Promises have replaced callback functions as the preferred programming style for handling asynchronous calls. A promise is a holder for a result (or an error) that will become available in the future (when the async call returns). Promises have been available in JavaScript through third-party libraries (for example, [jQuery](https://api.jquery.com/promise/) and [q](https://github.com/kriskowal/q)). ECMAScript 6 adds built-in support for promises to JavaScript.

In this unit, you create a simple application called ratefinder that returns a list of available mortgage rates.

## Part 1: Use a Promise

To illustrate the use of promises in this example, you use the new fetch() function. At the time of this writing, fetch() is available in the latest version of Chrome, Firefox, and Opera, but not in IE and Safari. You can check the current availability of fetch() [here](http://caniuse.com/#feat=fetch). You can read more about fetch() [here](http://jakearchibald.com/2015/thats-so-fetch/).

1. Create a file named ratefinder.html in the es6-fundamentals-student directory. implemented the file as follows:



1. Create a file named ratefinder.js in the es6-fundamentals-student/js directory. implemented the file as follows:



To keep things simple, this code uses a static data file: rates.json. The application would work the same way with a URL pointing to a remote service.

1. Open webpack.config.js in your code editor. In module.exports, modify the entry and output items as follows:



The **webpack** script will now compile two applications: **main.js** and **ratefinder.js**. It will create two compiled files based on the entry name: **app.bundle.js** and **ratefinder.bundle.js**.

1. On the command line, type the following command to rebuild the application:

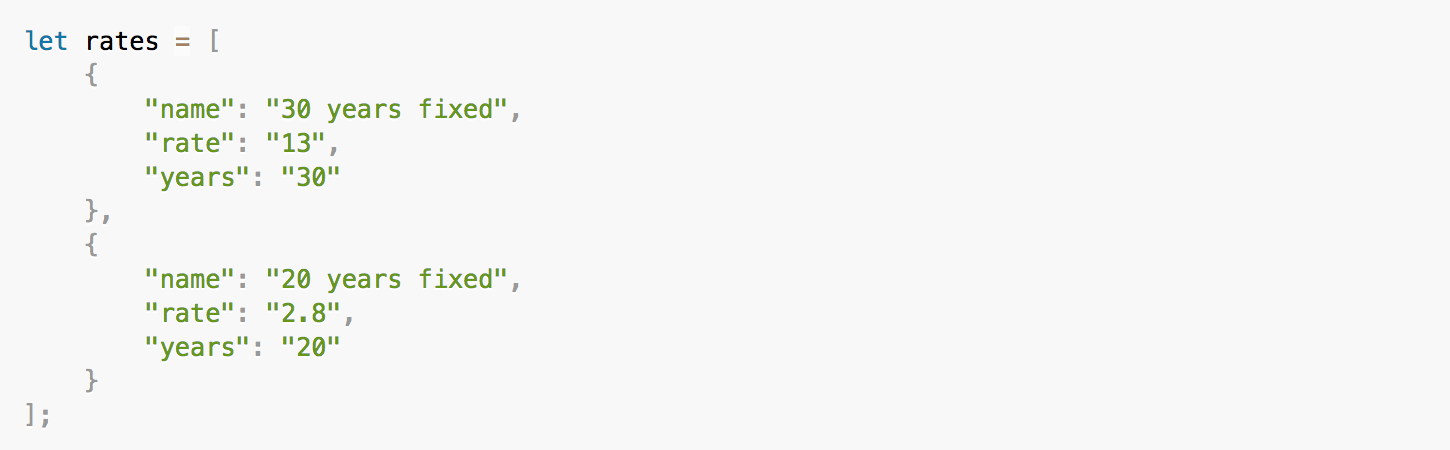
npm run webpack

1. Open a browser, access <http://localhost:8080/ratefinder.html>.

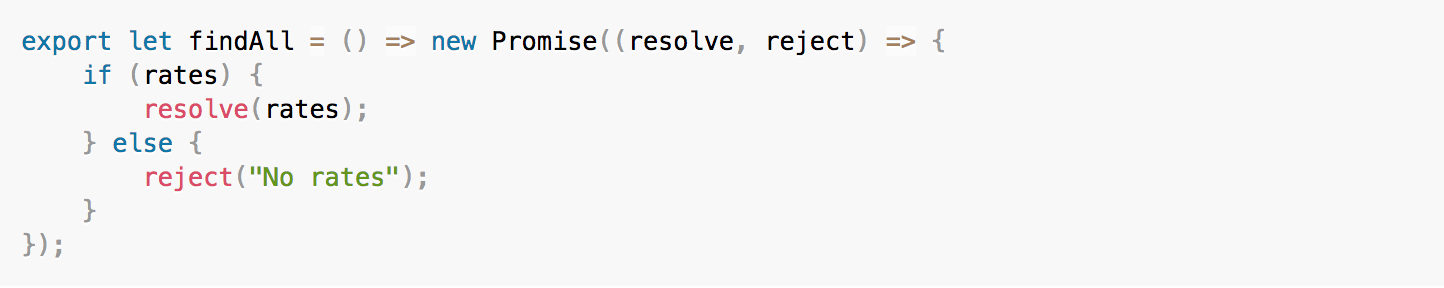
## Part 2: Create a Promise

Most of the time, all you’ll have to do is use promises returned by built-in or third-party APIs. Sometimes, you may have to create your own promises as well. In this section you create a mock data service to familiarize yourself with the process of creating ECMAScript 6 promises. The mock data service uses an asynchronous API so that it can replace an actual asynchronous data service for test or other purpose.

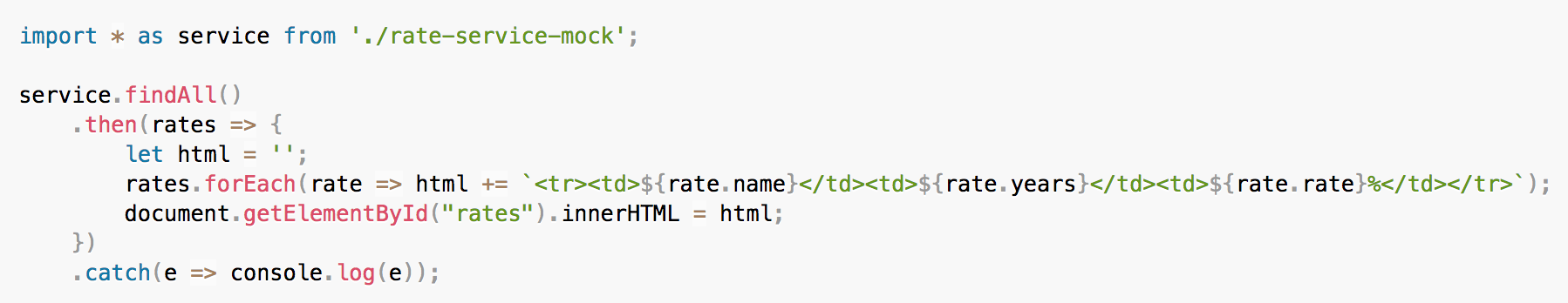
1. Create a new file named rate-service-mock.js in the js directory.
2. In rate-service-mock.js.js, define a rates variable with some sample data:



1. Define a findAll() function implemented as follows:



1. Open ratefinder.js. Change the implementation as follows:



1. On the command line, type the following command to rebuild the application:

npm run webpack

1. Open a browser, access <http://localhost:8080/ratefinder.html>.