Directive APIs and ‘&’

Step 1: Define Method In Controller

‘this’ refers to parent controller instance

```
function Controller() {
  this.method = function (arg1) {
    this.prop = "Hi " + arg1;
  };
}
```

‘arg1’ needs to come from child directive

Step 2: Declare Method Reference in Directive

```
function MyDirective() {
  var ddo = {
    scope: {
      myMethod: '&method'
    },
    ...
    templateUrl: 'template.html'
  };
  return ddo;
}
```

Attribute name to use in parent template on this directive

Property name to reference parent method in directive

Step 3: Declare In Parent's Template

```
<div ng-controller="Controller as ctrl">
  <my-directive
      method="ctrl.method(myArg)">
  </my-directive>
</div>
```

Reference to method in controller

Placeholder label for value to be passed in from directive

Step 4: Map Method & Args in Directive's Template

```
<button
 ng-click="dirCtrl.myMethod({myArg:'v1'});">
    Remove Item
</button>
```

Method name from isolate scope mapping

Map of parent template declared arg name to value from directive

Step 4: Map Method & Args in Directive's Template

controller's template.html

```
<div ng-controller="Controller as ctrl">
  <my-directive
      method="ctrl.method(myArg)">
  </my-directive>
</div>
```

directive's template.html

```
<button
 ng-click="dirCtrl.myMethod({myArg:'v1'});">
   Remove Item
</button>
```

Result

controller's template.html

```
<div
 ng-controller="… as ctrl">
 <my-directive
   method="ctrl.method(myArg)">
 </my-directive>
</div>
```

DDO

```
function MyDirective() {
  var ddo = {
    scope: {
      myMethod: '&method'
    },
    ...
    templateUrl: 'template.html'
  };
  return ddo;
}
```

directive's template.html

```
<button
 ng-click="dirCtrl.myMethod({myArg:'v1'});">
   Remove Item
</button>
```

Result

DDO

```
function MyDirective() {
  var ddo = {
    scope: {
      myMethod: '&method'
    },
    ...
    templateUrl: 'template.html'
  };
  return ddo;
}
```

controller's template.html

```
<div
 ng-controller="… as ctrl">
 <my-directive
   method="ctrl.method(myArg)">
 </my-directive>
</div>
```

directive's template.html

```
<button
 ng-click="dirCtrl.myMethod({myArg:'v1'});">
   Remove Item
</button>
```

Result

DDO

```
function MyDirective() {
  var ddo = {
    scope: {
      myMethod: '&method'
    },
    ...
    templateUrl: 'template.html'
  };
  return ddo;
}
```

controller's template.html

```
<div
 ng-controller="… as ctrl">
 <my-directive
   method="ctrl.method(myArg)">
 </my-directive>
</div>
```

directive's template.html

```
<button
 ng-click="dirCtrl.myMethod({myArg:'v1'});">
   Remove Item
</button>
```

Result

DDO

```
function MyDirective() {
  var ddo = {
    scope: {
      myMethod: '&method'
    },
    ...
    templateUrl: 'template.html'
  };
  return ddo;
}
```

controller's template.html

```
<div
 ng-controller="… as ctrl">
 <my-directive
   method="ctrl.method(myArg)">
 </my-directive>
</div>
```

directive's template.html

```
<button
 ng-click="dirCtrl.myMethod({myArg:'v1'});">
   Remove Item
</button>
```

Summary

- ‘&’ binding allows us to execute an expression (such a function value) in the context of the parent scope
- Parent template must declare an attribute providing:
  - Method reference to call on the parent
  - Argument keys for directive to bind values to
- Directive:
  - Calls the referenced method
  - Provides a map of argument key to value pairs
  - Allows directive to pass data back to parent from isolate scope