WEB APPLICATION ARCHITECTURES

Events

Objectives

In this exercise you will learn how to access DOM elements using code

Reference material

This exercise is based on material from the "Events" chapter.

Overview

- In this lab you'll exercise accessing DOM elements using code.

Estimated duration

The estimated duration for this lab is 25 minutes.

Completed solution

There is a completed solution for this lab.

Step by step instructions

1. Open the project you were working on in the previous lab

2. Add a new property to the $scope object, called "currentProduct", and initialise it to null

3. Add a <div> to the html page, perhaps underneath the table of data. In the <div>, add an ng-hide directive that will hide the <div> if currentProduct==null

4. Inside the <div>, show some details of the product held in the currentProduct property – perhaps its name and unit price. Make sure you format the unit price (or any other details) correctly. Hint: {{currentProduct.unitPrice | currency:'£'}}

5. Add a function to the products array called "selectProduct" which will, soon, handle click events. Hint: $scope.products.selectProduct = function (… etc.

   This function should take a parameter, called "product". It should set $scope.currentProduct to be equal to this parameter.

6. Add a click handler to the element which represents a row of data. The click handler should invoke the products.selectProduct() method, and should pass the current product as its parameter. Hint: the element to which you should add the click handler will probably be the same element that has the ng-repeat attribute.

Discussion

Take a good look at the code in the JavaScript controller.

Notice that there is nothing in this code which relies on the HTML doing anything specific. This is a key point in understanding how AngularJS applications work.

The controller simply:

- Keeps the data

- Keeps some properties about how the data should be displayed (but does not care how the HTML uses these properties)

- Knows what to do when a product is selected (but does not care *how* a product is selected, nor what is done with the selected product)

This allows for the HTML to be changed in the future, or completely re-written, without having to change the JavaScript at all.

Compare this to either pure JavaScript or to JQuery: in both cases, you would be making extensive use of classes and ids in your HTML, and the JavaScript would make use of these classes and ids to interact with the page. If the page changed in the future, it could completely break the application, because the new page may not use the same classes, etc., as the old page – but with AngularJS this will never be the case if you design your application correctly.