AngularJS JumpStart

Modules, Controllers and Scope

AngularJS Architecture Patterns

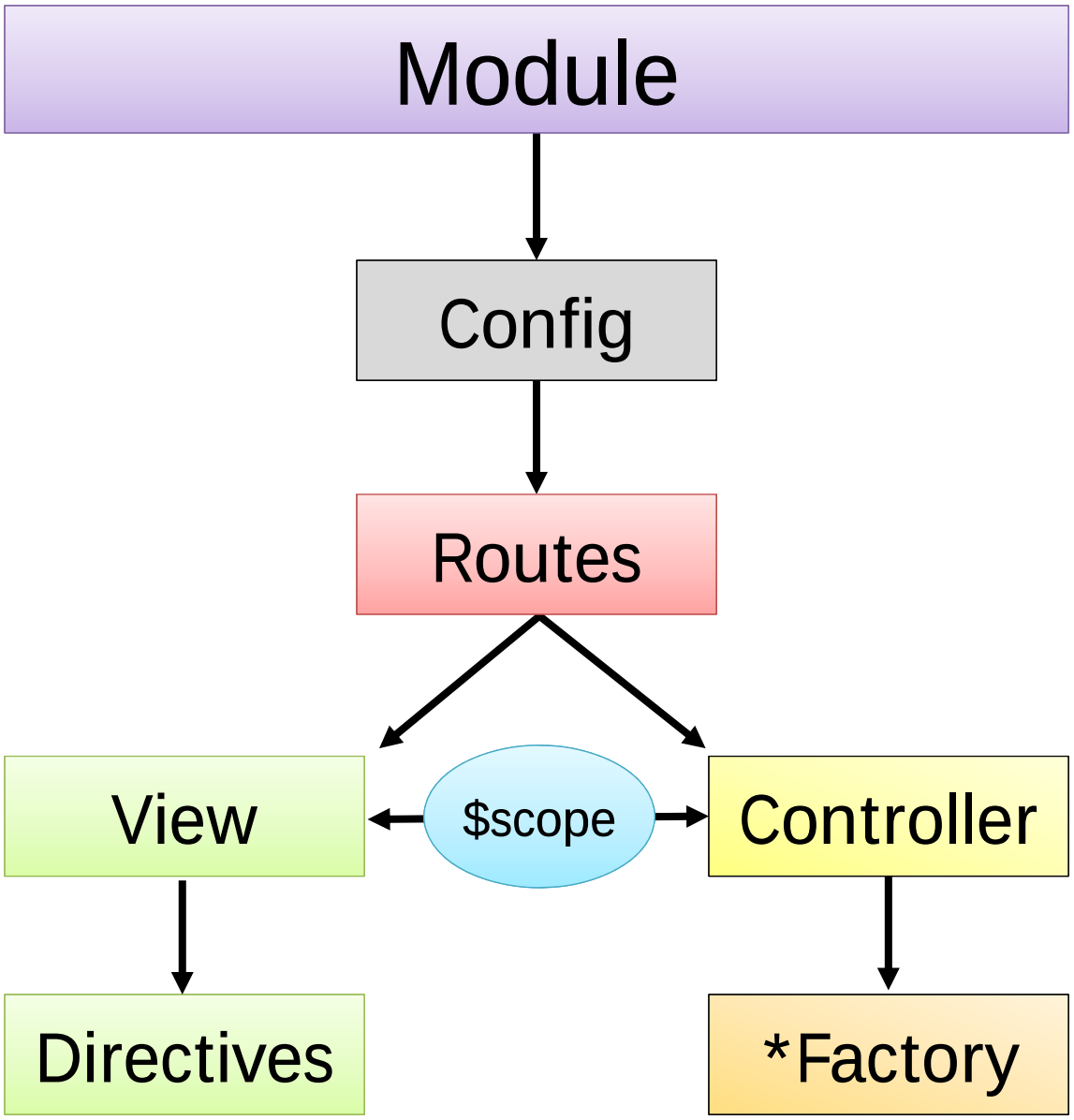
The Role of Controllers

The ng-controller Directive

The Role of Modules

Adding a Controller to a Module

The Big Picture



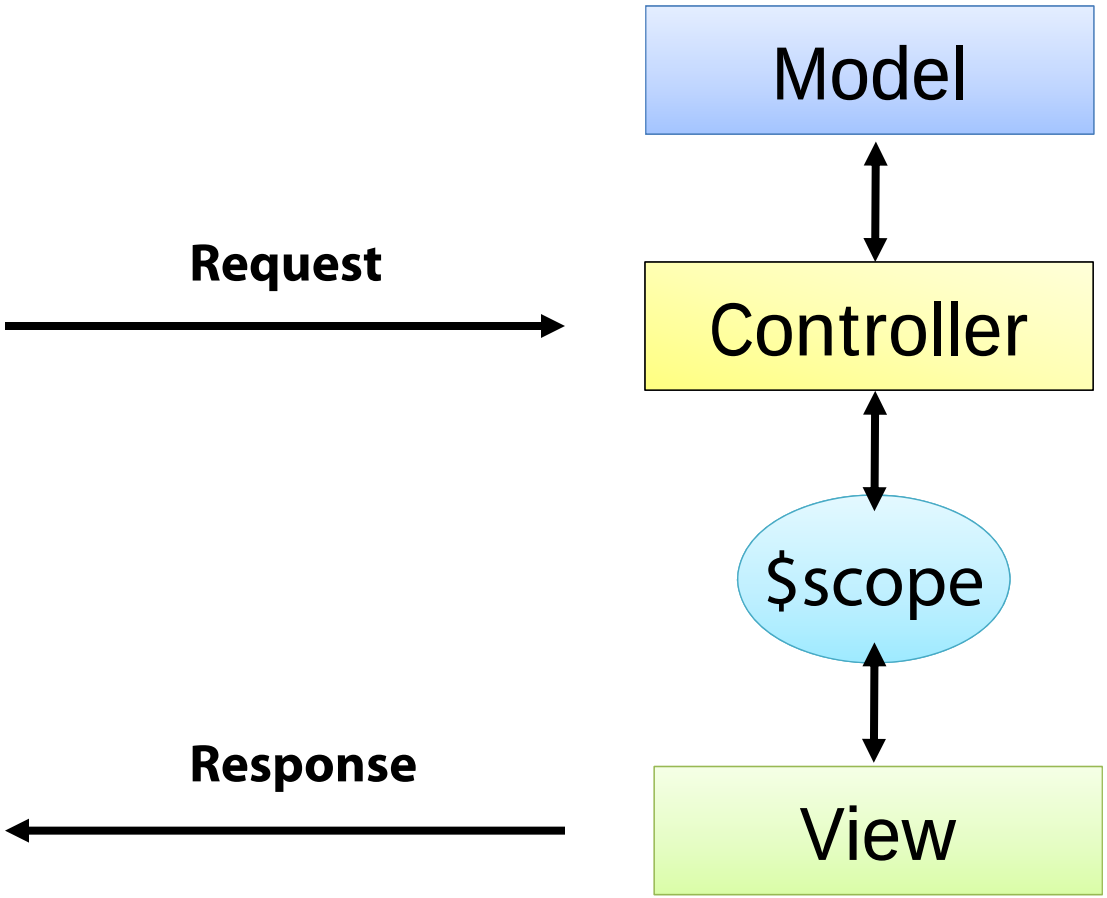
AngularJS Architecture Patterns

AngularJS Architecture Patterns

- **AngularJS relies on two key architecture patterns:**
  - Model-View-Controller (MVC)
  - Model-View-ViewModel (MVVM)

- **Patterns provide prescriptive guidance that can be used to build applications**

MVC + MVVM = MV*



The Role of Controllers

AngularJS Controllers

- **Controllers act as the "brain" for a view:**
  - Defines properties and methods
  - Handles showing/hiding controls and data in a view
  - Handles events triggered by a view
  - Knows how to retrieve data
  - Interacts with the view using the $scope object

The Role of $scope

- **$scope is "injected" into a controller**
- **Acts as the ViewModel**
- **Views bind to scope properties and functions**

Creating a Controller

$scope injected dynamically

```
<script>
    function SimpleController($scope) {
        $scope.customers = [
            { name: 'Danny Wallace', city: 'Phoenix' },
            { name: 'Jamie Riley', city: 'Atlanta' },
            { name: 'Heedy Thomas', city: 'Chandler' },
            { name: 'Jeff James', city: 'Seattle' }
        ];
    }
</script>
```

Note: Starting with AngularJS version 1.3.0 controller functions **must** be placed inside modules. You'll learn about modules later in this chapter and see how they can be used.

The ng-controller Directive

Tying a View and Controller Together

```
<div class="container" data-ng-controller="SimpleController">
    <h3>Adding a Simple Controller</h3>
    <ul>
        <li data-ng-repeat="cust in customers">
            {{ cust.name }} - {{ cust.city }}
        </li>
    </ul>
</div>

<script>
    function SimpleController($scope) {
        $scope.customers = [
            { name: 'Dave Jones', city: 'Phoenix' },
            { name: 'Jamie Riley', city: 'Atlanta' },
            { name: 'Heedy Wahlin', city: 'Chandler' },
            { name: 'Thomas Winter', city: 'Seattle' }
        ];
    }
</script>
```

Define the controller to use

Note: Starting with AngularJS version 1.3.0 controller functions **must** be placed inside modules. You'll learn about modules later in this chapter and see how they can be used.

Using "controller as"

```
<div class="container" data-ng-controller="SimpleController as ctrl">
    <h3>Adding a Simple Controller</h3>
    <ul>
        <li data-ng-repeat="cust in ctrl.customers">
           {{ cust.name }} - {{ cust.city }}
        </li>
    </ul>
</div>

<script>
    function SimpleController() {
        this.customers = [
           { name: 'Dave Jones', city: 'Phoenix' },
           { name: 'Jamie Riley', city: 'Atlanta' },
           { name: 'Heedy Wahlin', city: 'Chandler' },
           { name: 'Thomas Winter', city: 'Seattle' }
        ];
    }
</script>
```

Note: Starting with AngularJS version 1.3.0 controller functions **must** be placed inside modules. You'll learn about modules later in this chapter and see how they can be used.

The Role of Modules

What is a Module?

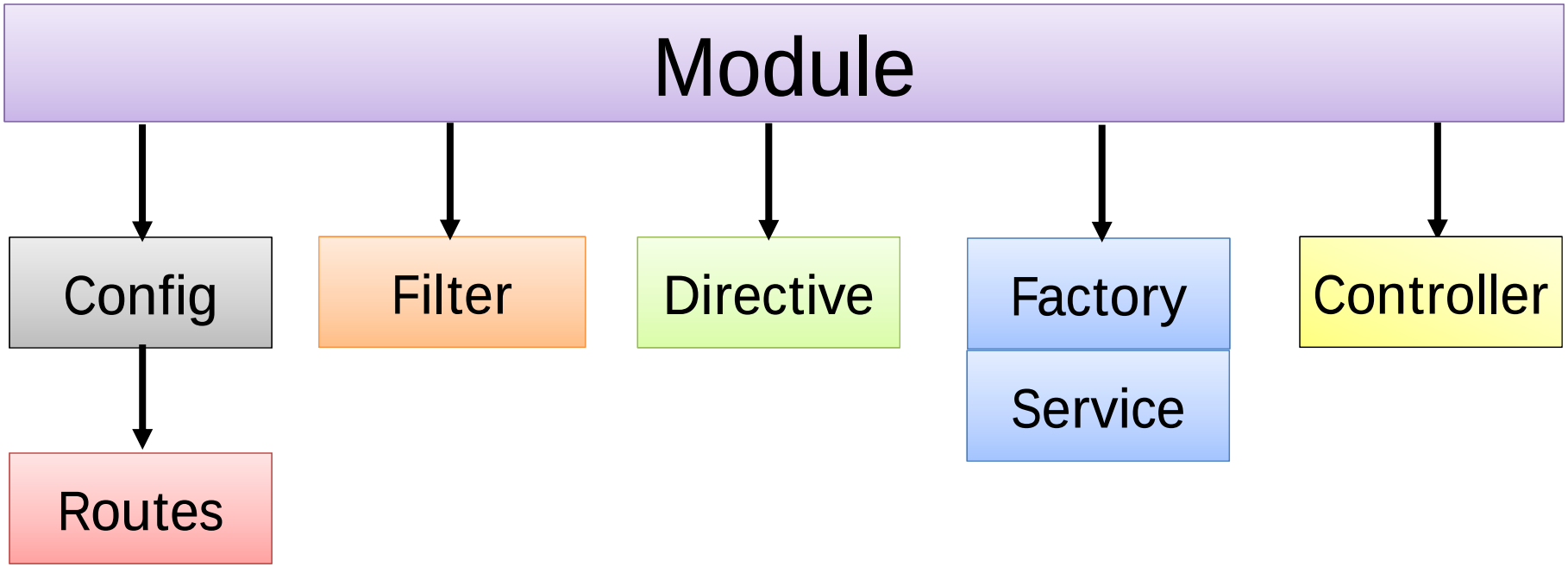
Module

Modules are containers for:

- Controllers
- Routes
- Factories/Services
- Directives
- Filters

Modules are Containers

```
<html ng-app="moduleName">
```



Defining a Module

- **Create a module using angular.module():**

```
var demoApp = angular.module('demoApp', []);
```

Dependencies

Injecting Dependencies into a Module

- Modules may rely on functionality from other modules
- Helper modules can be "injected" into a module:

```
var demoApp = angular.module('demoApp',
  ['helperModule']);
```

External module

Adding a Controller to a Module

Adding a Controller to a Module: Option 1

```
var demoApp = angular.module('demoApp', []);
```

Define a Module

Define a Controller

```
demoApp.controller('SimpleController', function ($scope) {
    $scope.customers = [
       { name: 'Dave Jones', city: 'Phoenix' },
       { name: 'Jamie Riley', city: 'Atlanta' },
       { name: 'Heedy Wahlin', city: 'Chandler' },
       { name: 'Thomas Winter', city: 'Seattle' }
    ];
});
```

Adding a Controller to a Module: Option 2

```
var demoApp = angular.module('demoApp', []);
```

Define a Module

Reference module

```
angular.module('demoApp').controller('SimpleController',
function ($scope) {
    $scope.customers = [
       { name: 'Dave Jones', city: 'Phoenix' },
       { name: 'Jamie Riley', city: 'Atlanta' },
       { name: 'Heedy Wahlin', city: 'Chandler' },
       { name: 'Thomas Winter', city: 'Seattle' }
    ];
});
```

Adding a Controller to a Module: Option 3

```
var demoApp = angular.module('demoApp', []);

...

(function() {

    var SimpleController = function ($scope) {
        $scope.customers = [
           { name: 'Dave Jones', city: 'Phoenix' },
           { name: 'Jamie Riley', city: 'Atlanta' }
        ];
    };

    angular.module('demoApp')
      .controller('SimpleController',SimpleController);

}());
```

Dealing with Minification

```
angular.module('demoApp')
  .controller('SimpleController', ['$scope', function ($scope) {
     $scope.customers = […];
  }]);
```

OR

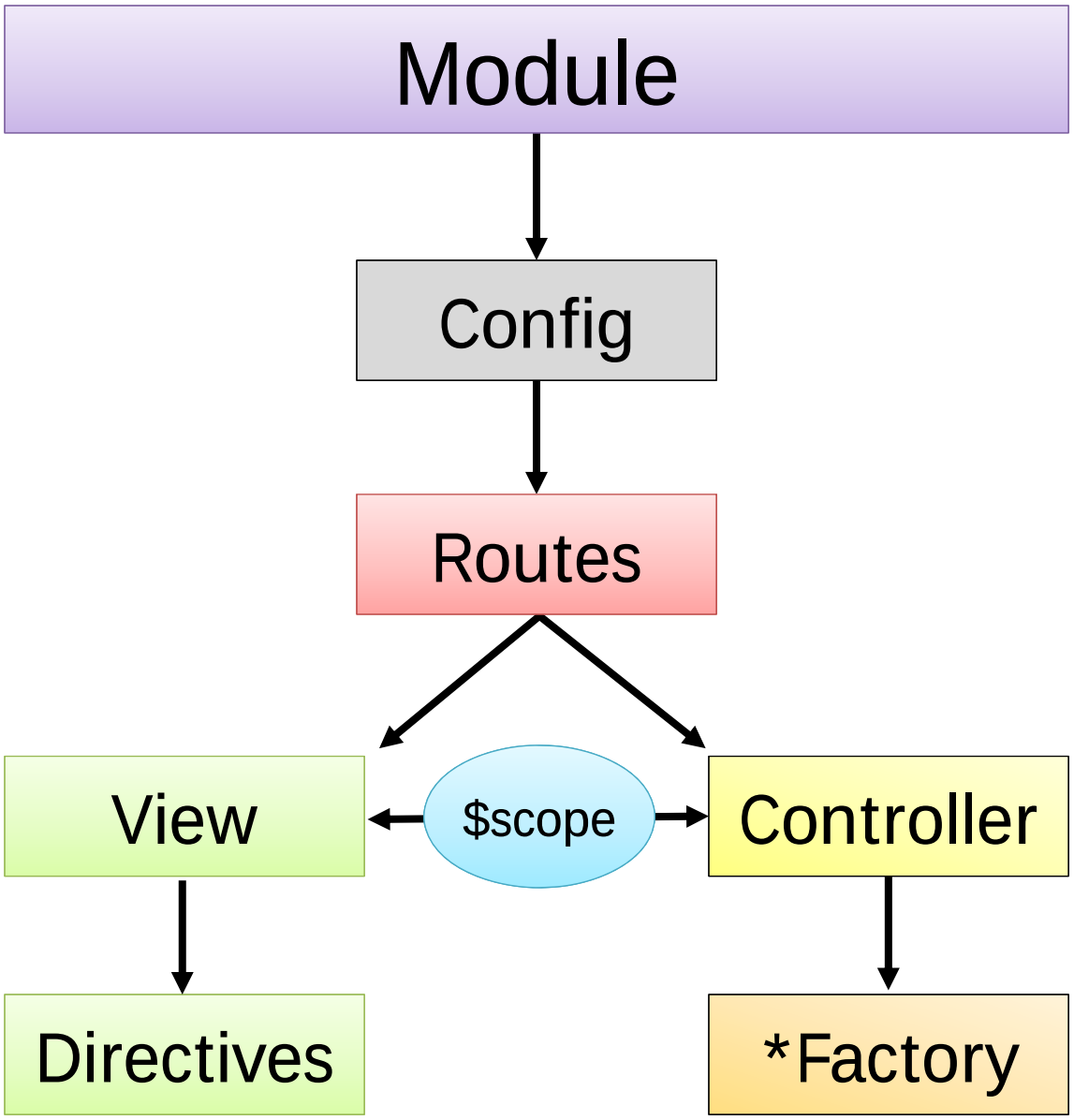
```
var SimpleController = function ($scope) {
    $scope.customers = […];
};

SimpleController.$inject = ['$scope'];

angular.module('demoApp')
  .controller('SimpleController', SimpleController);
```

Summary

The Big Picture



Summary

- **AngularJS relies on the MVC and MVVM architectural patterns**

- **Controllers act as the "brain" for a view:**
  - Know how to retrieve and store data
  - Interact with views using $scope

- **Modules act as containers for AngularJS applications**