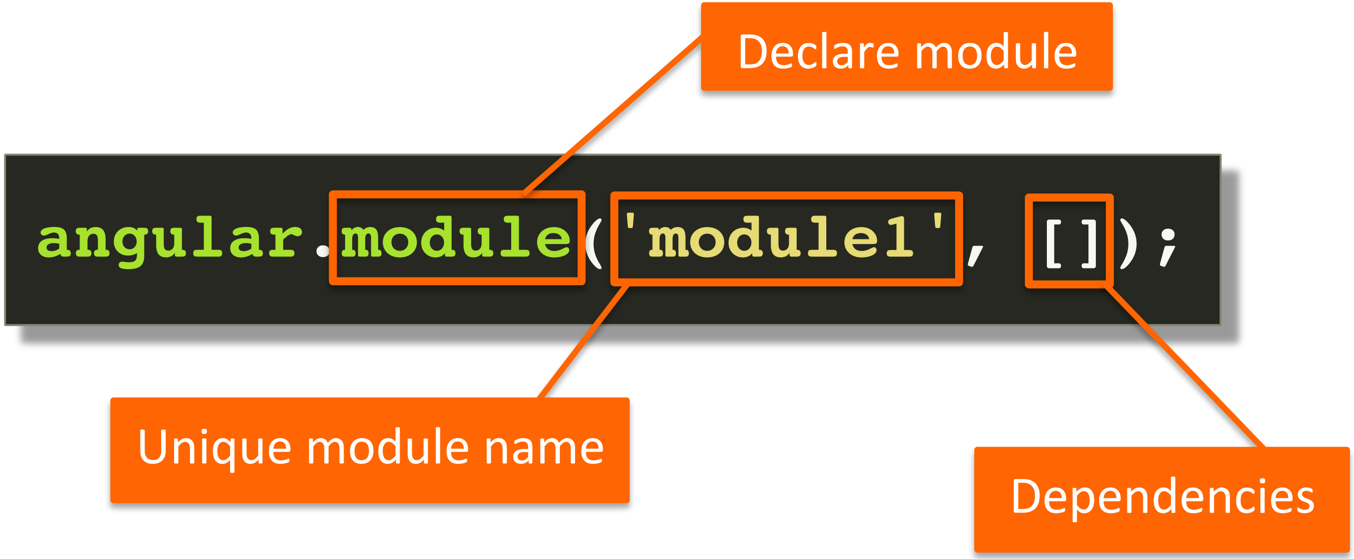
Modules

Step 1: Declare/Create Module



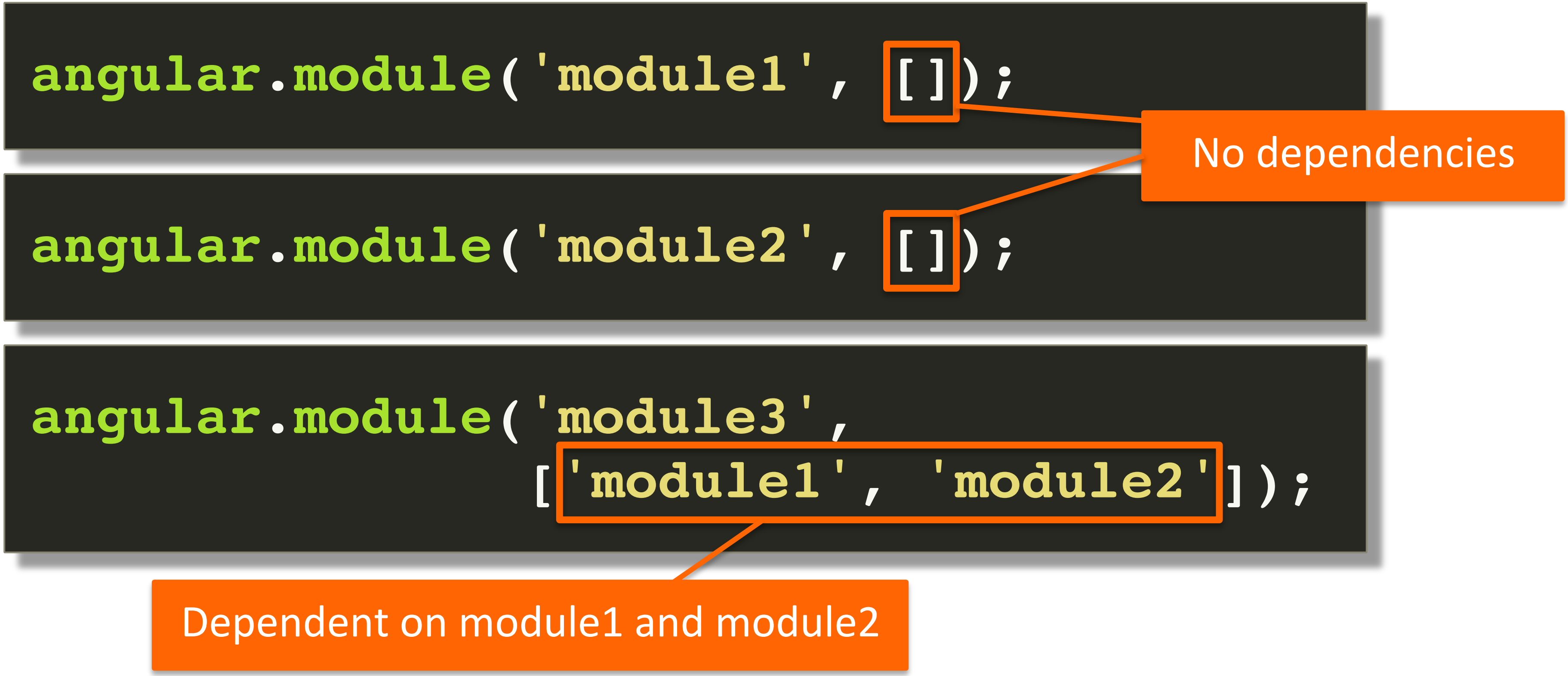
```
angular.module('module1', []);
```

Declare module

Unique module name

Dependencies

Step 1: Specify Other Modules As Dependencies



```
angular.module('module1', []);
```

```
angular.module('module2', []);
```

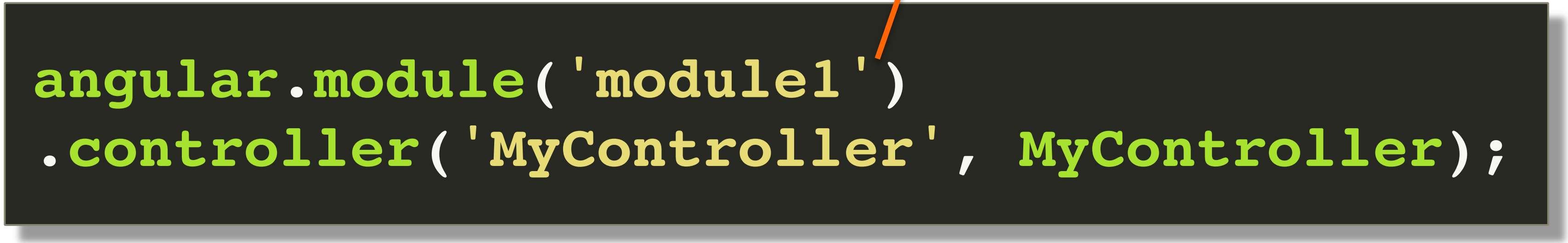
```
angular.module('module3',
               ['module1', 'module2']);
```

No dependencies

Dependent on module1 and module2

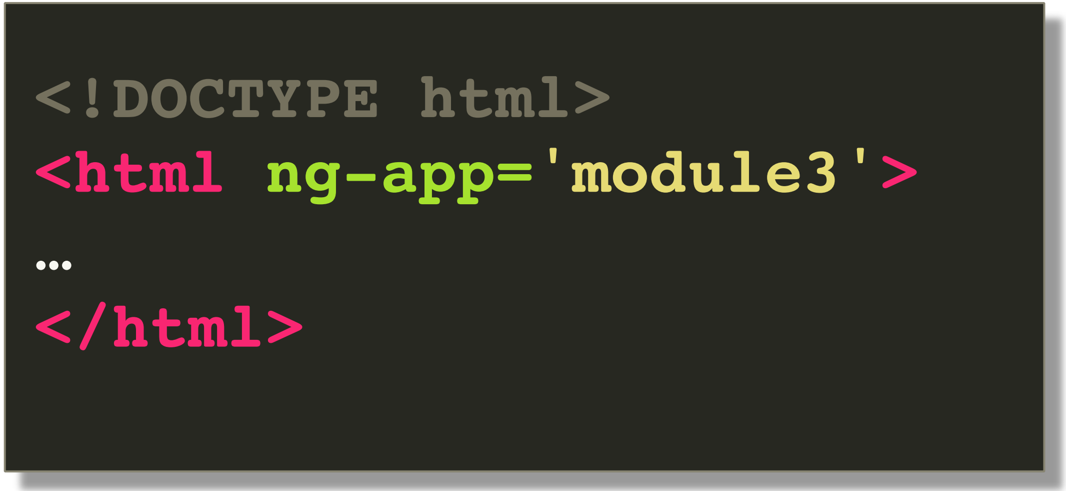
Step 2: Declare Module Artifacts

Second argument missing!



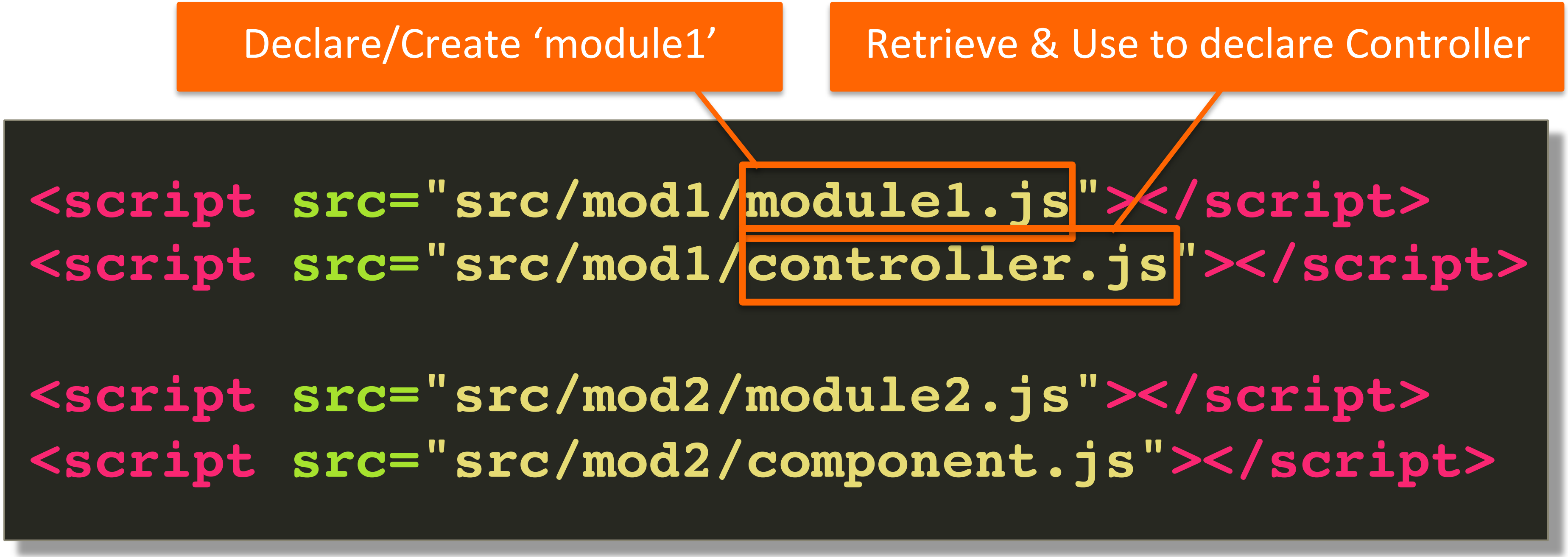
```
angular.module('module1')
.controller('MyController', MyController);
```

Step 3: ng-app=‘MainModule’



```
<!DOCTYPE html>
<html ng-app='module3'>
…
</html>
```

Splitting Javascript into Several Files



Declare/Create ‘module1’

Retrieve & Use to declare Controller

```
<script src="src/mod1/module1.js"></script>
<script src="src/mod1/controller.js"></script>

<script src="src/mod2/module2.js"></script>
<script src="src/mod2/component.js"></script>
```

Splitting Javascript into Several Files

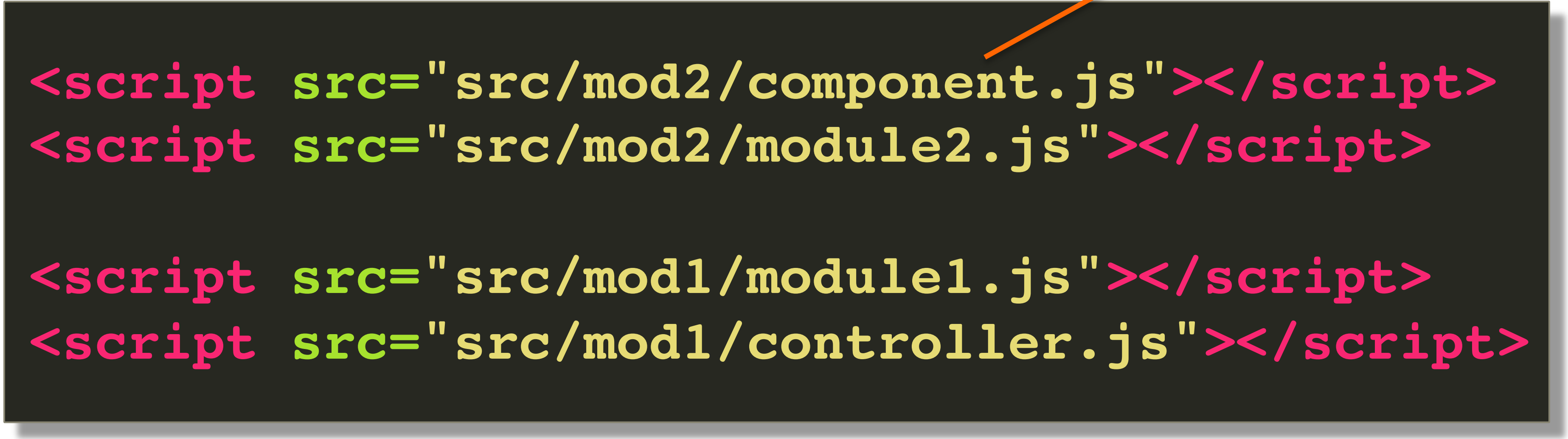
✔ module2 before module1

```
<script src="src/mod2/module2.js"></script>
<script src="src/mod2/component.js"></script>

<script src="src/mod1/module1.js"></script>
<script src="src/mod1/controller.js"></script>
```

Splitting Javascript into Several Files

Component uses module2



```
<script src="src/mod2/component.js"></script>
<script src="src/mod2/module2.js"></script>

<script src="src/mod1/module1.js"></script>
<script src="src/mod1/controller.js"></script>
```

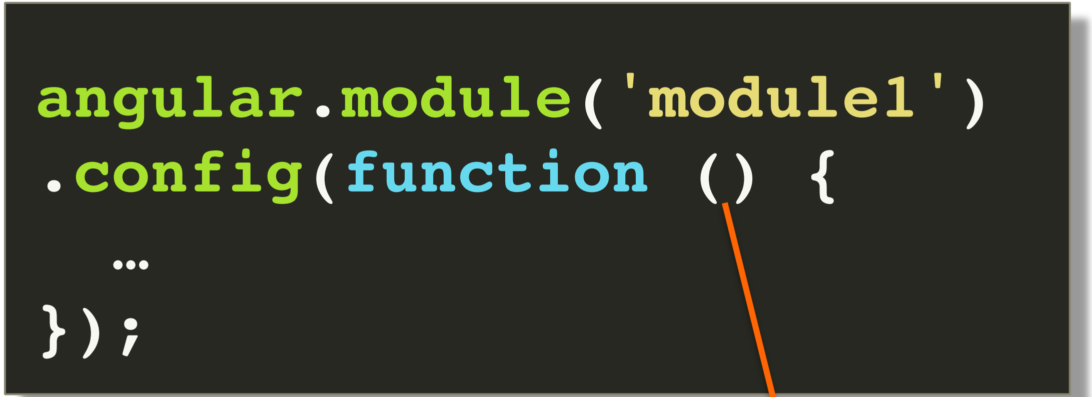
Splitting Javascript into Several Files

✗ Using ‘module2’ before it’s created

```
angular.module('module2').component({…});

angular.module('module2', []);
```

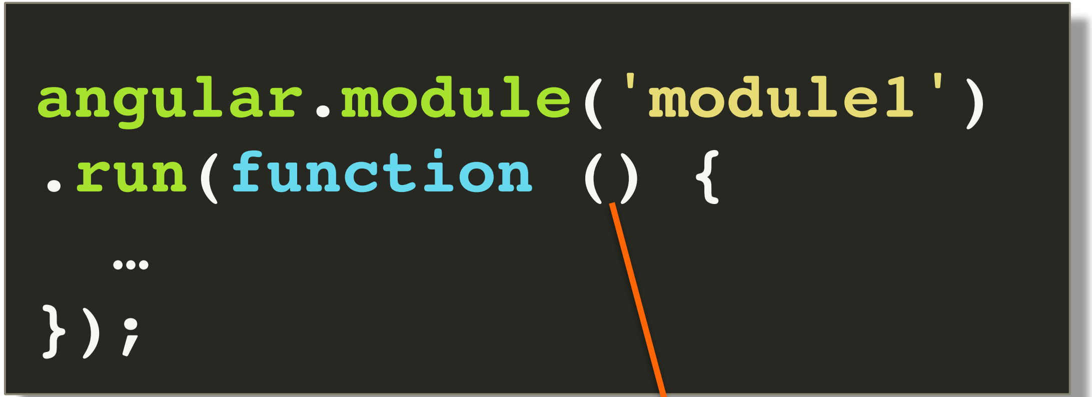
Configuration and Run Blocks



```
angular.module('module1')
.config(function () {
   …
});
```

Inject only Providers and constants

Configuration and Run Blocks



```
angular.module('module1')
.run(function () {
   …
});
```

Inject only instances
(like services) and constants

Multiple Modules Config & Run

```
angular.module('module1', []);
```

```
angular.module('module2', []);
```

```
angular.module('module3',
               ['module1', 'module2']);
```

Summary

- angular.module method takes 2 args to <u>create</u> a module:
  - Name of module
  - Array of string module name dependencies
- angular.module method with just name of module retrieves the previously created module
  - Then, you can declare components, controllers, etc., on it
- module.config method fires before module.run method
- All dependency modules get configured first
- It doesn’t matter which modules are listed first as long as module declarations are listed before artifact declarations on that module