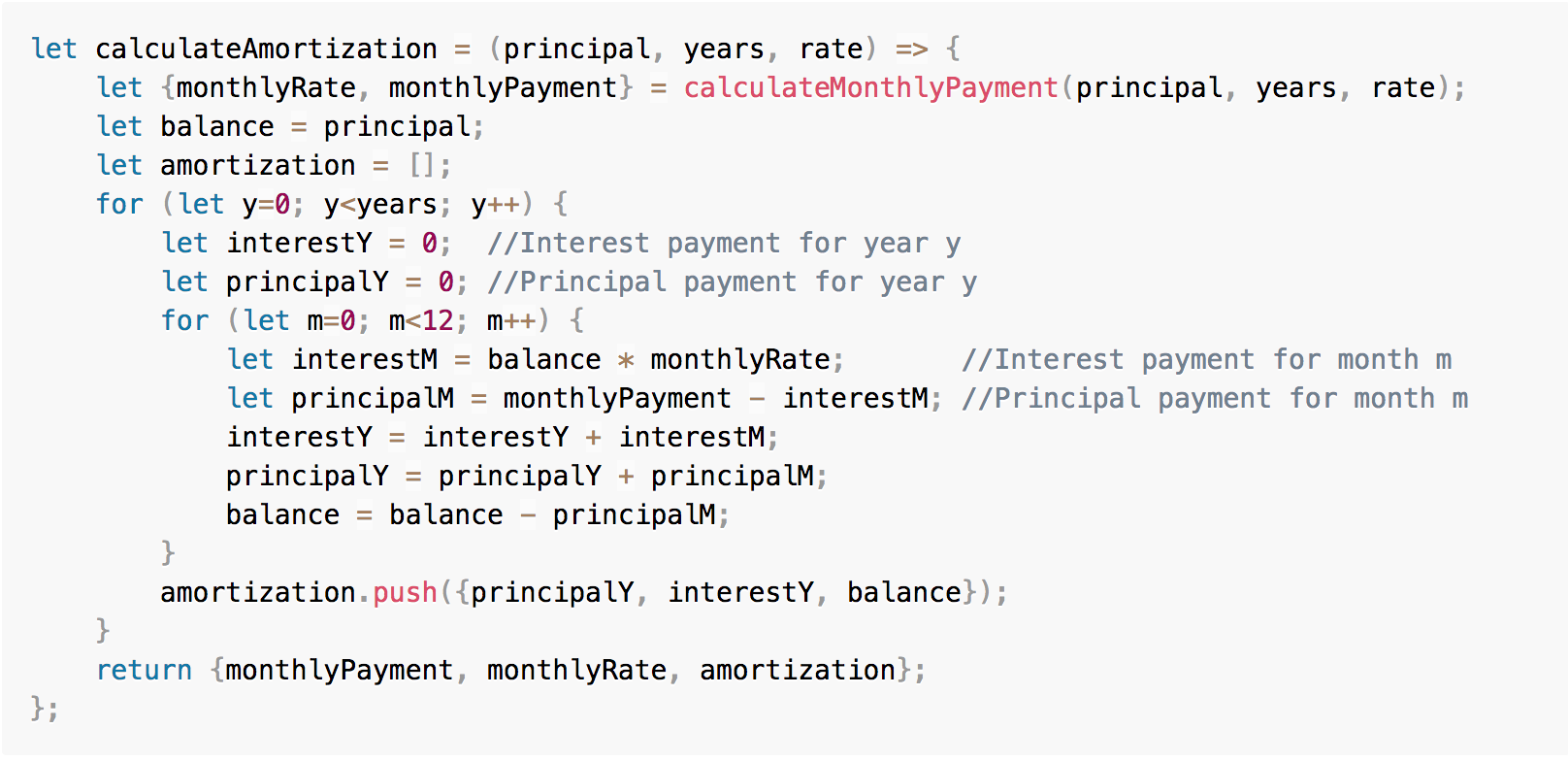
The ECMAScript 6 arrow function syntax is a shorthand for the ECMAScript 5 function syntax. It supports both block and expression bodies. The value of this inside the function is not altered: it is the same as the value of this outside the function. No more var self = this to keep track of the current scope.

In this unit, you add a new function to calculate the mortgage amortization. You also modify the existing functions to use the new ECMAScript 6 arrow function syntax.

1. Open js/main.js. Right after the calculateMonthlyPayment function, add a calculateAmortization function defined as follows:



1. Modify the calculateMonthlyPayment function signature as follows:

let calculateMonthlyPayment = (principal, years, rate) => {

1. Modify the signature of the **calcBtn** click event handler as follows:

document.getElementById('calcBtn').addEventListener('click', () => {

1. In the **calcBtn** click event handler, invoke calculateAmortization function instead of calculateMonthlyPayment:

let {monthlyPayment, monthlyRate, amortization} = calculateAmortization(principal, years, rate);

1. As the last line of the **calcBtn** click event handler, log amortization data to the console (you’ll display the amortization table in the application in the next unit):

amortization.forEach(month => console.log(month));

This is an example of an expression body.

The complete implementation of the button click handler looks like this:



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

npm run babel

1. Open a browser, access [http://localhost:8080](http://localhost:8080/), and click the **Calculate** button. Open the developer console: you should see the amortization values in the console log.

