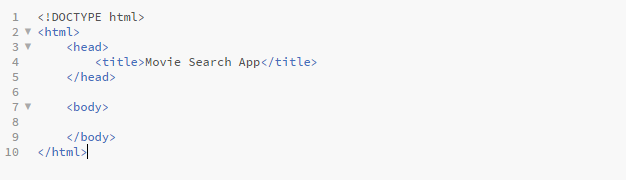
Quick Bytes Movie App

Let’s start off by laying out how our information will appear.

1. To begin, we’ll set up the standard HTML document structure and add our title.

2. Next link the CSS files and JQuery files that we need to import. These will make our layout look better on all devices.

3. Now let’s plan out what information we will displaying and where it will be displayed. Let’s start off by wrapping our entire content area in a div and give it the ID of ‘container’. Next we’ll make the input area to search for movies. Then we’ll make a label for our search bar. Be sure to notice how we added ‘for=”title”’. This will associate our label with our search field that we’ll give the ID ‘title’. To add a search bar, we’ll use an ‘input’ tag which will give us the text area we want. Now we add a button to press once the user types in the movie.

4. Before we go any further, we need to decide what information to display once the search comes back. There’s only one problem; we don’t know what information we’re going to be provided. We’ll be using an online database of movies called The Open Movie Database. We can search for a movie from our code and the website will return information about that movie. Open your browser and paste this: <http://www.omdbapi.com/?t=Up&plot=full&r=json>. The result is going to look unreadable, but I’ll format it for you to read easily

{

"Title": "Up",

"Year": "2009",

"Rated": "PG",

"Released": "29 May 2009",

"Runtime": "96 min",

"Genre": "Animation, Adventure, Comedy",

"Director": "Pete Docter, Bob Peterson",

"Writer": "Pete Docter (story by), Bob Peterson (story by), Tom McCarthy (story by), Bob Peterson (screenplay), Pete Docter (screenplay)",

"Actors": "Edward Asner, Christopher Plummer, Jordan Nagai, Bob Peterson",

"Plot": "Seventy-eight year old Carl Fredricksen travels to Paradise Falls in his home equipped with balloons, inadvertently taking a young stowaway.",

"Language": "English",

"Country": "USA",

"Awards": "Won 2 Oscars. Another 73 wins & 75 nominations.",

"Poster": "http://ia.media-imdb.com/images/M/MV5BMTk3NDE2NzI4NF5BMl5BanBnXkFtZTgwNzE1MzEyMTE@.\_V1\_SX300.jpg",

"Metascore": "88",

"imdbRating": "8.3",

"imdbVotes": "652,757",

"imdbID": "tt1049413",

}

This was a search for the movie **Up** produced by Disney. When we search for a movie on the database, it returns the information in the form of *JSON*. This is a very common format for data to be sent in and is easy for us to use. Each quotation on the left is followed by its value on the right. For example, the “Language” for the movie is “English”. We could include all of this information on our site, but we’ll just stick with the title, year, actors, plot, and the movie poster.

5. Now let’s finish our layout by adding in the elements we want to show. We’ll make each of our information pieces a ‘div’ for now. This may seem strange, but we’ll input the paragraphs and text once we get the information back from the search.

6. Now let’s get into the movie search. We’ll start by opening and closing a ‘script’ tag, where all of our JavaScript will be inserted. To start off we’ll add ‘var xmlhttp’. This will hold the xmlHTTPRequest() object used for server communication. Next, we need a way to start our search once the user presses the button. The code added will make the button trigger an event once it is clicked. To do this we’ll add an event listener that will trigger another function when the button is clicked. By using the ‘window.onload’, it will execute this code when the page first is loaded.

7.Now we need to actually search for the movie once the button is clicked. We’ll create a function called GetMovieInfo. To start, we need to get the name of the movie the user typed in. This can be done by using “document.getElementById('title').value”, which will obtain the value of the input field.

Remember that URL I had you put into your browser? That was just for the movie **Up**. But what if we want to search for different movies each time? We are going to have to make a new URL each time the user searches a movie. Instead of having the ‘?t=Up’ in our URL, we can add our title that the user searched for. This is called a query string. Now we are ready to make our database search.

8.Make a new XMLHttpRequest and have our code call a function processData (we’ll create it later), once the search is done. Finally, we open our request and pass in our URL. Because we are ‘getting’ data from the website, we need to pass in ‘GET’. To finish off, we send our request.

9.Now we are ready to use the information we got from the website and finally put it on our site. However, we only want to display the information if our search is done and it was successful. We’ll use an ‘if’ statement to check if our readyState equals 4 and our status== 200. You can read about what these mean here: <http://www.w3schools.com/ajax/ajax_xmlhttprequest_onreadystatechange.asp>

10.Now we can begin to use the information we received from the database. Start by making a variable called ‘movieJSON’, which will be set to the data we received. Then we will use JSON to *parse* the data and turn it into the form we looked at earlier. By parsing the data, we turn it into a form that can be easily used. Now we can set all the movie info we need into different variables. Finally, we can begin to put this movie info onto our page.

11. Set the ‘src’ of our image to the posterURL for the movie. This will get the image online and display it in our image tag. Then, use ‘.innerHTML’ to add elements to the webpage. By inserting new elements along with the data about the movie, our webpage will finally display our movie search. To finish off, we’ll add some styling to make our arraignment a bit better.



