**SERVICES**

Services I

So far we've made AngularJS apps by adding data to a controller, and then displaying it in a view.

But what happens when the data contains hundreds of items, or if it's constantly changing like weather or financial data? Hardcoding data into a controller won't work anymore.

A better solution is to read the live data from a server. We can do this by creating a *service*.

**1.**

In the browser frame on the right, visit <https://s3.amazonaws.com/codecademy-content/courses/ltp4/forecast-api/forecast.json>. It's a JSON object containing a city\_name and an array days containing weather data for the next five days.

**2.**

Create a service named forecast that fetches the weather data from the server. In the new file **js/services/forecast.js**. Type in this code exactly as you see here:

app.factory('forecast', ['$http', function($http) { return $http.get('https://s3.amazonaws.com/codecademy-content/courses/ltp4/forecast-api/forecast.json') .success(function(data) { return data; }) .error(function(err) { return err; }); }]);

**3.**

Include **js/services/forecast.js** in **index.html** in line 62 as a new <script> element.

**4.**

In the controller, modify MainController by passing in the forecast service, like this:

app.controller('MainController', ['$scope', 'forecast', function($scope, forecast) { // ... }]);

**5.**

Inside MainController, use the forecast service to save the weather data into $scope.fiveDay, like this:

forecast.success(function(data) { $scope.fiveDay = data; });

Services II

Well done! The city\_name now shows up in the view. How does it work?

1. First in **js/services/forecast.js**, we made a new *service*. We used app.factory to create a new service named forecast
2. The forecast service needs to use AngularJS's built-in $http to fetch JSON from the server. Therefore, we add $http to the forecast service as a dependency, like this:

['$http', function($http) { // ... }]

Now $http is available to use inside forecast.

1. Then, inside forecast, we use $http to construct an HTTP GET request for the weather data. If the request succeeds, the weather data is returned; otherwise the error info is returned.
2. Next in the controller, we used the forecast service to fetch data from the server. First we added forecastinto MainController as a dependency so that it's available to use. Then within the controller we used forecast to asynchronously fetch the weather data from the server and store it into $scope.fiveDay
3. As before, any properties attached to $scope become available to use in the view. This means in **index.html**, we can display the city\_name using an expression as done before.

**1.**

In the browser, visit <https://s3.amazonaws.com/codecademy-content/courses/ltp4/forecast-api/forecast.json>. Looking at the format of the data in the days array, each day has four pieces of data - datetime, icon, high, and low.

**2.**

Visit [http://localhost:8000](http://localhost:8000/) to view the AngularJS app. Let's finish the view so that it displays the weather for each day.

Notice in the view, we're using ng-repeat to loop through each item in the days array. Inside this loop, display a day's four pieces of data.

Remember to use ng-src to display an image.

**3.**

Use the date filter to format the datetime.

Generalizations

Why are services useful? Instead of filling the controller with code to fetch weather data from a server, it's better to move this independent logic into a service so that it can be reused by other parts of the app.

What can we generalize so far?

* Directives are a way to make standalone UI components, like <app-info>
* Services are a way to make standalone communication logic, like forecast which fetches weather data from a server