked … $(this) … to the styling of the active tab.

Getting the Tab that was Clicked

```
$("#tabs > ul > li > a").click( function(){
      $("#tabs > ul > li > a").css( { "background" : "#A2A2A2", "color" : "#CECECE" } );

      $(this).css( { "background" : "#EAEAEA", "color" : "#333" } );

      const thisTab = $(this).attr("href");

      //alert(thisTab);
} );
```

Then get the HREF attribute for the link that was clicked and put it in a variable called “thisTab” for easy access. You can add the alert and see what this gives you.

Fade Out the Visible Tab

```
$("#tabs > ul > li > a").click( function(){
      $("#tabs > ul > li > a").css( { "background" : "#A2A2A2", "color" : "#CECECE" } );

      $(this).css( { "background" : "#EAEAEA", "color" : "#333" } );

      const thisTab = $(this).attr("href");

      $("#tabs > div:visible").fadeOut(200);
} );
```

Then take the div that is visible, in other words, the tab that is currently active and showing, and fade it out.

Fade in the New Tab

```
$("#tabs > div:visible").fadeOut(200, function(){} );
```

Once that div has completed the fadeOut animation, use the callback function to fadeIn the div that matches the href for the link we clicked.

```
$("#tabs > div:visible").fadeOut(200, function(){ $(thisTab).fadeIn(200); } );
```

The Whole Script

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

<script>

$("#tabs > ul > li > a").click( function(){

      $("#tabs > ul > li > a").css( { "background" : "#A2A2A2", "color" : "#CECECE" } );

      $(this).css( { "background" : "#EAEAEA", "color" : "#333" } );

      const thisTab = $(this).attr("href");

      $("#tabs > div:visible").fadeOut(200, function(){
            $(thisTab).fadeIn(200);
      } );
} );

</script>
```

Summary

This is a great use of jQuery. The use of the animation library is good here because it is not something that is going to be very browser intensive.

The last things to do with the script are to take care of the best practices and do the following:

1. Put it in a separate linked file
2. Add the IIFE closure
3. Add the “use strict” directive
4. Link it in the head of the page with the defer property set.