### 3. What methods does the application need?

Let’s answer this by stage of the project.

0. Project Setup  
  
No methods created yet, only static html and css

1. Creating Static Components

Note that it is rather silly to store the same track information in two places (SearchResultTrack and PlayListTrack) but without passing information between components there’s not yet a way around that.

|  |  |
| --- | --- |
| App.render() | render method of App component class, to return JSX elements that will include in the root of index.html:   * <h1>Ja<span class="highlight">mmm</span>ing</h1> * a div with attribute class with value ‘App’, that includes:   + the HTML-content to be rendered by component SearchBar and   + a div with attribute class with value ‘App-playlist’, that includes:     - the HTML-content to be rendered by component SearchResultTracks     - the HTML-content to be rendered by component PlayListTracks |
| SearchBar.render() | render method of SearchBar component class, to return JSX elements that will include in DOM-element ‘App’ of document index.html:   * a div with attribute class with value ‘SearchBar’, that includes   + <input placeholder="Enter A Song Title" />   + <a>SEARCH</a>   Clicking on the link isn’t supposed to do anything yet for now. |
| SearchResultTracks.render() | render method of SearchResultTracks component class, to return JSX elements that will include in DOM-element ‘App-playlist’ of document index.html:   * a div with attribute class with value ‘SearchResults’, that includes:   + <h2>Results</h2>   + the HTML-content to be rendered by component SearchResultTrackList |
| PlayListTracks.render() | render method of PlayListTracks component class, to return JSX elements that will include in DOM-element ‘App-playlist’ of document index.html:   * a div with attribute class with value ‘Playlist’, that includes:   + <input value='New Playlist' />   + the HTML-content to be rendered by component PlayListTrackList   + <a class="Playlist-save">SAVE TO SPOTIFY</a> |
| SearchResultTrackList.render() | render method of TrackList component class, to return JSX elements that will include in DOM-element ‘SearchResults’ or ‘Playlist’ of document index.html:   * a div with attribute class with value ‘TrackList’, that includes five times:   + the HTML-content to be rendered by component SearchResultTrack |
| PlayListTrackList.render() | render method of PlayListTrackList component class, to return JSX elements that will include in DOM-element ‘Playlist’ of document index.html:   * a div with attribute class with value ‘TrackList’, that includes three times:   + the HTML-content to be rendered by component PlayListTrack |
| SearchResultTrack.render() | render method of SearchResultTrack component class, to return JSX elements that will include in DOM-element ‘TrackList’ of ‘SearchResults’ of document index.html:   * a div with attribute class with value ‘Track’, that includes:   + <div class="Track-information">  <h3><song placeholder></h3>  <p><Artis placeholder> | <Album placeholder></p> </div>   + <a class="Track-action">+</a> * it should take the information in the placeholders from a hard-coded object in SearchResultTrack.js that contains key-value pairs for one track, with properties song, artist and album, with dummy values. |
| PlayListTrack.render() | render method of PlayListTrack component class, to return JSX elements that will include in DOM-element ‘TrackList’ of ‘PlayList’ of document index.html:   * a div with attribute class with value ‘Track’, that includes:   + <div class="Track-information">  <h3><song placeholder></h3>  <p><Artis placeholder> | <Album placeholder></p> </div>   + <a class="Track-action">-</a> * it should take the information in the placeholders from a hard-coded object in PlayListTrack.js that contains key-value pairs for one track, with properties song, artist and album, with dummy values. |

2. Passing information to Components  
  
Not we don’t need two separate Track-components classes anymore, since we can now pass information representing whether the last line in each track should display a plus or minus sign. In the next step we will add a dummy method so clicking on the plus or minus will display some message to the console.

Now we can delete PlayListTrackList.js and rename SearchResultTrackList.js to TrackList.js and refactor the names inside TrackList. Also we can now delete PlayListTrack.js and rename SearchResultTrack.js to Track.js and refactor the names inside to Track.

Change some render methods:

|  |  |
| --- | --- |
| App.render() | render method of App component class, to return JSX elements that will include in the root of index.html:   * <h1>Ja<span class="highlight">mmm</span>ing</h1> * a div with attribute class with value ‘App’, that includes:   + the HTML-content to be rendered by component SearchBar and   + a div with attribute class with value ‘App-playlist’, that includes:     - the HTML-content to be rendered by component SearchResultTracks       * *now with an attribute that passes an array with five dummy searchResultTracks*     - the HTML-content to be rendered by component PlayListTracks       * *now with an attribute that passes an array with three dummy playListTracks* * *it should take the information in the arrays from two hard-coded arrays of objects (searchResultTracks and playListTracks) in App.js (cut/paste from SearchResultTrack.js and PlayListTracks.js) that contain key-value pairs for several copies of one track, with properties song, artist and album, with dummy values.* |
| SearchBar.render() | no change |
| SearchResultTracks.render() | render method of SearchResultTracks component class, to return JSX elements that will include in DOM-element ‘App-playlist’ of document index.html:   * a div with attribute class with value ‘SearchResults’, that includes:   + <h2>Results</h2>   + the HTML-content to be rendered by component TrackList     - *now with an attribute that passes the passed in array with five dummy searchResultsTracks* |
| PlayListTracks.render() | render method of PlayListTracks component class, to return JSX elements that will include in DOM-element ‘App-playlist’ of document index.html:   * a div with attribute class with value ‘Playlist’, that includes:   + <input value='New Playlist' />   + the HTML-content to be rendered by component TrackList     - *now with an attribute that passes the passed in array with three dummy playListTracks*   + <a class="Playlist-save">SAVE TO SPOTIFY</a> |
| TrackList.render() | render method of TrackList component class, to return JSX elements that will include in DOM-element ‘SearchResults’ or ‘Playlist’ of document index.html:   * a div with attribute class with value ‘TrackList’, that includes five times:   + the HTML-content to be rendered by component Track   + *it should take the information to be passed on to Track from the info now passed in from App (through SearchResultTracks or PlayListTracks)*   + *this should now be accomplished by processing the passed in array of tracks, instead of a five / three times hard-coded rendering of the SearchResultTrack or PlayListTrack component.*   + *further, it should take an attribute that passes a ‘plus’ value* |
| Track.render() | render method of Track component class, to return JSX elements that will include in DOM-element ‘TrackList’ of ‘SearchResults’ of document index.html:   * a div with attribute class with value ‘Track’, that includes:   + <div class="Track-information">  <h3><song placeholder></h3>  <p><Artis placeholder> | <Album placeholder></p> </div>   + <a class="Track-action">+</a> * *it should take the information in the placeholders from the info now passed in by TrackList.instead of from the hard-coded object in this file* |

3. Setting the State of Components

Now we will deal with handling events. Five things can happen in the app:

* User can type a search term in SearchBar
  + SearchBar will then need to update its state-property ‘term’
* User can click SEARCH in SearchBar
  + SearchBar will then need to call an App-method that will handle the request to Spotify
  + App will need to handle the actual (GET-) request (dummy method for now)
* User can click + in Track in SearchResultTracks
  + Track will then need to call an App-method that will update App’s state-property playListTracks (which will now result in PlayListTracks showing the added track, since passing in that info was already set up)
* User can click - in Track in PlayListTracks
  + Track will then need to call a App-method that will update App’s state-property playListTracks (which will now result in PlayListTracks not showing the removed track anymore)
* User can click SAVE TO SPOTIFY in PlayListTracks
  + PlayListTracks will then need to call an App-method that will handle the request to Spotify
  + App will need to handle the actual (POST-) request (dummy method for now)

Note that there is no visual feedback in the SearchResultTracks component as to whether a track has been added to the playlist or not. That might be a nice feature to add in Unit 4 of the course. It may be removed from the search results or be marked is some visual way as being added to the playlist.

|  |  |
| --- | --- |
| App.searchSprotify(term) | (dummy) method of App component class, to search Spotify for tracks (by Song, Album or Artist, using full-text search). Method will now only show message in console: ‘Searching Spotify with $(term)’. |
| App.addTrack(track) | method of App component class, to add track to state-property playListTracks (array of Tracks). |
| App.saveToSpotify(track) | (dummy) method of App component class, to save trackinfo (Song, Album, Artist) to the account of the logged in user on Spotify. Method will now only show message in console: ‘Saving track ($song, $album, $artist) to Spotify’. |
| SearchBar.handleTermChange(e) | eventHandler method of SearchBar component class, for handling triggering of eventListener for event change of content of field ‘term’. This method will change state of state-property ‘term’. |
| SearchBar.handleSearch(e) | eventHandler method of SearchBar component class, for handling triggering of eventListener for event click on link with content ‘SEARCH’. This method will call searchSpotify method of App component. |
| SearchResultTracks.handleAdd(track) | eventHandler method of SearchResultTracks component class, for handling triggering of eventListener for event click on link with content ‘+’ . This method will call addTrack method of App component. |
| PlayListResults.handleRemove(track) | eventHandler method of PlayListTracks component class, for handling triggering of eventListener for event click on link with content ‘-’ . This method will remove track from the passed-in prop playListTracks. |
| PlayListResults.handleSave(e) | eventHandler method of PlayListResults component class, for handling triggering of eventListener for event click on link with content ‘SAVE TO SPOTIFY’. This method will call saveToSpotify method of App component. |

4. Interacting with the API

The two methods that interact with the API will now be updated and three methods will be added, to object Spotify in new file Spotify.js.

|  |  |
| --- | --- |
| App.searchSprotify(term) | method of App component class, to search Spotify for tracks (by Song, Album or Artist, using full-text search). This method will call search method of Spotify object. |
| App.saveToSpotify(track) | method of App component class, to save trackinfo (Song, Album, Artist) to the account of the logged in user on Spotify. This method will call save method of Spotify object. |
| Spotify.search(term) | method of Spotify object, that will send the search GET-request to Spotify API endpoint and return an array of JSON objects with track-info Song, Artist and Album for each track.. |
| Spotify.save(track) | method of Spotify object, that will send the save POST-request to Spotify API endpoint and log the response or network-error to the console. |
| Spotify.getAccessToken() | method of Yelp object that will be called by the other two methods in this object, before creating and sending their actual request. |