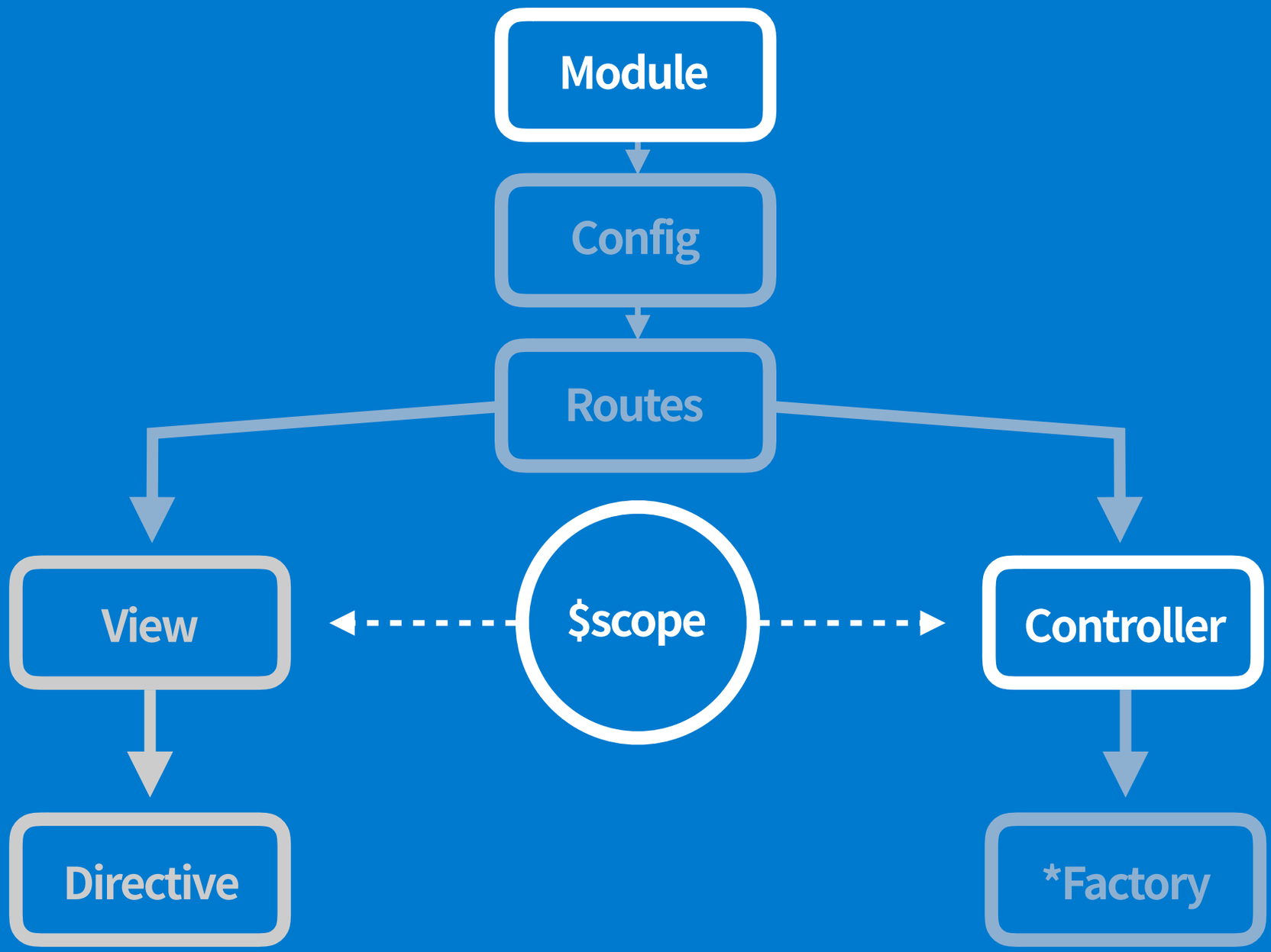
Controller / Scope / Module

Point

Module / Controller / Scope

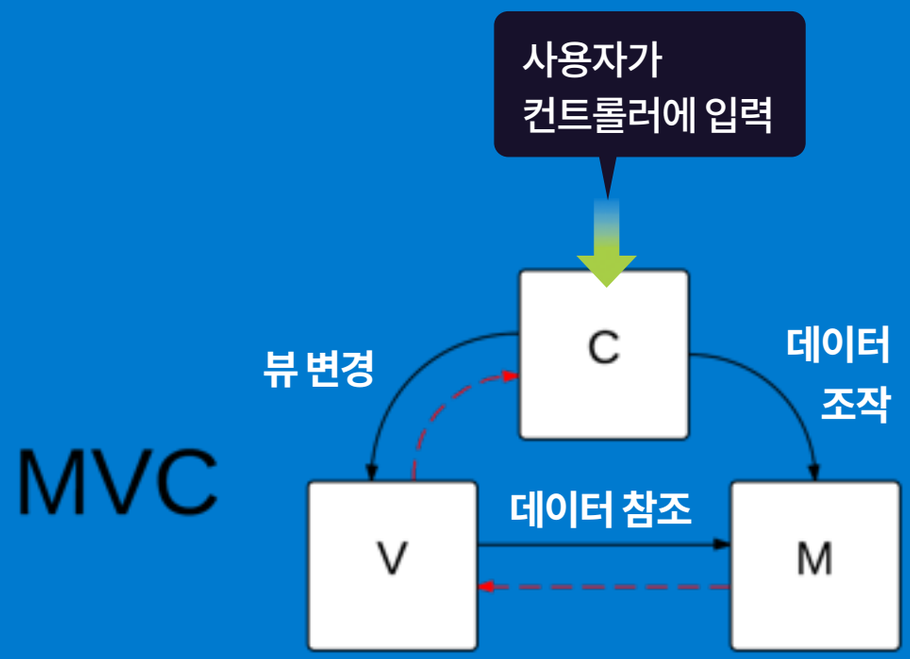


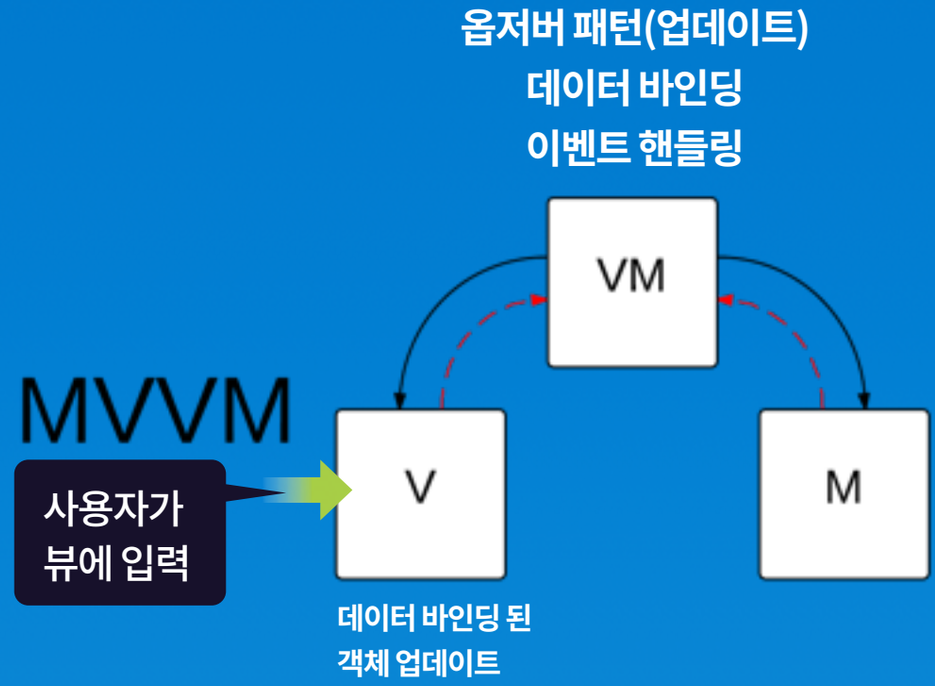
Architecture Patterns

Architecture Patterns

AngularJS는 2가지 디자인 패턴(M-V-C, M-V-VM)을 사용한다.

※ 패턴은 애플리케이션 제작 시 일관된 코딩 스타일을 약속하는 가이드일 뿐, 강제화 된 규칙은 아니지만 패턴을 사용함으로서
모듈화(Modularity), 유연성(Flexibility), 테스트 용이성(Testability), 유지보수(Maintainable) 를 향상시킬 수 있음.





직접 연결
간접 연결

Model 데이터/로직

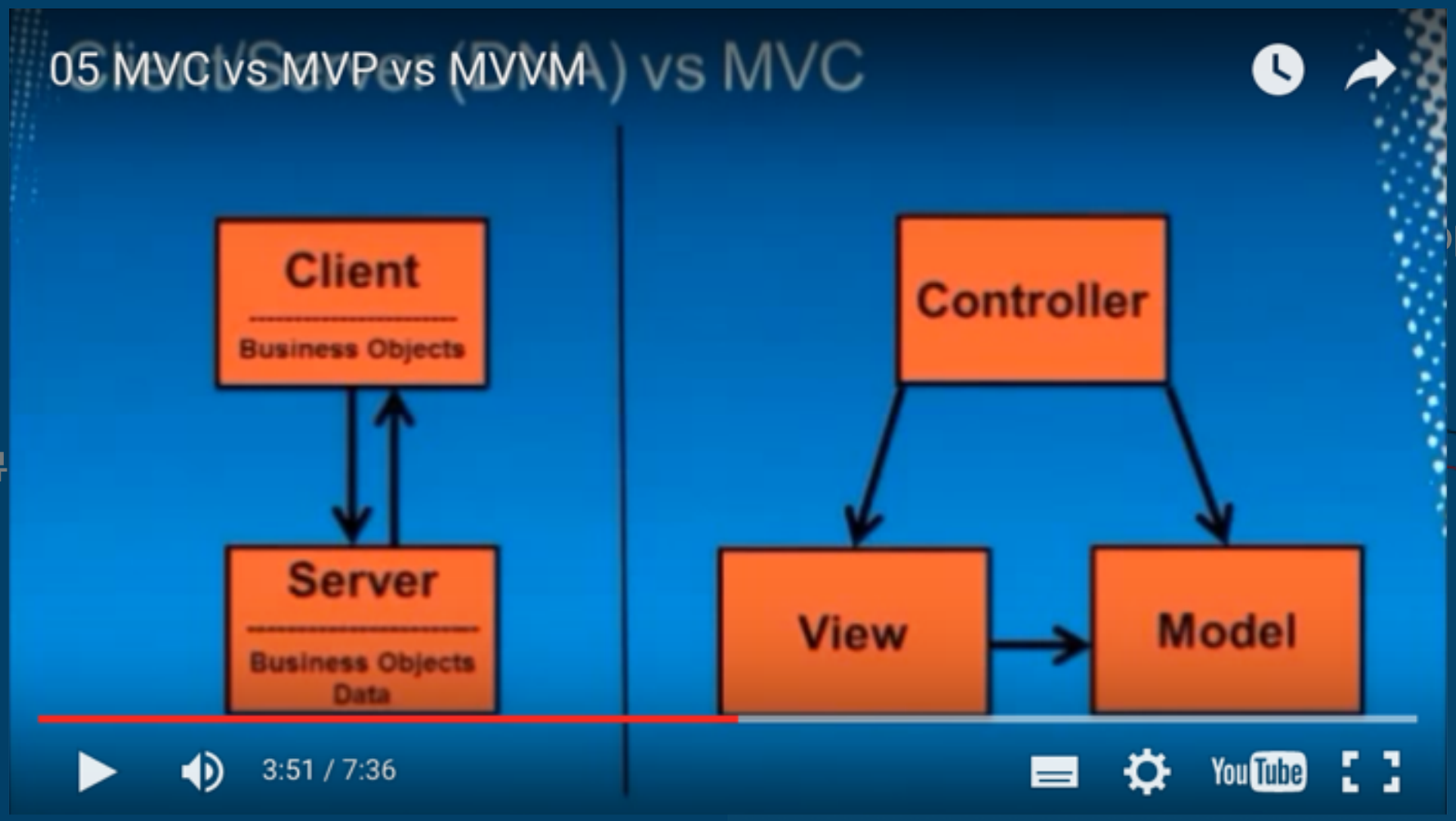
View GUI 레이아웃/컴포넌트

Controller/ViewModel Model/View를 연결하는 접착제

Architecture Patterns

AngularJS는 2가지 디자인 패턴(M-V-C, M-V-VM)을 사용한다.

※ 패턴은 애플리케이션 제작 시 일관된 코딩 스타일을 약속하는 가이드일 뿐, 강제화 된 규칙은 아니지만 패턴을 사용함으로서
모듈화(Modularity), 유연성(Flexibility), 테스트 용이성(Testability), 유지보수(Maintainable) 를 향상시킬 수 있음.



https://www.youtube.com/watch?v=ElHslw5wMWg

MVC

뷰

이트)

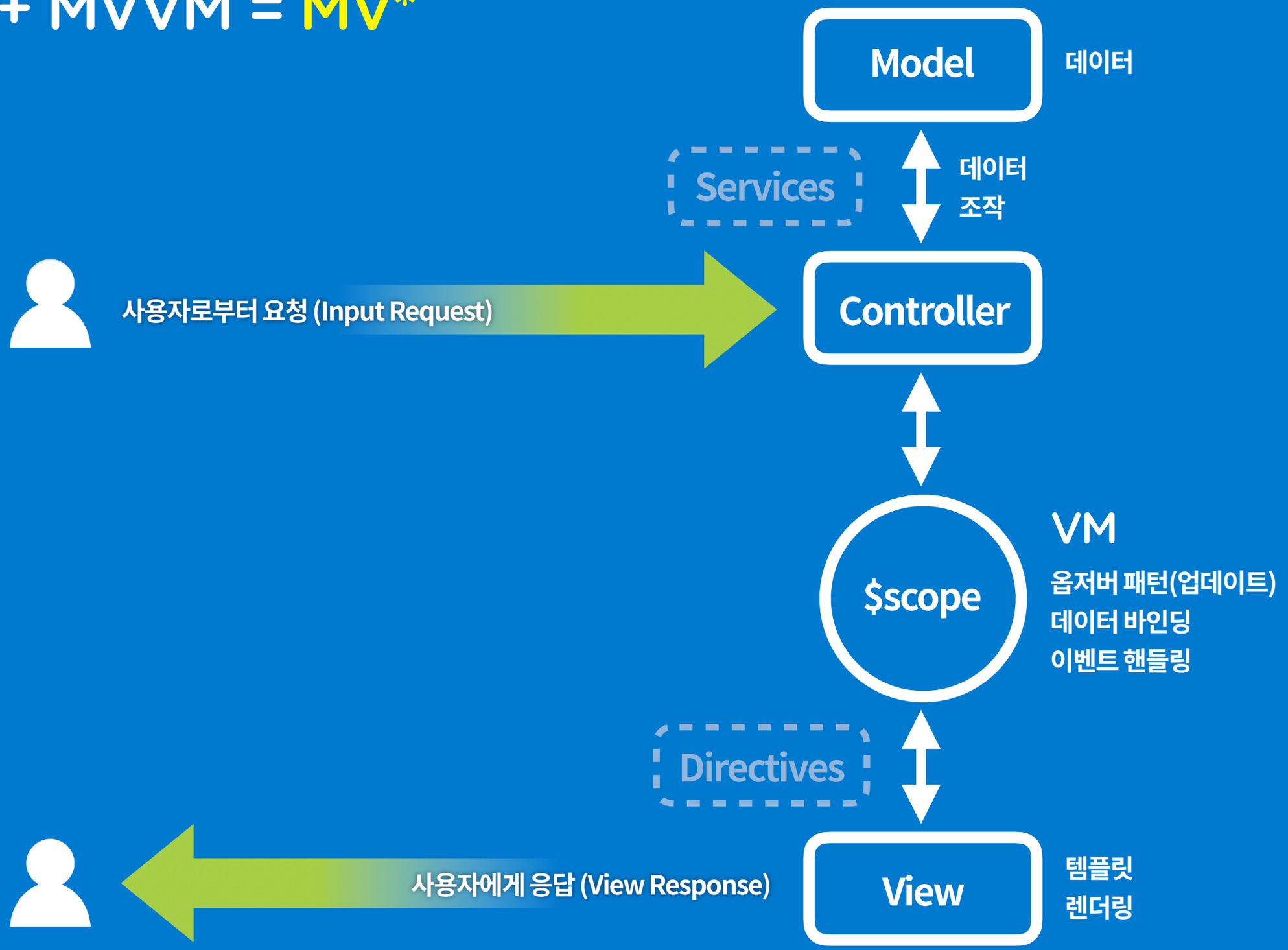
직접 연결

간접 연결

Model 데이터/로직 View GUI 레이아웃/컴포넌트 Controller/ViewModel Model/View를 연결하는 접착제

Architecture Patterns

MVC + MVVM = MV*



Controller

Role of Controllers

Controller

컨트롤러는 뷰(View)를 제어하는 브레인(Brain) 역할

- 속성/메소드 정의
- 뷰(View)에서 데이터/컨트롤을 감춤/보임 조작
- 뷰(View)로부터 이벤트 감지/작동 조작
- 데이터 조작(읽기/쓰기)
- $scope 객체(VM)를 사용하여 뷰(View)와 상호작용(Interaction)

$scope

$scope 객체는 View와 Controller를 연결하는 매개체(Mediator)

- 컨트롤러(Controller) 내부에 포함(주입, Injected)되어 접근 가능
- 뷰모델(ViewModel) 역할
- 뷰(View)는 $scope 객체의 속성/메소드에 연결

Define Controller

컨트롤러(Controller) 내부에 주입된 $scope 객체에 접근하여 속성/메소드를 설정할 수 있다.

$scope객체의 속성/메소드는 뷰(View)에서 접근 가능하다.

yamoo9_example.html

yamoo9_example.html

```
// AngularJS - 컨틀로러(Controller)
function myController( $scope ) {

    // $scope 객체 - 속성 설정
    $scope.data = [
        {
            'name'        : 'DOM',
            'description' : 'DOM = 문서객체모델'
        },
        {
            'name'        : 'Javascript',
            'description' : '자바스크립트'
        }
    ];

    // $scope 객체 - 메소드 설정
    $scope.saveData = function( data ) {
        $scope.data.push( data );
    }

}
```

컨트롤러에 동적으로
주입된 $scope 객체

함수를 컨트롤러로 바로 사용할 수 있는 것은
AngularJS 1.2.x 버전까지!

ngController - Directive

yamoo9_example.html

yamoo9_example.html

```
1  <div class="container" data-ng-controller="myController"> ... </div>

   <script>
   // AngularJS - 컨틀로러(Controller)
   function myController( $scope ) {
       // $scope 객체 - 속성 설정
       $scope.sortBy = 'name';
       $scope.reverse = false;

       $scope.categories=[
           {
               'name'        : 'DOM',
               'description' : 'DOM = 문서객체모델'
           },
           {
               'name'        : 'Javascript',
               'description' : '자바스크립트'
           },
           {
               'name'        : 'Node.js',
               'description' : '서버사이드 Javascript 환경'
           },
           {
               'name'        : 'AngularJS',
               'description' : 'Javascript SPA 프레임워크'
           }
       ];
       // $scope 객체 - 메소드 설정
       $scope.doSort = function(prop) {
           $scope.sortBy = prop;
           $scope.reverse = !$scope.reverse;
           if (prop === 'name') {
               $scope.name_reverse = !$scope.name_reverse;
           } else {
               $scope.desc_reverse = !$scope.desc_reverse;
           }
       }
   }
   </script>
```

뷰에 컨트롤러 연결

AngularJS 1.3.x 버전 이후부터는
이 방법을 사용할 수 없어요.

Controller with as

컨트롤러(Controller) 내부에서 $scope 객체를 사용하지 않고도 뷰(view)에 연결 가능한 방법은 this 참조를 사용하는 것이다. 단 컨트롤러 as 문법을 사용하여야 컨트롤러 this 객체에 접근 가능.

yamoo9_example.html

```
<script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.2.9/angular.min.js"></script>
</head>
<body>

<div class="container" data-ng-controller="myController as myCtrl">
    <h2 class="headline">AngularJS <code>ngRepeat</code> 디렉티브를 활용하여 데이터 반복 순환 처리</h2>
    <input type="search" data-ng-model="search">
    <ul class="ngRepeat-demo">
        <li data-ng-repeat="category in myCtrl.categories | filter:search | orderBy:'name'">
            <h4>{{category.name}}</h4>
            <p data-ng-bind="category.description"></p>
        </li>
    </ul>

    <hr>

    <table class="ngRepeat-demo">
        <caption class="a11y-hidden">강의 카테고리</caption>
        <tr>
```

```
// AngularJS - 컨틀로러(Controller)
function myController() {
    this.sortBy = 'name';
    this.reverse = false;
    this.name_reverse = false;
    this.desc_reverse = false;
    this.categories=[ ... ];
    this.doSort = function(prop) {
        this.sortBy = prop;
        this.reverse = !this.reverse;
        if (prop === 'name') {
            this.name_reverse = !this.name_reverse;
        } else {
            this.desc_reverse = !this.desc_reverse;
        }
    }
}
```

Module

Role of Modules

Module

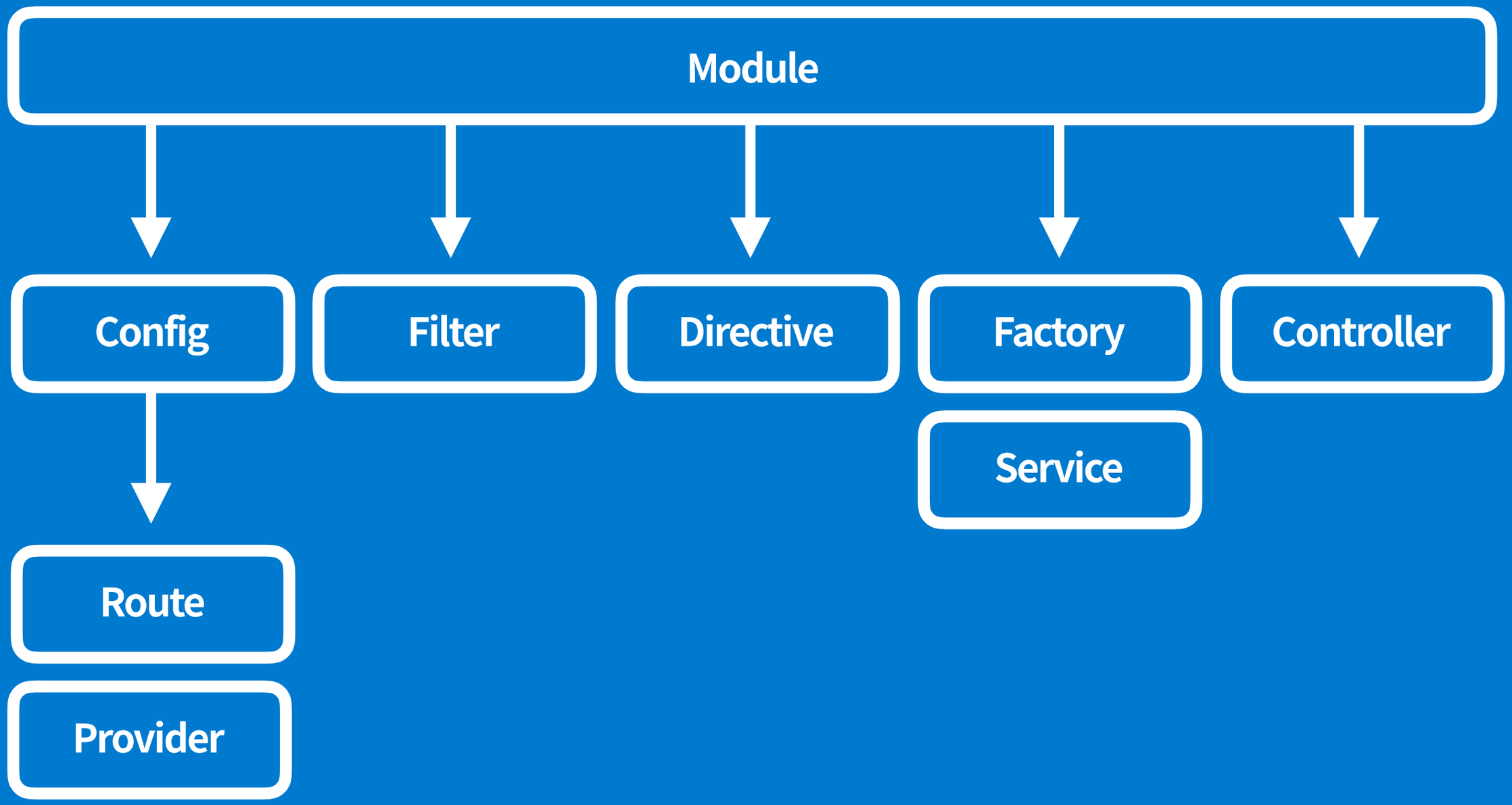
모듈(Module)은 컨테이너(Container)

- 컨트롤러(Controllers)
- 라우트(Routes)
- 팩토리(Factories)/서비스(Services)
- 디렉티브(Directives)
- 필터(Filters)

Module

모듈(Module)은 컨테이너(Container)

```
<html lang="ko-KR" data-ng-app="moduleName">
```



Create a Module

angular.module()을 사용하여 모듈 생성

생성된 모듈을 참조할 변수 설정

의존 모듈이 필요한 경우 [ ] 안에 포함 (Injecting Dependencies)

yamoo9_example.html

yamoo9_example.html

```
// AngularJS - 모듈 생성
var myApp = angular.module('myApp', []);


// AngularJS - 모듈 생성 (의존 모듈 포함)
var myApp = angular.module('myApp', ['modele1', 'module2']);
```

AngularJS 의존 모듈 추가

Adding a Controller - 1

컨트롤러(Contorller)를 모듈(Module)에 추가하는 첫 번째 방법은
정의된 모듈을 참조하는 변수를 통해 컨트롤러를 연결한다.

yamoo9_example.html

생성된 모듈 참조 변수

```
1
var myApp = angular.module('myApp', []);

myApp.controller('myController', function($scope) {...});
```

Adding a Controller - 2

컨트롤러(Contorller)를 모듈(Module)에 추가하는 두 번째 방법은
angular.module()에 모듈 이름을 전달한 모듈 참조에 컨트롤러를 연결한다.

yamoo9_example.html

yamoo9_example.html

```
1

angular.module('myApp', []);

angular.module('myApp')
        .myApp.controller('myController', function($scope) {...});
```

angular.module()을 통한 모듈 참조

Adding a Controller - 3

컨트롤러(Contorller)를 모듈(Module)에 추가하는 세 번째 방법은
정의된 함수를 컨트롤러에 연결한 후, 모듈 참조에 연결한다.

yamoo9_example.html

yamoo9_example.html

```
1

angular.module('myApp', []);

function myController($scope) {...}

angular.module('myApp').controller('myController', myController);
```

함수를 컨트롤러 정의에 연결한 후, 모듈 참조에 추가

Avoiding String Compress on Build Process

빌드(Build) 과정에서 코드를 최적화(압축)하는 과정에 모듈 이름을 줄여쓰게 되면 오류가 발생하게 된다. 이를 미연에 방지하기 위한 방법은 [ ]을 이용하여 모듈 이름을 줄여쓰지 못하도록 설정한다.

yamoo9_example.html

```
var myApp = angular.module('myApp', []);
myApp.controller('myController', ['$scope', function($scope) {...}]);
```

코드 압축 시, [] 안에 명시된 문자는 줄여 쓰지 않는다.

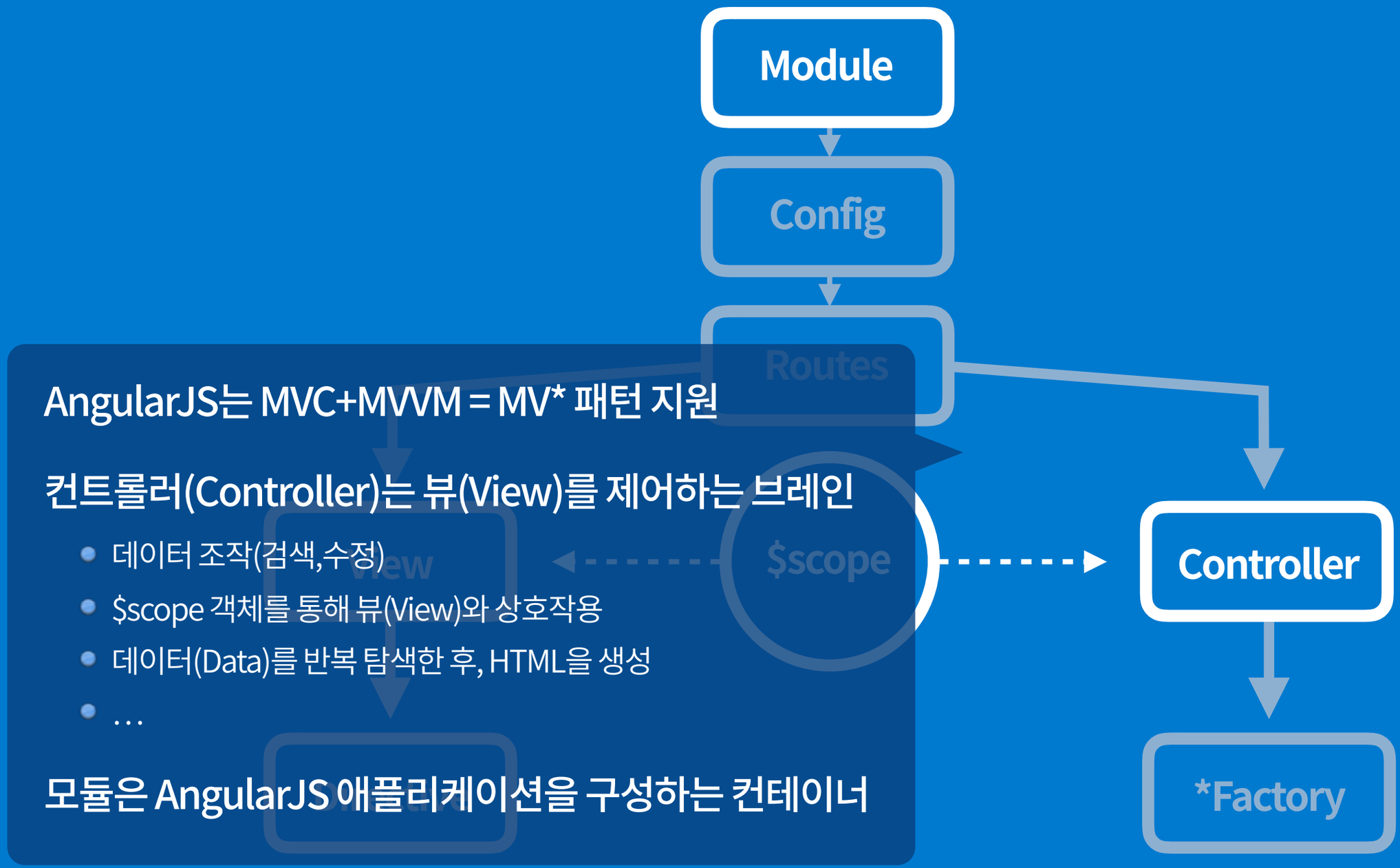
```
var myApp = angular.module('myApp', []);

(function(){
    function myController($scope) {...}
    angular.module('myApp').controller('myController', myController);
    myController.$inject = ['$scope'];
})();
```

코드 압축 시, [] 안에 명시된 문자는 줄여 쓰지 않는다.

Summary

Module / Controller / Scope



AngularJS는 MVC+MVVM = MV* 패턴 지원

컨트롤러(Controller)는 뷰(View)를 제어하는 브레인

- 데이터 조작(검색,수정)
- $scope 객체를 통해 뷰(View)와 상호작용
- 데이터(Data)를 반복 탐색한 후, HTML을 생성
- …

모듈은 AngularJS 애플리케이션을 구성하는 컨테이너