



# Learn Angular

BY MAKING A QUIZ APP  
FROM SCRATCH

[HUNGRYTURTLECODE.COM](http://HUNGRYTURTLECODE.COM)

## Part 19



# Check Out The Whole Course Index

1. [Getting Started](#)
2. [Ng-controller directive and the \(mis\)use of \\$scope](#)
3. [Looping around with the ng-repeat directive](#)
4. [Markup for the Bootstrap Modal](#)
5. [Using Angular Filters to create real time search](#)
6. [The powerful ng-click directive](#)
7. [Services in Angular make everything easier](#)
8. [What is this infamous dependency injection in Angular?](#)
9. [Let's build a Factory](#)
10. [The ng-class directive](#)
11. [More Bootstrap markup – the well](#)
12. [Adding some logic to the controller](#)
13. [Making things disappear with ng-if](#)
14. [The \\$index property for ng-repeat](#)
15. [Reusing code is always a good idea](#)
16. [Using Bootstrap to help styling an error message](#)
17. [The final prompt after the quiz](#)
18. [Marking the quiz](#)
19. You are here
20. [Reusing and slightly modifying some previous Bootstrap](#)
21. [More than one way to use ng-class](#)
22. [Another Angular Filter](#)
23. [More usage of Ng-if](#)
24. [Finishing The App](#)

## First Steps To Getting The Results

We have finished the quiz controller so now it is time to start the results controller. In the [last part](#) we created a property called resultsActive, which is what we will use to trigger the results area to show using an ng-show. We will also delve into some more angular dependency injection.

If you want to see the app for yourself, [check it out here](#).

The git repo [can be found here](#).



Click the image to go to the video on youtube or go here [https://www.youtube.com/watch?v=N6N3KhkKk3o&list=PLqr0oBkln1FBmOjK24\\_B4y\\_VAA8736wPq&index=19](https://www.youtube.com/watch?v=N6N3KhkKk3o&list=PLqr0oBkln1FBmOjK24_B4y_VAA8736wPq&index=19)

[The next part can be found here](#)

Before we can start showing the results area we need to create it. So in the HTML, under the quiz controller markup let's create a new div with a new [ng-controller directive](#). Again using the “controller as” syntax.

```
<div ng-controller="resultsCtrl as results">
  <!-- results controller markup -->
</div>
```

Now the controller itself needs to be created. So create a new file called results.js inside the controllers directory. The script will be started with an IIFE as using and the rest should also look familiar to you by now.

```
(function(){
  angular
    .module("turtleFacts")
    .controller("resultsCtrl", ResultsController);

  function ResultsController(quizMetrics, DataService){
    var vm = this;
  }
})();
```

## More Angular Dependency Injection

I mentioned that we will change the visibility of the results controller markup depending on the value of the resultsActive property on the quizMetrics factory object. This means that we need access to that factory from inside the results controller – this means dependency injection!

```
ResultsController.$inject = ['quizMetrics', 'DataService'];

function ResultsController(quizMetrics, DataService){
  // Code for results controller here
}
```

Also don't forget to add the script tag to the bottom of the html page.

```
<!-- Our application scripts -->
<script src="js/app.js"></script>
<script src="js/controllers/list.js"></script>
<script src="js/controllers/quiz.js"></script>
<script src="js/controllers/results.js"></script>
<script src="js/factories/quizMetrics.js"></script>
<script src="js/factories/dataservice.js"></script>
```

With the quizMetrics injected into the results controller we can use the ng-show directive to display the results markup.

```
<div ng-controller="resultsCtrl as results" ng-show="results.quizMetrics.resultsActive">
</div>
```

## A Bit Of A Problem

If you now run through the application you will see that when the user clicks yes on the prompt at the end of the quiz, the list controller shows again. This is because we only hide the list controller when quizActive is true. But when the yes button is clicked in that prompt it sets quizActive back to false and resultsActive to true. So the list controller shows with the results controller below it.

We need to amend this by changing the ng-hide on the list controller to hide when either the quiz controller or the results controller is active.

```
<div ng-controller="listCtrl as list" ng-hide="list.quizMetrics.quizActive || list.quizMetrics.resultsActive">
</div>
```

## On To The Next One, On To The Next One...

With that done, we now have the results controller showing when it should be. The [next step](#) from here is to start populating the results controller with some content. That is what we will do in the [next part](#).

See you in [part 20](#)

Adrian