

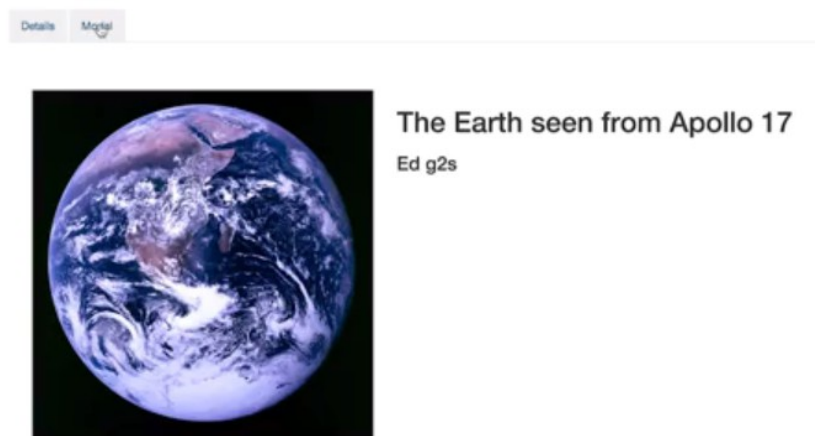
2.3.1.6: Displaying the same data with different templates, Use a Bootstrap modal

Hello again. We've started to see the real power of templates to be able to set their traits, contents, and structure. And next week, we're gonna go through a whole range of techniques and make some powerful websites. But to finish off this week, I want to do one more thing with you which is flip over what we did in the last video. So the last video,

- we looked at how we can use a single template and put two different bits of data on it, that's the main thing you're gonna do with templates. But actually, you can kind of do the opposite and use templates to display a single set of data in two different ways.

So if we look at this new version of the website, I've got two little buttons here, at the top. The first one displays an image with all these details. You can see the image, you can see the message dates, the title or the author. But, I can also create a different view. The mass data view's useful if you wanna know everything about it or you want to extract information, it's not the nicest looking way of viewing an image.

My photo albums



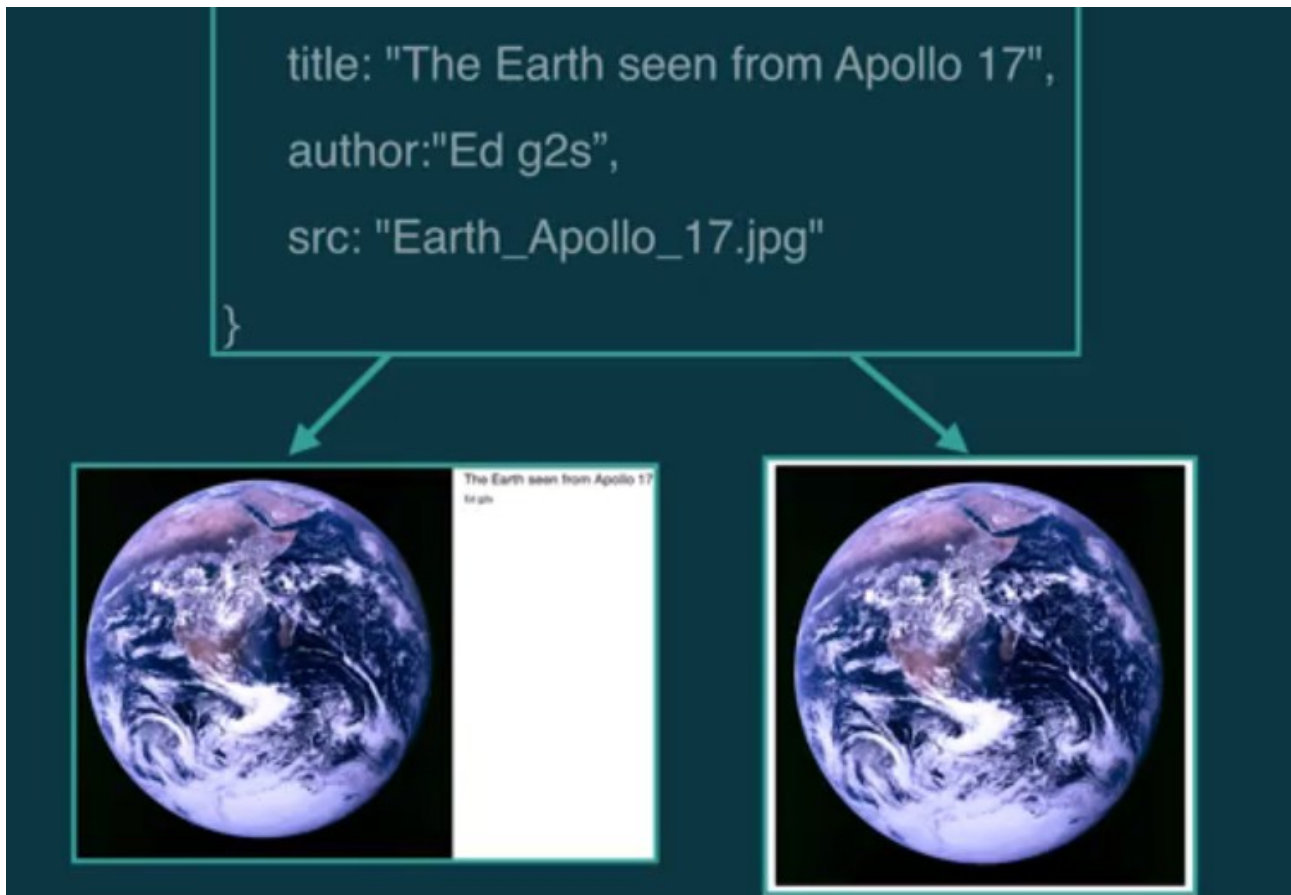
So let's have another way, which is creating this nice pop up view which appears in front of the page, it's got a nice border, it looks much nicer.



So I'm gonna take you through how you display both of these different images from the same set of data, from the same data object.

So what we have is an object and it's exactly the same object we've been using all through this week. And what we wanna do is take that object, run it through a template and we get a web page, that's what we've done in the last couple of sessions.

But again, there's nothing to stop us taking that same object, running it through a separate template and getting another kind of HTML that looks very different and I'll tell you how to do that.



We'll, it's essentially using the same techniques we used in the previous videos, so there's nothing too difficult. But before we launch into those techniques, I wanna quickly show you how I created that nice sort of modal pop-up that I'm gonna use the word modal. That's a term we use for a piece of an user interface, in this case, face to fact HTML, it appears in top of and covers up the original interface. So that you're suddenly interrupted for your main interactions with a webpage by this and you can, you focusing on that for the moment and then you can make it disappear. It's used a lot for dialog boxes, it pops up and you have to click. You sort of read what it says and you click on it to go away. But it's also quite nice, in this kinda context, for displaying an image in a way that you focus really on the look of the image, away from all the details and metadata.

So, let's have a look at modals.

```

<div id="imageModal" class="modal fade" role="dialog">
  <div class="modal-dialog" style="width:800">

    <div class="modal-content">

      <div class="modal-body">
        
      </div>

    </div>

  </div>
</div>

```

So this is the code that we're using for modal, we're using the Bootstrap modal system. So again, as usual, in the last few videos where we're having to include Bootstrap and this is quite complex code. So there's a lot of divs to create this Bootstrap modal,

- I won't go into the full details, but the reason there's so many divs is there's quite a lot we can customize in how the modal works, the framing, whether there's an on off button.

Our modal was really very simple and all of the stuff we're doing is in this modal body, div, which just contains the main content of the body and all it really is is an image tag. So we've got an image tag, it's got a width of a 100% and the source is coming in from our template, so it's a template expression. So this is all we're really looking for, you can, if you want to find out more about modals, I'd recommend you look at the Bootstrap website. There's lots of documentation on there to help you get working on them. So this is the HTML for the modal, next up, I want to show you how actually use it.

```

/* we can only show the modal once the
template has been instantiated
because it does not exist before */
$("#imageModal").modal('show');

// create a call back for when the model is
// hidden so we can re-display the detail template
$('#imageModal').on('hidden.bs.modal', function () {
  $("#detailsbtn").click();
});

```

Modals are a bit different from standard HTML in that they appear and disappear. So we need a little bit of JavaScript to get that working. And as usual, we're turning to jQuery and we're using this jQuery-like call `$("#imageModal").modal('show');`

In fact, it's an extension of jQuery that Bootstrap provides and it's this modal function. So we're grabbing using jQuery to grab the modal diff. Recording this modal function and which is at this point is showing, cuz were passing in show as a parameter.

And we're seeing the code, we're also using the opposite way, passing in hide as a parameter to hide the modal. So that's really all we all need to do to make it pop up.

Show the modal

```
$("#imageModal").modal('show');
```

bind a function to an event

```
$('#imageModal').on('hidden.bs.modal', function () {  
    $("#detailsbtn").click();  
})
```

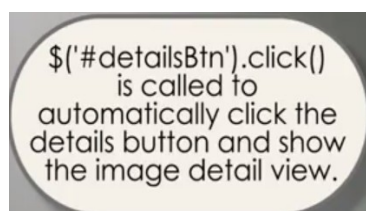
The only other thing I'm gonna show you is this other bit of code which does something when we're hiding the modal. So you can click away from the modal and it will disappear but we want something instant shown once we made it disappear. So we gonna bind a function to this event. So, we're using the `jQuery.on` function. The `jQuery.on` function is a function that binds some functionality to the main event happens in your webpage. We've kind of seen this before. So the click function finds function to the moment you click a button on an element. And `on` is the more general version of that.

- So the click is just a short-hand for `onclick` and, but, instead of `onclick`, we can have `on` for a wide, massive range of things that can happen on a web page and mouse-overs, clicks, keyboards, lots of things like that. But also, the custom events that are specific to a particular type of element.

And this time, we are looking at a custom event which is specific to modals. So we're using `jQuery.on` to bind some function onto this event. The first argument is the event that we're actually responding to, in this case, is the name of the event,

- in this case is `hidden.bs`, standing for Bootstrap, `.modal`. This is a very specific event that only applies to Bootstrap modals and it's the event that happens when the modal has been hidden, when it disappears when we've clicked away.

And what we're doing is we're binding a function to that and the function is very simple, all it does is click on the details button. So what that means is that we're just trying to, once we close the modal view we're bringing up the details view. We've already got some functionality to do that. We've got the details button, when we press that it brings up the details view. Instead of re-implementing that, we just trigger an artificial click, so it's kind of virtually clicking on that button.



`$('#detailsBtn').click()`
is called to
automatically click the
details button and show
the image detail view.

So let's have a look at the actual code. As before, we've got our nav toolbar with our tabs.

```
<!-- this is a simple nav bar for selecting between display methods -->
<ul class="nav nav-tabs">
  <li role="" id="detailsbtn"><a href="#">Details</a></li>
  <li role="" id="modalbtn"><a href="#">Modal</a></li>
</ul>
```

This time, it's details and modal, but otherwise the names are changed but pretty much the same. We've got our content div where you gonna put all our content that starts off on empty, exactly, it's the same as before.

```
<!-- the content of the web page starts off empty
      because we will fill it later from the template -->
<div id="content" class="container-fluid" role="main">
</div>
```

And here is our details template, so it's one of our two templates and this one's exactly the same as what we've seen in the last video.

```
<script id="detail-template" type="text/x-handlebars-template">
  <br>
  <div class="row-fluid">
    <div class="col-sm-5">
      
    </div>
    <div class="col-sm-7">
      <h1>{{title}}</h1>
      <h3 class="author">
        {{author}}
      </h3>
    </div>
  </div>
</div>
</div>
</script>
```

The one that's different is the modal templates and this is just the code I showed you on the slide a few minutes ago, bringing up this modal dialogue.

```
<script id="modal-template" type="text/x-handlebars-template">
  <div id="imageModal" class="modal fade" role="dialog">
    <div class="modal-dialog" style="width:800">

      <div class="modal-content">

        <div class="modal-body">
          
        </div>

      </div>

    </div>
  </div>
</script>
```

- Important thing to remember is that modals are different, so in the details template, just adding that to the HTML will create it and you'll be able to see it straight away. The modal, when you add in a modal to your HTML, is not immediately visible. You have to show it with Javascript

```

<script type="text/javascript">

    // This time we have two templates we compile both of them and store the
    // results in separate variables
    var source    = $("#detail-template").html();

    var detail_template = Handlebars.compile(source);

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

    var modal_template = Handlebars.compile(source);

    // this is the data object we will be using
    var data = {
        src:
        "https://upload.wikimedia.org/wikipedia/commons/archive/9/97/20101017074210%21Th
e_Earth_seen_from_Apollo_17.jpg",
        title:"The Earth seen from Apollo 17",
        author:"Ed g2s"
    };

    var imageData = {
        title: "image1",
        author: "Tuxtso",
        src:
        "https://upload.wikimedia.org/wikipedia/commons/archive/9/97/20101017074210%21Th
e_Earth_seen_from_Apollo_17.jpg",
    };
    //print out the variables
    console.log(imageData.author + " " + imageData.title + " " +
imageData.src);

    // this instantiates the detail template
    $("#detailsbtn").click(function () {

        // hide the modal if it is showing
        $("#imageModal").modal('hide');

        //use the detail template to generate html
        // and put it in the DOM
        var html    = detail_template(data);
        $('#content').html(html);
    });

    $("#modalbtn").click(function () {

        //use the modal template to generate html and put it in the DOM
        var html    = modal_template(data);
        $('#content').html(html);

        /* we can only show the modal once the template has been instantiated
        because it does not exist before */
        $("#imageModal").modal('show');

        // create a call back for when the modal is
        // hidden so we can re-display the detail template
        $('#imageModal').on('hidden.bs.modal', function () {
            $("#detailsbtn").click();
        });

    });

</script>

```

And then we just do what we've done in the previous video, we just bring up the template, create some HTML from it and add it to our content div. The Modal onclick is a little bit more complex. We bring up the HTML, we use the template to create an HTML added for content, as usual. And then at that point, we have to show you the modal, because it doesn't automatically show, as I said.

- Now it is really important that you only call this function after you've added your HTML. Because until you've added the HTML, the image modal elements do not exist. This is a really important thing to know when we're working with templates, is that we've got stuff on our webpage that doesn't exist straight away because we haven't written HTML.

It's only created later as we execute the JavaScript. So, it's important to bear in mind, as you're writing your JavaScript, that you can only do stuff which accesses the elements in your template after you've instantiated the templates and created the HTML. So you have to just be a little bit careful. You can easily get a gotcha where you're sort of calling some HTML element but the template hasn't been instantiated and so it does nothing and you're you spend the time scratching your head and sort of working out what to do.

Okay, so the last thing is this function which is doing something when the modal is hidden. As I've shown you on the slides, it just waits for that event and then it calls a function that will, clicks details button and show the details. So that's the code, let's have another look back at the actual webpage. Okay, so this is the webpage again and as I said to you before, I can click on details, I bring up one view of the data. Click on modal, I bring up another view of the data.

So now we've seen that we can, as well as using templates, one template to show many different images using the same format. We can reformat how we show one image in different contexts. So we're showing that the power of templates and the separation of structure and content, that really we could be really fluid with what we do, appearance wise, and still keep a common core of content and that is a really exciting possibility.