React – Intro

In this lab, you will create a React app in an Express app.

# Objectives

In this lab, you will

* Use an existing Express app
* Create an initial index.html file
* Import the React framework using <script> tags
* Write the React app in the /public/javascripts/components.js file
* Run the app and add some Todo items

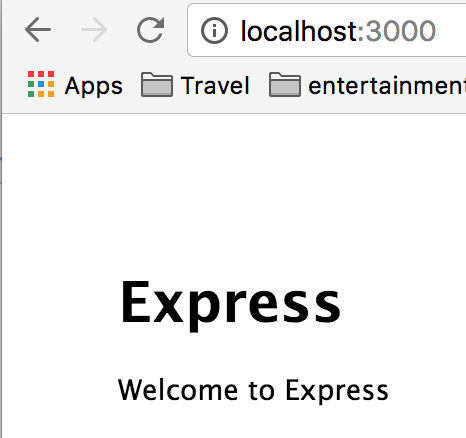
# Run the Kraken app to verify correctness

1. The lab folder already contains an Express app. You can create your own by running express from the command line. Change to the lab folder and run:

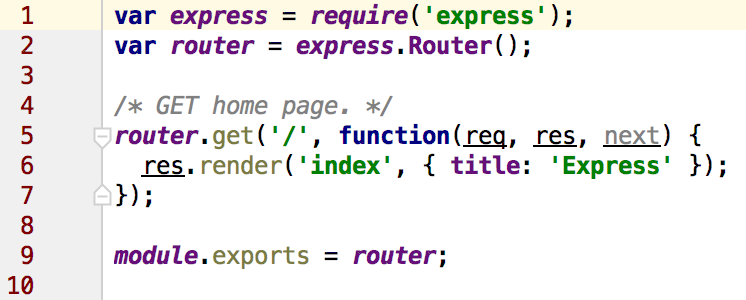
npm install

npm start

1. Open the browser to localhost:3000 and the page should look like this:



1. The output comes from routes/index.js as shown. This renders the views/index.jade file producing the above page.



# Configure Express to use React

1. Create the /public/react.html file with the following contents.

<!DOCTYPE **html**>  
<**html lang="en"**>  
<**head**>  
 <**meta charset="UTF-8"**>  
 <**title**>TODO List</**title**>  
 <**link rel="stylesheet" href="stylesheets/style.css"**>  
</**head**>  
<**body**>  
<**h1**>This is the react.html file.</**h1**>

<**div id="todoListBox"**></**div**>  
  
<**script src="components/jquery/dist/jquery.js"**></**script**>  
<**script src="components/react/react.js"**></**script**>  
<**script src="components/react/react-dom.js"**></**script**>  
<**script src="components/babel/browser.js"**></**script**>  
<**script src="javascripts/components.js" type="text/babel"**></**script**>  
</**body**>  
</**html**>

1. Notice the JavaScript libraries at the bottom of the page. We will use bower to manage these files. Bower is a tool much like npm that manages libraries for the client, not the server.
2. Create a new bower configuration file using:

bower init

Use the default values for all inputs.

1. Edit the bower.json file to add the path for the modules we are about to install. Add the indicated directory directive:

{  
 **"name"**: **"lab"**,  
 **"description"**: **""**,  
 **"directory"**: **"public/components"**,  
 **"main"**: **""**,  
 **"authors"**: [  
 **"Clay Boyd <clboyd@paypal.com>"** ],  
 **"license"**: **"MIT"**,  
 **"homepage"**: **""**,  
 **"ignore"**: [  
 **"\*\*/.\*"**,  
 **"node\_modules"**,  
 **"bower\_components"**,  
 **"public/components"**,  
 **"test"**,  
 **"tests"** ]  
}

1. Create a file, .bowerrc and add the following text:

{**"directory"** : **"public/components"**}

1. Now install the components with:

bower install jquery react babel --save

1. Create the React component file, /public/javascripts/components.js with the following content

*// add the TodoItem below  
  
// add the TodoListBox below****ReactDOM***.**render**(  
 <**TodoListBox** />, ***document***.getElementById(**'todoListBox'**)  
);

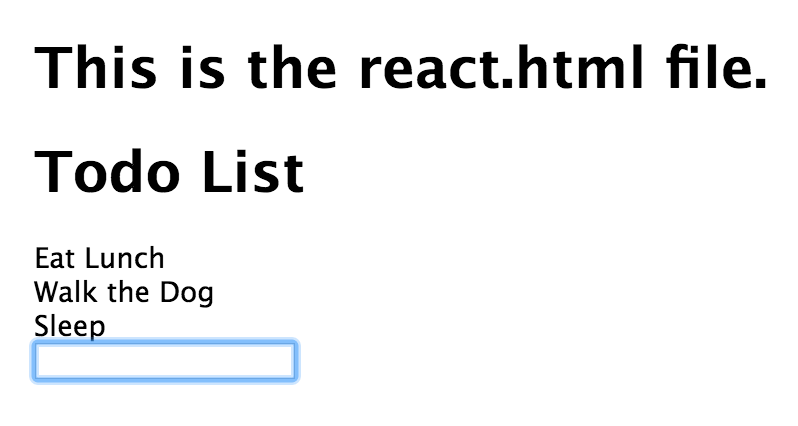
1. Use the following code for the TodoItem definition:

*// add the TodoItem below***var *TodoItem*** = ***React***.**createClass**( {  
 render() {  
 **return** (  
 <**div**> {**this**.**props**.**content**} </**div**>  
 );  
 }  
});

1. Notice the reference to the content attribute used by the parent.
2. Use the following code for the TodoListBox definition:

*// add the TodoListBox below***var *TodoListBox*** = ***React***.**createClass**({  
  
 getInitialState() {  
 **return** {  
 **todos**: [**'Eat Lunch'**, **'Walk the Dog'**, **'Sleep'**],  
 **current**: **''** };  
 },  
  
 render() {  
 **var** views = **this**.state.**todos**.**map**((item, idx) => <**TodoItem key=**{idx} **content=**{item} />);  
  
 **return** (  
 <**div className="todo-list-box"**>  
 <**h1**>***Todo*** List</**h1**>  
 {views}  
 <**form onSubmit=**{**this**.addOne}>  
 <**input type="text" onChange=**{**this**.change} **value=**{**this**.state.**current**}/>  
 </**form**>  
 </**div**>  
 );  
 },  
 change: **function** (event) {  
 event.preventDefault();  
 **this**.setState({**'current'**: event.**target**.**value**});  
 },  
 addOne: **function**( event ) {  
 event.preventDefault();  
 **var** item = **this**.**state**.**current**.trim();  
 **if** (item.**length** > 0) {  
 **this**.setState( {**'todos'**: **this**.**state**.**todos**.concat( item ), **'current'**: **''**});  
 }  
 },  
  
});

1. Notice how it passes the todo item to the <TodoItem> component (in yellow).
2. Notice how it links to the controller code (in green).
3. Open the page, <http://localhost:3000/react.html> which loads the React component. The final page should look like



Congratulations. You have completed this lab.