**DwBP - 8 - Using the iframe Player Implementation**

In the last two tasks, we've written code to meet a couple of use cases where we use the advanced, also called in-page embed, player implementation where the video tag and the associated script are together on the HTML page. In this lesson, we're going to use the Standard, or iframe, player implementation. We're going to see what it looks like and discover the advantages and disadvantages of using the iframe.

Our use case this time is very similar to something we've done earlier with the Advanced code, and that is we're going to write code to change the video that's playing in the player based upon user interaction based on a button click.

Let’s look at the solution so we know exactly what we are trying to accomplish in this task. The change video button will work whether the first video is started or not. In this case the videos are NOT autoplay. The functionality is the same as task 2, but this time using a different player implementation.

Let's go into the Media module and see how we can get the Standard/ iframe player implementation code. In the media module pick a video to publish by clicking on it. Now click Publish and Embed and Web Player. Pick a player to use, then scroll down to the code section.

The Standard code is shown by default. Change any settings, and when you click on the code it is automatically copied to the clipboard. That’s what is involved getting the Standard code.

In task 4 solution we see the Standard player code and a button.

Let’s take a close look at the script block. It is important to now that really this is a JavaScript exercise, not really a Brightcove Player exercise. But it is good to understand that.

The high level solution to the problem is we are going to programmatically with the query string that contains the video ID that is in the player. When we do this, the browser will automatically reload the player with the new video in the player.

The code does this:  
- Gets a handle on the iframe element  
- Creates a variable that contains the new query parameter, which contains the new video ID  
- Extracts the value of the src attribute from the iframe and stores it in a variable named theSrc  
-

First thing I'm going to do is go through and get a handle, a reference to the iframe tag. And e're going to use get elements by tag name, and that actually returns an array. And that's why this square bracket zero is here. Even though there's only one, we still need to say, hey, we want the zero-th one.

Here's a string called new video that simply is the new query stream. It's the video ID property with a new video assign. We grabbed from the iframe tag, the SRC. So that's the whole string. We remove the query string using the substring JavaScript method, and we say go find the question mark and remove everything after that. And then we simply create another variable where we add the string without the query string and the new query string together.

That give me a little complex. So what I've done is in the console here I'm going to display the SRC without the video, and also the new video so we can see that. You can actually turn on the console. I want make this a larger, come down here, bring this up. And I need to click the change video button, and there we see our two variables, SRC without the video. And just as I said, there's the URL but without the query string. Here's the new query string, and then we add those two together and put it back into the iframe tag assigning at the source.

So let's see it in action here. I'm going to close the console. Refresh the page since I clicked the button. I do not have play going by default. If you wanted both of these to autoplay, you could simply set autoplay in studio. So I'm going to start this one playing. I'm going to click change the video. There's the new video, and I can click play. Again, if you wanted them both to play automatically, simply go into studio and for this player set start video automatically to true.

So there's the code that takes care of the use case we have in this section, this task we've accomplished, and that is using the iframe implementation and as well, changing the video based upon user interaction. Let's take a couple minutes now that we're somewhat familiar with the iframe implementation to talk about some good and bad things about the iframe and a few other details, and also some review.

First of all, the advantages of iframe player. Number one, there will be no collisions with existing JavaScript and/or CSS. So it's possible if you haven't handled your JavaScript in an elegant manner, you could have conflict simply by-- perhaps you have the same name for different functionality in two different places and you bring them together and now change video is defined in two places. That cannot happen when you're using an iframe player. It's nearly automatically responsive. You just have to go in and make sure that your width and height are set to percentages.

And then another nice thing about the iframe implementations is that it travels nicely to other social media apps. So if you want to share a video like in Facebook or on Twitter, it will use the iframe implementation to do that.

There are some times when you have to be careful about using the iframe implementation. One of those is code in the containing page needs to listen for or act on player events. So for instance, before we had some code in the player and we wanted something to happen, like display some text in the HTML page, based upon the load start event. That you can't do with an iframe implementation.

Also if you want to pull styles in and affect your player based on styles from the containing page, you won't be able to do that. And also, there may be times where it simply won't fit your application logic. For instance, let's say at the end of a video or at a call to action you want to change the page you're on. Well if you have that logic in the iframe, the page will change, but all that will change is in the iframe. And so you'll be in the new page just in the iframe, and all the surrounding HTML page will be the same. So there's good things and bad things about using the iframe player.

Here's a quick review of what we did. You see we have this query string that is part of the source and we removed that and replace it. Here's a high level list of what we did. And here's a resulting code based upon that list, and we can see that the substring JavaScript method is used to extract the character.

I probably made it seem like there's no way to communicate between the HTML page and the iframe. Actually, you can, but it can be a little difficult. I've written one sample out here where play the video from the iframe parent when using an iframe, but I have the play video button here and when I play that, you can see that start the video. What this uses, again, is not part of the Brightcove player API, but JavaScript post message. So this is how you can send message in. You actually have the iframe listen for an event that comes in, and you can then send information in there.

So this case, I sent a note saying, hey, when you hear this, click from the outer page, go ahead and play the video. So you can do it, but it can be challenging.

All right, so we have accomplished another task. And what we've done is we've changed the video in an iframe player implementation. And we know we did a click on a button and then we used some string manipulation basically to change the video that was playing.

Next, we're going to use a great feature of Brightcove player call plug-ins. And there are some plug-ins that are supplied by Brightcove, and then also you can create your own plug-ins. See you soon.