2.4.1.7: Switching displays in the gallery

In the last two videos, I showed you how to represent the content of a photos albums app in terms of erasing objects. And then I showed you how to represent the structure in terms of templates. All we need to do now is go from one to the other and to do that we need a bit of JavaScript code.

> I want to quickly remind you of what we're looking for here. We've got a little app with albums, you click on albums, it opens up all the photos in that album. When you click on the photo, you display it big. And if you click on the slideshow tab it takes the current album and gives you a slide show. If you click on photos of the current album it shows you there's a list.

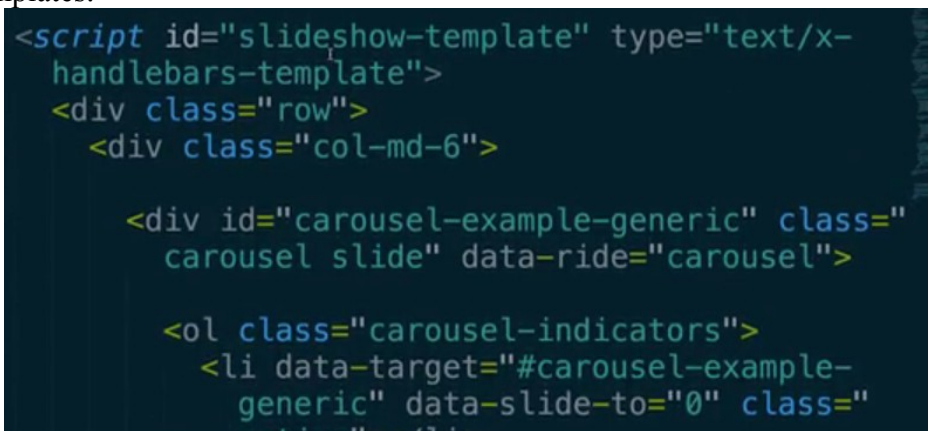
I want to flag up that I'm talking about the current album that's quite important.

> When you click on an album it remembers which is the last one you looked at, so that if you want to start a slide show it shows you the right album. And similarly with the current photo as well, you might want to remember that as well

I'm gonna show you some code which remembers what you've clicked on so that you can access it later. Let's very quickly recap the contents and structure.

```
var gallery = {
    albums : [
        {
            name : "Travels",
            thumbnail : "images/img_1.jpg",
            photos : [
                {
                    src : "images/img_1.jpg",
                    title : "grafitti",
                    description : "some derelict
                },
                {
                    src : "images/img_6.jpg",
                    title : "fountain",
```

The content consists of this big JavaScript object called gallery, which contains a number of albums. Each of which has name, thumbnail, and some photos. And we're displaying these via a number of templates.



```
<script id="slideshow-template" type="text/x-
  handlebars-template">
  <div class="row">
    <div class="col-md-6">

      <div id="carousel-example-generic" class="
        carousel slide" data-ride="carousel">

        <ol class="carousel-indicators">
          <li data-target="#carousel-example-
            generic" data-slide-to="0" class="
```

- And I had four templates in here. One for albums, one for photos and albums, one for individual photos and this last one for a slide show.

```
<ul class="nav nav-tabs">
      <li role="" class="active"><a href="
      #" id="albums-tab" >Albums</a></li>
      <li role=""><a href="#" id="photos-
      tab" >Photos</a></li>
      <li role=""><a href="#" id="
      slideshow-tab" >Slideshow</a></li>
    </ul>
<br/><br/>
```

Quickly once go to the top of this file and show you that we also have these navigation tabs up here. These can be used to bring up the slide show or the photos or return to the album view. That gives us the html structure that we're going to work with.

```
var albums_template, photos_template, photo_templ
var current_album = gallery.albums[0];
var current_photo = current_album.photos[0];

function showTemplate(template, data){
    var html    = template(data);
    $('#content').html(html);
}

$(document).ready(function(){

    var source   = $("#albums-template").html();
    albums_template = Handlebars.compile(source);

    source   = $("#photos-template").html();
```

So I've now put all the main JavaScript code inside this file called gallery.js which now controls how we map from the data onto the structure via the templates. What this gallery.js needs to do is know what we should be displaying at a given time. So user clicks on something, on a button, on an image. And it knows that it has to bring up, select a bunch of data and show the appropriate template for that data. And that is the basics of what we're gonna do with this code. Let me go through it from the top.

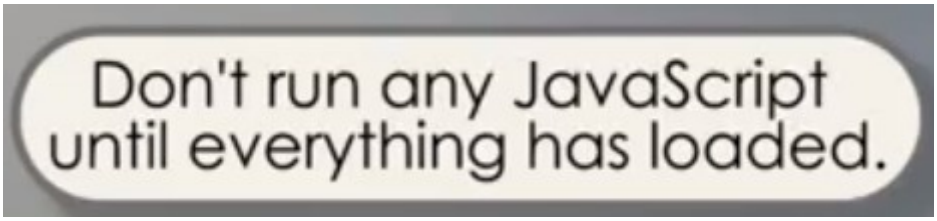
The first thing you can see are a bunch of variables to abscend the templates. So we're going to compile those templates and store them so we can use them later. We're going to compile them all at the beginning. We've also got variables for the current album and the current photo. This is what I was telling you just now that we want to remember which album was last clicked on so when we display a slide show, displaying photos, we're displaying the right one. And that's as simple as storing it in a variable.
The last thing I've got up here is a little function called **showTemplate()**. Now a lot of this code is basically going to take a template, apply some data to it and put some content or html. And so I'm gonna encapsulate all of that into a single function so I can use it again and again. This just makes my code a lot easier. And this is particularly possible because all of the examples are put in the same place, content or html **$('#content').html(html);**

So, doing this is particularly easy because in all my examples I'm just gonna put my template content into this div called content.html. Let me just quickly show you that that's true. Whenever I click on anything. It always appears in the same place in this main part of my photo album and it's never affecting the navigation parts at the top of the title.

So with these basics in place, I can have the main code in my JavaScript file. Now, I'm going to show you something a little bit new here, which is this **$(document).ready(function(){**.

> Up to now, we've been always just putting some JavaScript at the end of the html. The reason we put it at the end is we want all the html file to load before we start running the JavaScript.

But, actually, html can take a while to load and process and sometimes. And it might not be quite ready for the JavaScript to act on it, to run on it, so it might be trying to access elements that have not loaded yet.



> In a small file that's not gonna be an issue, in a big project it's gonna be increasingly an issue. So we really want to make sure we're only running JavaScript code when all the html is loaded.

And this is how we do it. It's a special j-query function called Ready and it's called on document **$(document).ready(function(){ -** document here represents your entire html document and your entire web page. And it just calls this anonymous function. This anonymous function is just going to do all the stuff we need to set up for our page to work. We're going to start by loading and compiling all of our templates. So we've got all of our templates stored in variables so that we can use them later.

Then further down, we've got these click functions for the albums tab, photos tab, and slide show tab. So the three main tabs of our interface.

- $("#albums-tab").click(function ()
- $("#photos-tab").click(function ()
- $("#slideshow-tab").click(function ()

We can see the tabs here, albums, photos, slideshow. And each of those triggers some templates to be loaded.

Let's look at the albums tab as an example.

```
("#albums-tab").click(function () {
    showTemplate(albums_template, gallery);
    $(".nav-tabs .active").removeClass("active");
    $("#albums-tab").addClass("active");

    $(".album-thumbnail").click(function (){
        console.log("clicked on an album");
        var index = $(this).data("id");
        current_album = gallery.albums[index];
        showTemplate(photos_template, current_albu
        $(".photo-thumbnail").click(function (){
            console.log("clicked on a photo");
            var index = $(this).data("id");
            current_photo = current_album.photos[i
            showTemplate(photo_template, current_p
```

The first thing we're gonna do is show the templates. In this case the albums template and we're using the entire gallery object so if we go up here we're gonna use all of this object which contains all the albums because the albums view is an overview of everything in the gallery There's a little bit of code here which is manipulating the tab itself. If you look when I click on a tab, it shows the current tab is this kind of grey, active thing, and the others are sort of white in the background. So you can see which one I just clicked on. So it's quite important to display those properly, so that we know where we are and how to navigate. This is a little bit of JavaScript which does that, we're using the active class to be able to say which is the active tab. Which one we should be displaying and what we need to do is remove the active class from all of the tabs. That are currently active, so that basically means removing it from the currently active tab, and then add the active tab to the tab we've just selected, ie homes tab. Once we've loaded all our albums up we want to make them into active. So, we are grabbing all the elements with class album thumbnail.

```
{{#each albums}}
<div class="col-xs-12 col-md-3">
    <div class="album-thumbnail"  data-id="
      {{@index}}">
      <img class="crop-img" src="
        {{thumbnail}}" alt=""/>
```

If we go to our template we can see the album thumbnail is the class we've applied to the whole that holds the album view including the image and it's name. We're grabbing all of those and giving them a click function. And what we do is, this code that I showed you earlier, we're grabbing the data ID out of that. Using that to select the correct album from the array and then showing that album by the template we're showing the current album by the photos template. What we're doing here is we're, rather than just simply displaying it, we're not, we're also saving the current album into the current album variable. As I said before this allows us to remember which album we just clicked on. And so if I click on the photos tab or the slide show tab it will display the correct one. And finally inside that we have yet some more code which tells us what to do when we click on a photo thumbnail. So again, it selects the currents, the photo by the data id, saves that and displays it. Pretty much the same code here as here except in this case it's working on photos, not on albums.

The other tabs are pretty similar, in fact, the photos tab. Does exactly the same thing, almost exactly the same thing as clicking on an album except it doesn't say the current album, it just displays it. In fact, it might have been a little bit better to, instead of repeating the code twice, when I select an album, I could have done a virtual click by calling jquery.click. On the photos tab so I just saved a current album and done the virtual click on photos tab and that would have had the same effect but I've have had to only written the code once. That would have been slightly better style but I've shown it to you twice here to show sort of show you at each stage exactly what is happening.

The last tab is the slide show tab and that's really very simple. All it does is show the slideshow template. Then it does this stuff with the nav buttons to make them active, one active and the rest inactive. That gives us pretty much all the functionality of our app.

The only thing we need to do is display the first page. **$("#albums-tab").click();** because we don't want people to see a blank page when they start. So what we do is use, do this, which is using J Crew to select the albums tab and then calling click and this does like a virtual click on it. So what do I mean by virtual click, it means that I'm calling all the code that would normally happen when you would click on the albums tab, so I'm just displaying the albums. So when you first load the page. The first thing you see is the albums, which is exactly right.

And here it is for one last time. Albums. Click on an album. You see the photos. Click on slide show. You see it's a different way. Go back. Select a different album. You get a different slide show. Select photos. You can see all the photos there. And you can click on any one of them and display that photo. There's obviously a lot more complicated stuff that we could do with the photos app. I haven't included the search code I've just given you in this one, but we could do that very easily.

And there are lots of sort of little bells and whistles we can do, but this is a really good starting example. You can now see how a fairly large web page would work and how you had the different functionality in there. It's a very nice example for you to see and you can go on and use it, but it's also, I wanted for it to be a starting point for you to go and start building your own web pages to do different functionality that uses all of the things I've shown you. So HTML, CSS JavaScript, Bootstrap, templates, arrays, objects, data, and various forms of interaction via JavaScript. And with these basic tool sets and the kind of outline I've just given you, I think you can do some pretty exciting and pretty impressive projects.