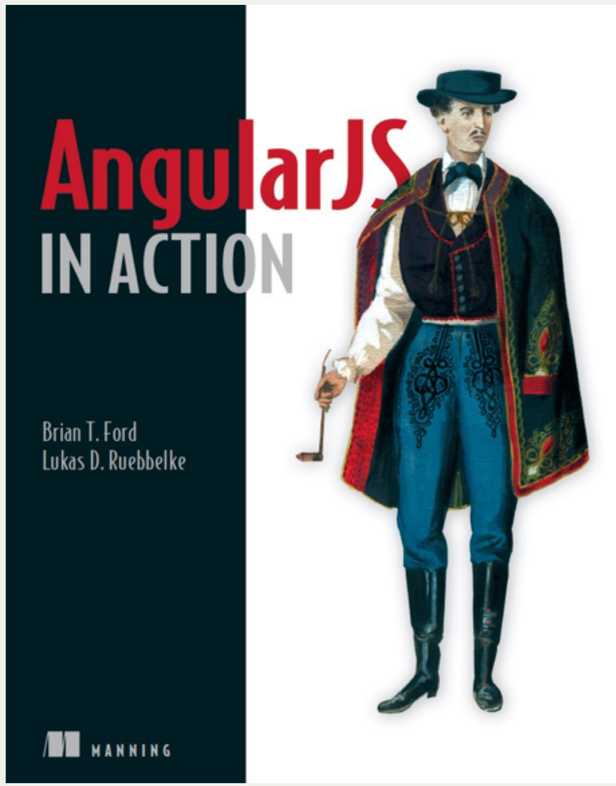
ANGULARJS IN DEPTH


AngularJS
IN ACTION
Brian T. Ford
Lukas D. Ruebbelke
MANNING

LUKAS RUEBBELKE

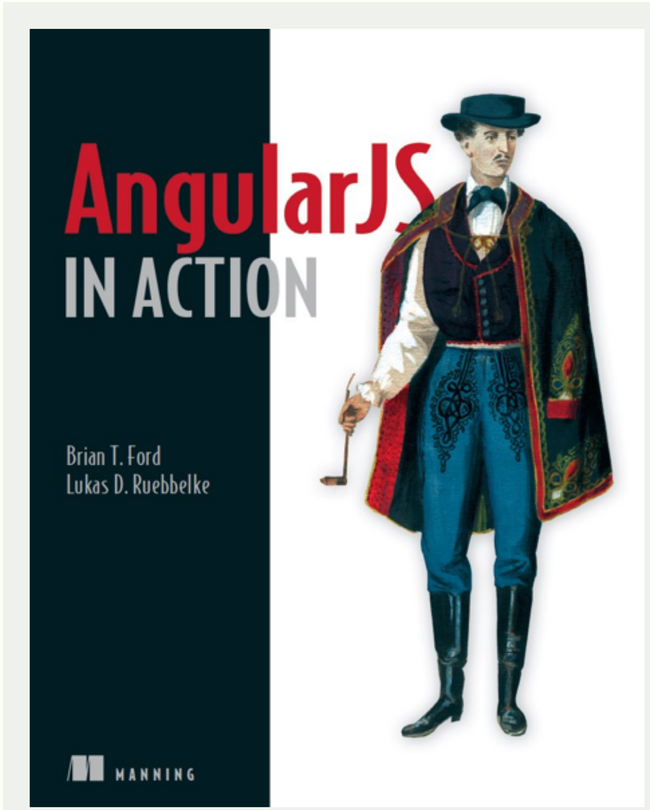
Co-author of **AnglarJS in Action**

Blog at **http://onehungrymind.com/**

Run the **Phoenix Web Application User Group**

Email me at **simpul@gmail.com**

Twitter me at **@simpulton**



THE SCHEDULE

- Hello **AngularJS**
- Build a Strong **AngularJS** Foundation
- **AngularJS** Directive Basics
- Server Side Integration with **AngularJS**
- Testing with **AngularJS**
- Advanced **AngularJS** Directives
- **AngularJS** Animations

HELLO ANGULARJS

- History of **AngularJS**
- **AngularJS** Elevator Pitch
- Hello **AngularJS** From Scratch
- Hello **AngularJS** with Yeoman

THE HISTORY OF ANGULARJS

Developed in **2009** by **Misko Hevery**.

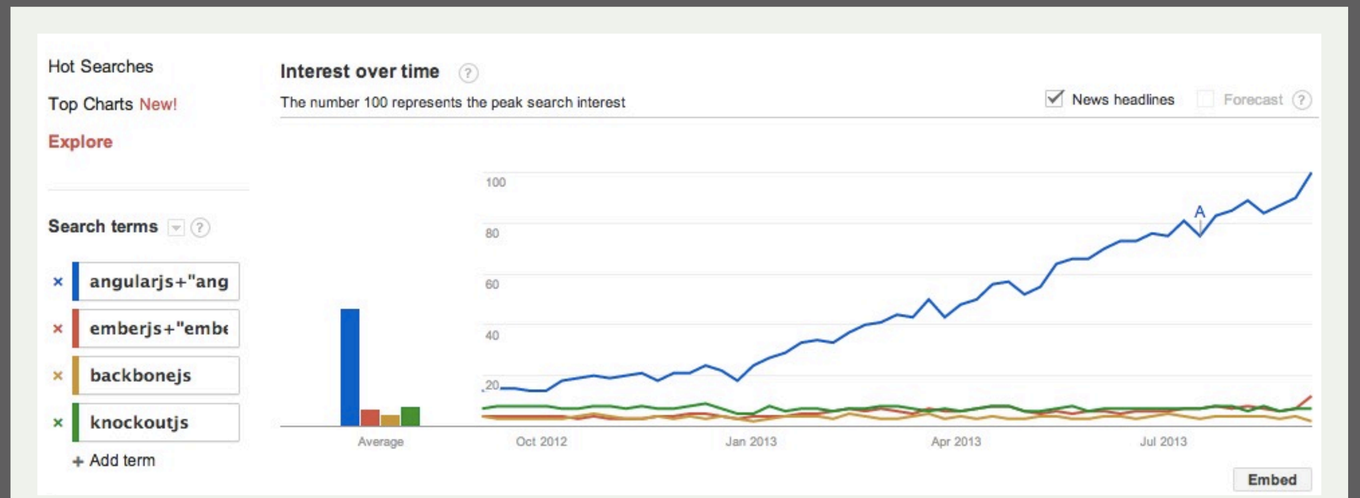
Publicly released as version 0.9.0 in **October 2010**

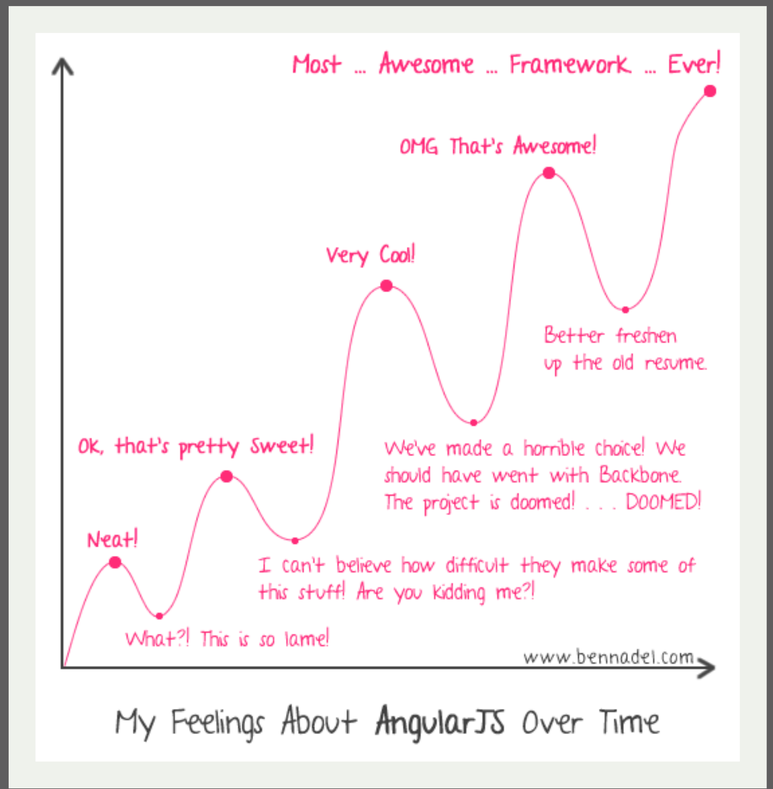
Currently at version 1.0.7 stable or 1.2.0rc1 unstable

THE ANGULARJS ELEVATOR PITCH

An intuitive framework that makes it **easy** to organize your code.
Testable code makes it **easier** to sleep at night.
Two-way data binding **saves** you **hundreds** of lines of code.
Templates that are HTML means you **already** know how to write them.
Data structures that are just JavaScript make integration **really easy**.

99 Problems but scope ain't one of them!


Hot Searches
Top Charts New!
Explore
Search terms
angularjs+"ang
emberjs+"embe
backbonejs
knockoutjs
+ Add term
Interest over time
The number 100 represents the peak search interest
News headlines
Forecast
100
80
60
40
20
A
Average
Oct 2012
Jan 2013
Apr 2013
Jul 2013
Embed



My Feelings About **AngularJS** Over Time

Friday, September 6, 13

BY HAND

```
// script.js
angular.module('MyApp', [])
.controller('MainCtrl', function($scope) {
  $scope.world = 'Front End Masters';
});

<!-- index.html -->
<html ng-app="MyApp">

<body ng-controller="MainCtrl">
  <h1>Hello {{world}}</h1>
</body>
```

YO

GRUNT

BOWER

Friday, September 6, 13

WITH YEOMAN

npm install -g yo
npm install -g generator-angular
mkdir **noterious** && cd $_
yo angular **noterious**
grunt server
grunt test
grunt

Friday, September 6, 13

THE 80/20 RULE

The **majority** of the work you are going to do with **AngularJS** will fall within a **core** set of **AngularJS** components.

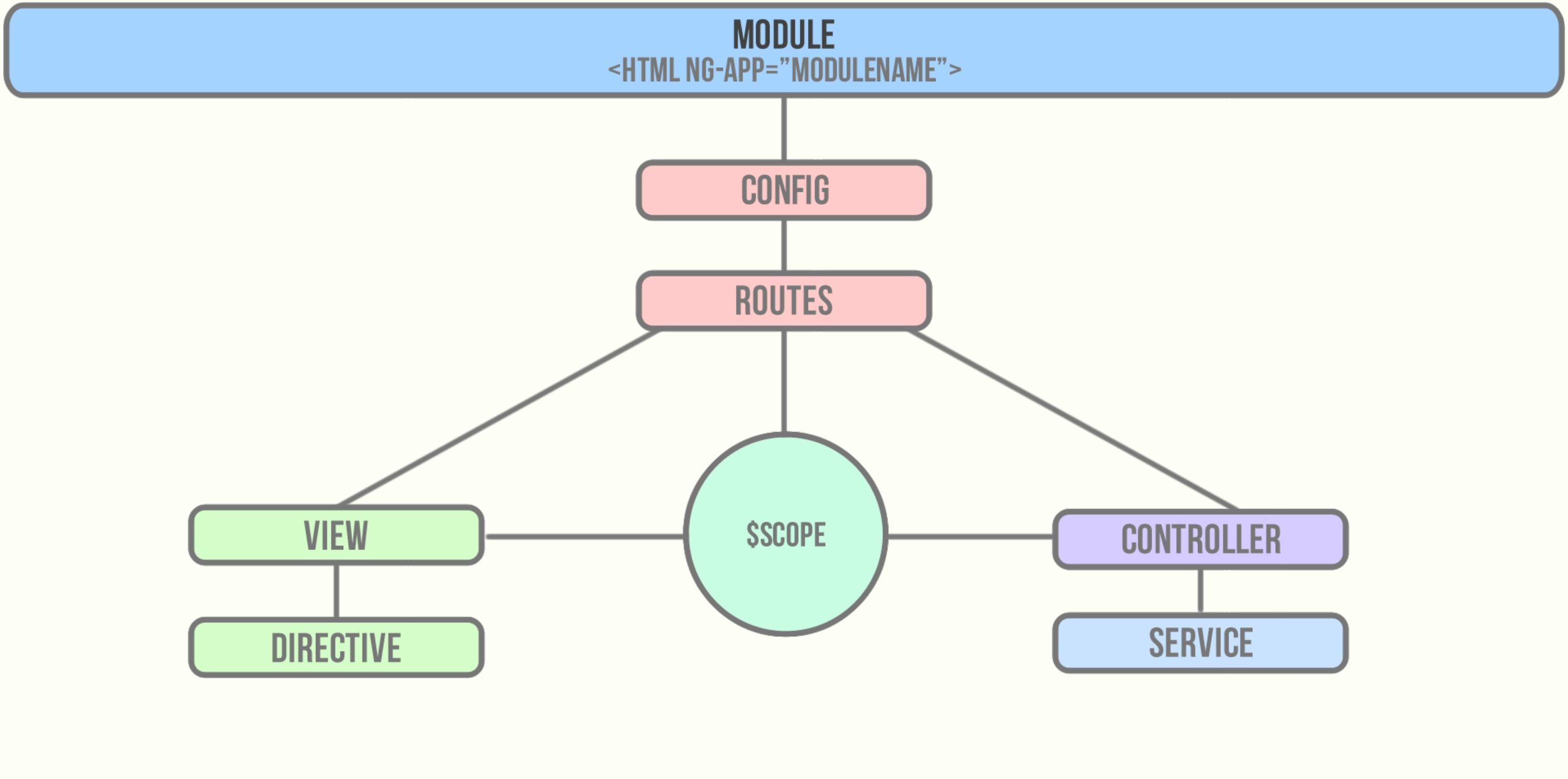
BUILD A STRONG ANGULARJS FOUNDATION

- The **AngularJS** players
- $compile
- $digest and $apply
- Model View Whatever
- Controller and $Scope
- View and Templates
- Models and Services
- Routes

THE MAJOR ANGULARJS PLAYERS

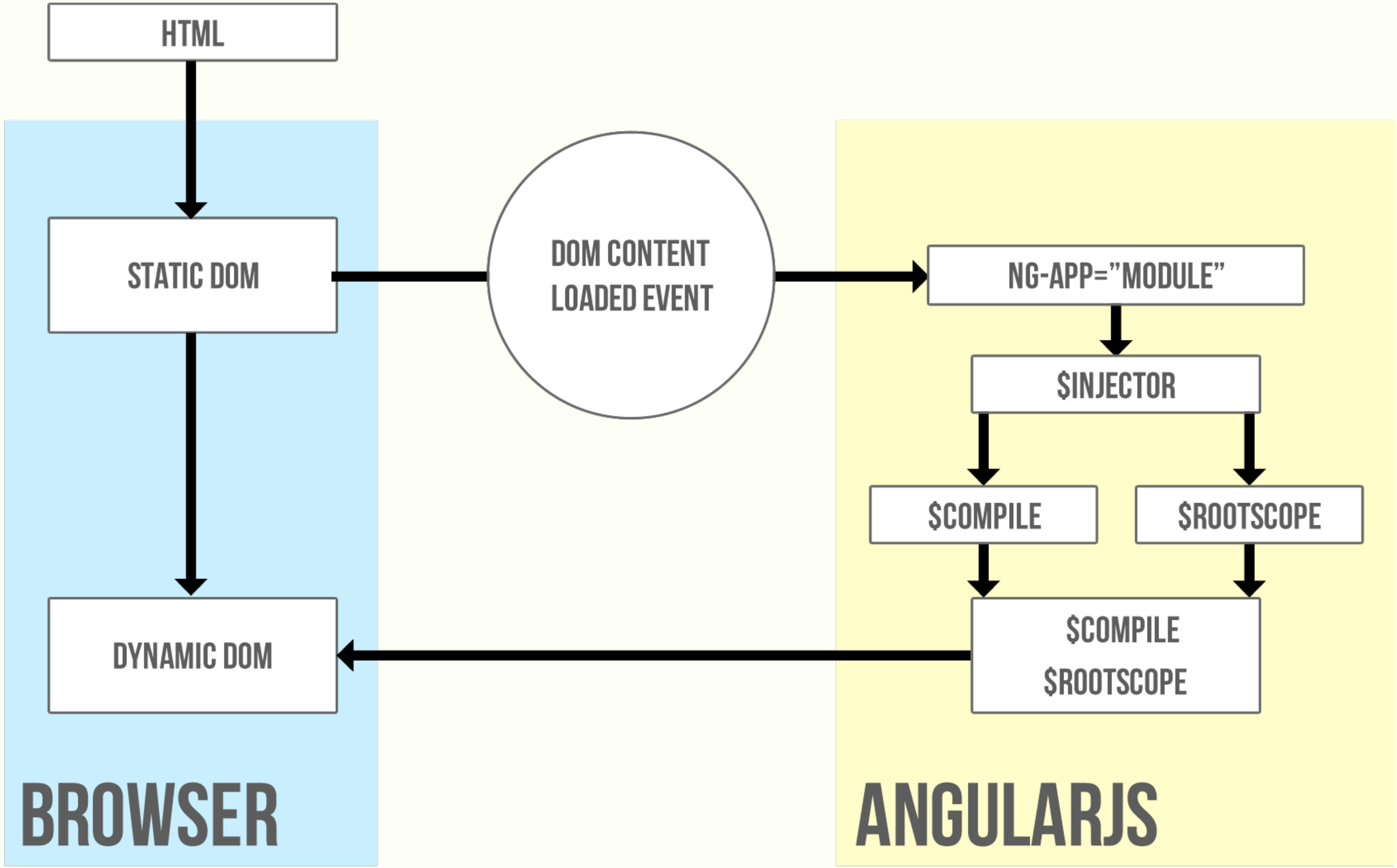
Who are the **major** players in AngularJS?

How do they **fit** together?


MODULE
<HTML NG-APP=”MODULENAME”>
CONFIG
ROUTES
VIEW
$SCOPE
CONTROLLER
DIRECTIVE
SERVICE

$COMPILE

$compile compiles DOM into a template function that can then be used to link scope and the View together

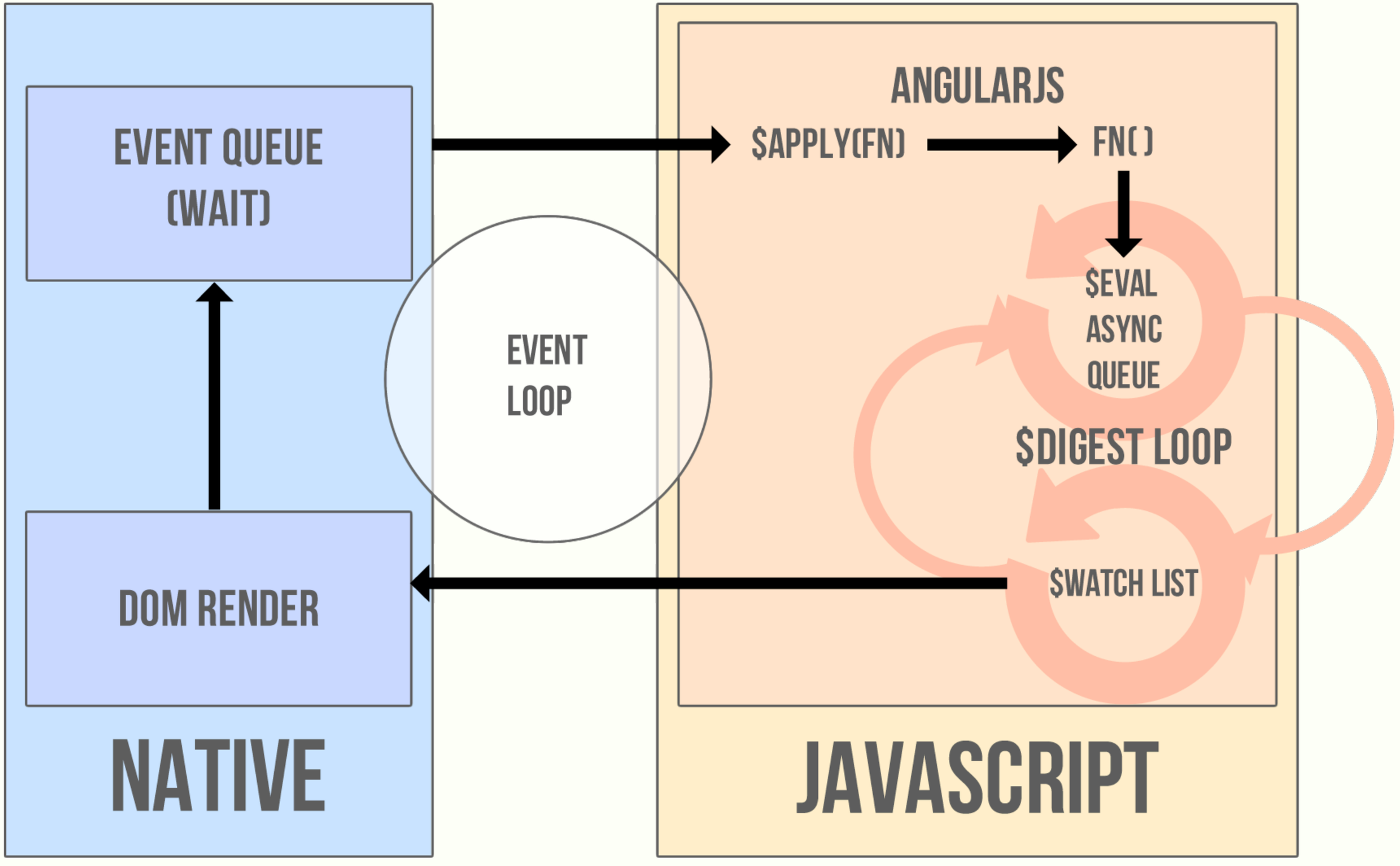

HTML
STATIC DOM
DOM CONTENT LOADED EVENT
NG-APP=”MODULE”
$INJECTOR
$COMPILE
$ROOTSCOPE
$COMPILE
$ROOTSCOPE
DYNAMIC DOM
BROWSER
ANGULARJS

$DIGEST AND $APPLY

$digest processes all of the watchers of the current scope

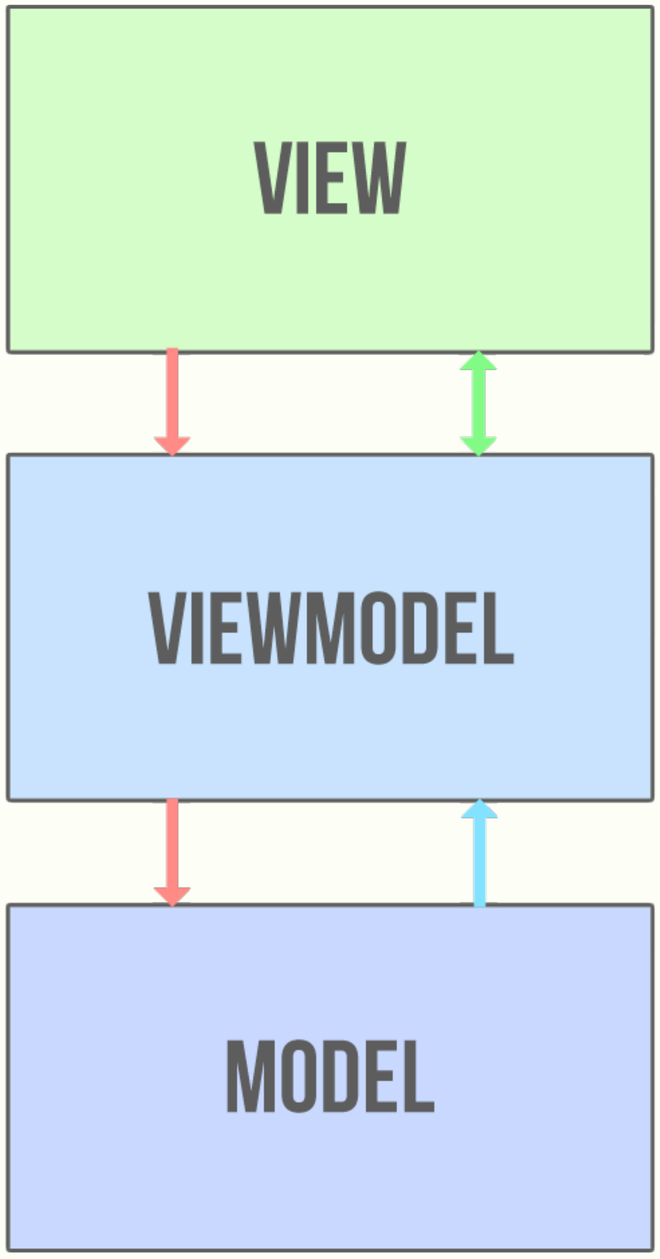
$apply() is used to notify that something has happened outside of the AngularJS domain

$apply forces a **$digest** cycle


EVENT QUEUE
(WAIT)
EVENT
LOOP
DOM RENDER
NATIVE
ANGULARJS
$APPLY(FN)
FN( )
$EVAL
ASYNC
QUEUE
$DIGEST LOOP
$WATCH LIST
JAVASCRIPT

MVWHATEVER

Choose whichever pattern helps you be more productive



MVVM

COMMANDS

DATA BINDINGS

CHANGE NOTIFICATION

CONTROLLER AND $SCOPE

$scope is the **glue** between the Controller and the View

The Controller is responsible for constructing the model on **$scope** and providing commands for the View to act upon

$scope provides context

CONTROLLER

IMPERATIVE BEHAVIOR

SCOPE

GLUE

VIEW (DOM)

DECLARATIVE VIEW

Friday, September 6, 13

BEST PRACTICES

Controllers should...

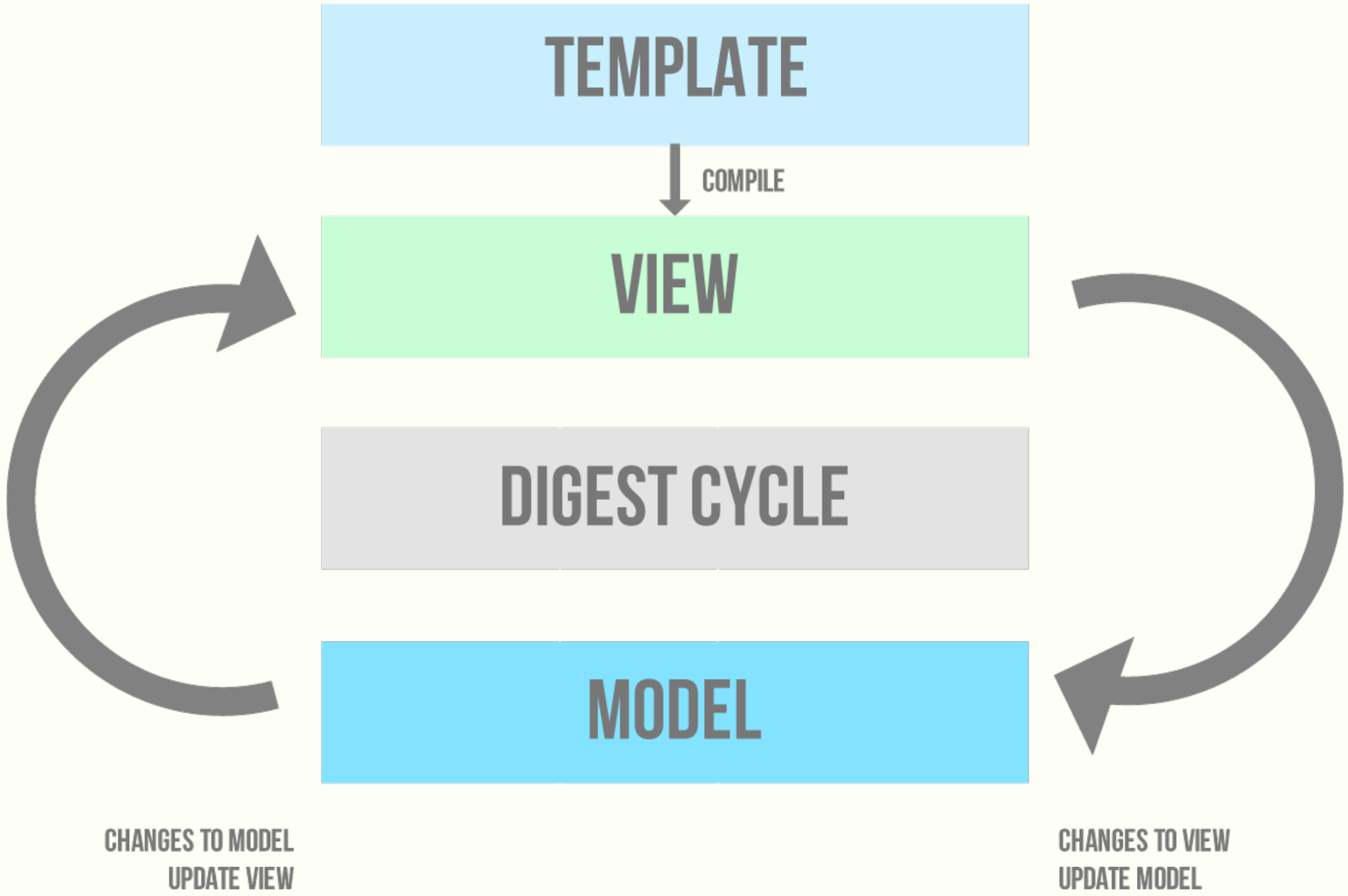
Not know Anything about the view they control
be small and focused
Should not talk to other controllers
Should NOT own the domain model

VIEW AND TEMPLATE

The View is AngularJS compiled DOM

The View is the product of $compile merging the HTML template with $scope

DOM is no longer the single source of truth.


TEMPLATE
COMPILE
VIEW
DIGEST CYCLE
MODEL
CHANGES TO MODEL
UPDATE VIEW
CHANGES TO VIEW
UPDATE MODEL

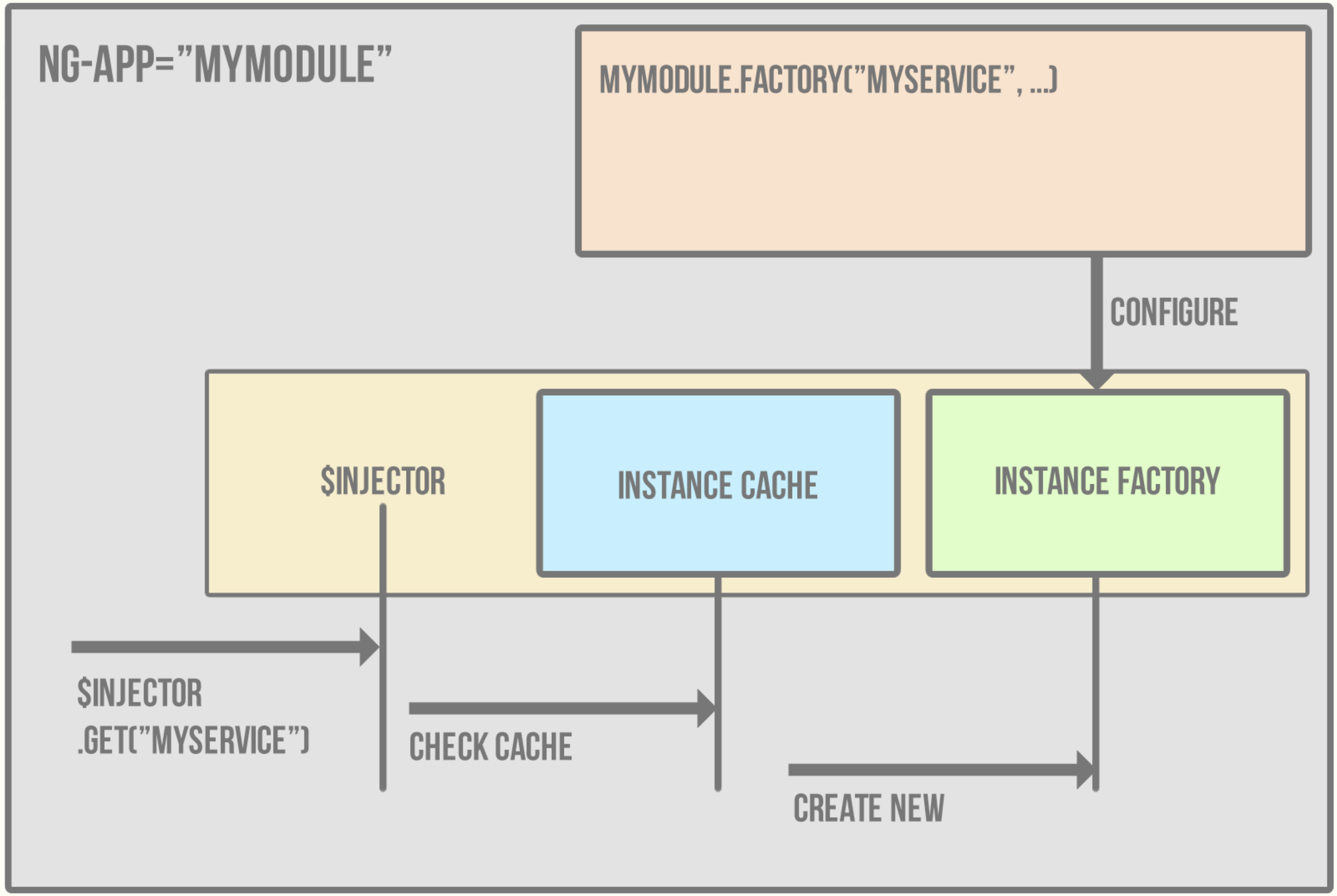
MODEL AND SERVICES

Services carry out common tasks specific to the web application

Services are consumed via the **AngularJS** Dependency Injection subsystem

Services are application singletons

Services are instantiated lazily


NG-APP=”MYMODULE”
MYMODULE.FACTORY(”MYSERVICE”, ...)
CONFIGURE
$INJECTOR
INSTANCE CACHE
INSTANCE FACTORY
$INJECTOR
.GET(”MYSERVICE”)
CHECK CACHE
CREATE NEW

ROUTES

$route is used for deep linking URLs to Controllers and Views

Define routes using **$routeProvider**

Typically used in conjunction with **ngView** directive and **$routeParams** service

ANGULARJS DIRECTIVE BASICS

- Directives as a DSL
- The Simplified World View
- Directive Definition Object
- The Controller Function
- The Link Function

WHY A DSL?

People find DSLs valuable because a well-designed DSL can be much easier to program with than a traditional library. This improves programmer productivity, which is always valuable. In particular it may also improve communication with domain experts, **which is an important tool for tackling one of the hardest problems in software development.**

Martin Fowler

FIXED LANGUAGES ARE BROKEN

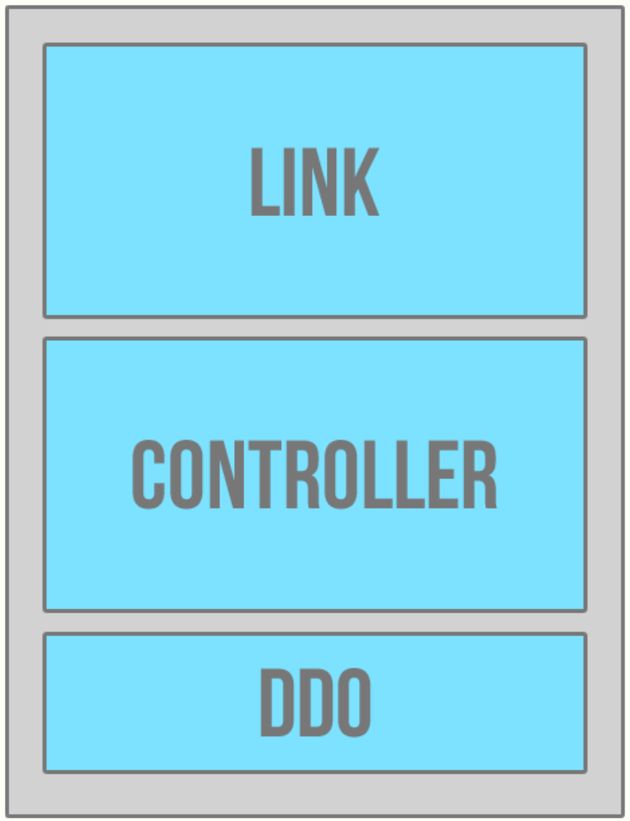
[Without the ability to reprogram the language] programmers resort to repetitive, error-prone workarounds. **Literally millions of lines of code have been written to work around missing features in programming languages.**

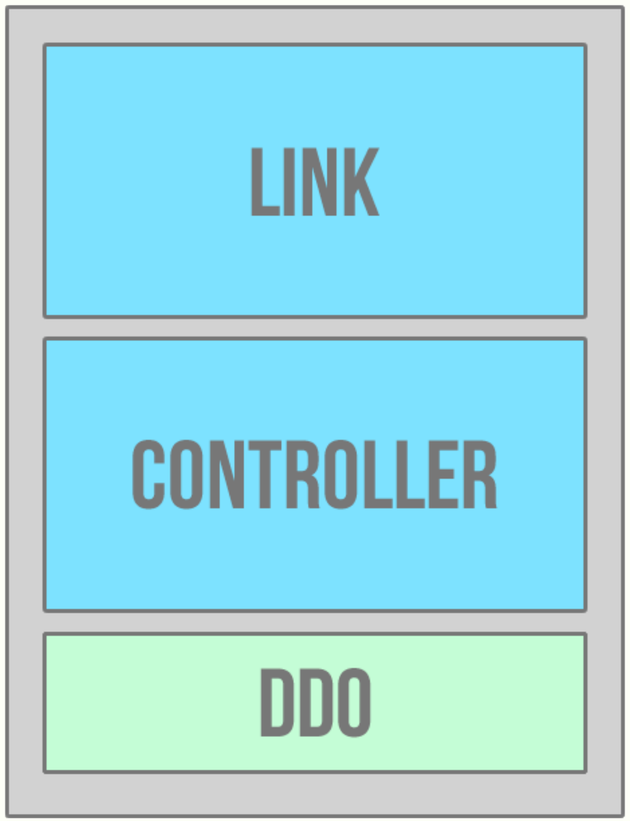
Stuart Halloway

THE SIMPLIFIED WORLD VIEW

Most simple directives consist of three things (or less)

Having a firm grasp on these three things will make advanced directives much more approachable

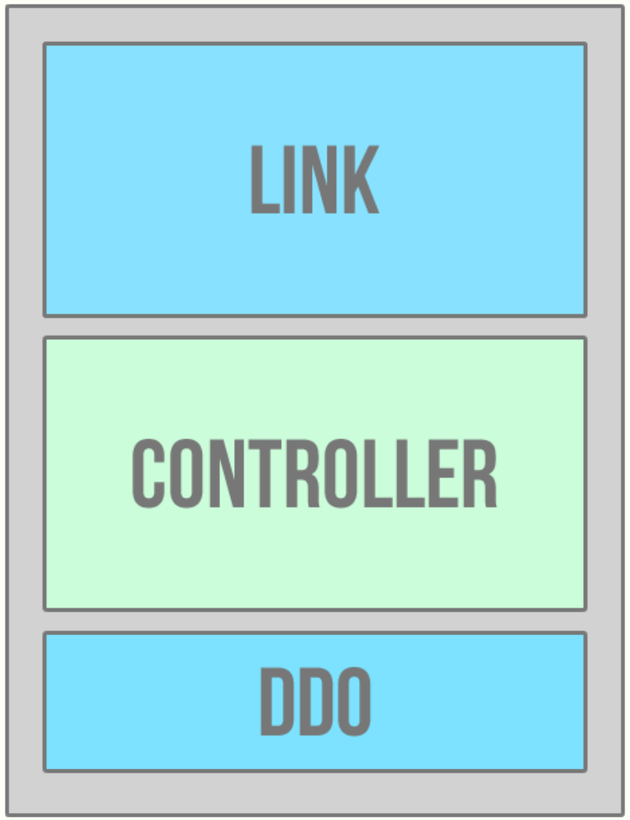

LINK
CONTROLLER
DDO


LINK
CONTROLLER
DDO

DIRECTIVE DEFINITION OBJECT

The Directive Definition Object (DDO) tells the compiler how a directive is supposed to be assembled

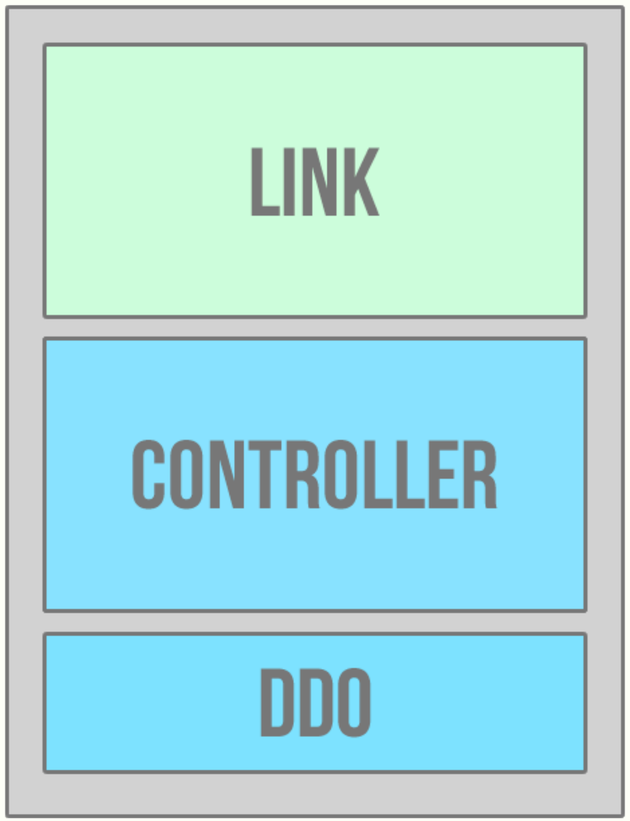
Common properties are the **link** function, **controller** function, **restrict**, **template** and **templateUrl**


LINK
CONTROLLER
DDO

THE CONTROLLER FUNCTION

The controller is constructed during the pre-linking phase

Receives $scope which is the current scope for the element


LINK
CONTROLLER
DDO

THE LINK FUNCTION

The link function is where DOM manipulation happens

The link function comes with **scope**, **element** and **attrs**

scope is the the same **$scope** in the controller function

element is a jQuery* instance of the DOM element the directive is declared on

attrs is a list of attributes declared on the element

SIMPLE DIRECTIVE

Boards directive.

SIMPLE DIRECTIVE WITH JQUERY

jQuery Transmit

SERVER SIDE INTEGRATION WITH ANGULARJS

- REST and $HTTP
- Promises
- Real Time Communication

$HTTP

The **$http** service is a core Angular service that facilitates communication with the remote HTTP servers via the browser's **XMLHttpRequest** object or via **JSONP**.

The **$http** API is based on the deferred/promise APIs exposed by the **$q** service

$HTTP SHORTCUT METHODS

$http.get

$http.head

$http.post

$http.put

$http.delete

$http.jsonp

PROMISES

The returned value of calling **$http** is a **promise**

You can use the **then** method to register callbacks

These callbacks receive a single argument - an object representing the response

REAL TIME COMMUNICATION

Firebase.

E2E TESTING WITH ANGULARJS

- The Environment
- Unit Testing
- E2E Testing

THE ENVIRONMENT

Karma is the current scenario runner

Karma is testing framework agnostic

The easiest way to get up and running with Karma is with Yeoman

UNIT TESTING

Test isolated pieces or units of logic

We accomplish this with matchers and spies

MATCHERS

Expectations are built with the function **expect** which takes a value, called the **actual**. It is chained with a **Matcher** function, which takes the expected value.

```
expect(a).toBe(b);
expect(a).not.toBe(null);
```

SPIES

Jasmine's test doubles are called **spies**. A spy can stub any function and tracks calls to it and all arguments. There are special matchers for interacting with spies.

```
spyOn(mockUserService, 'userExists').andCallThrough();

scope.userExists();
expect(mockUserService.userExists).toHaveBeenCalled();
```

E2E TESTING

Used for simulating user interactions

You will write **scenarios** to describe behavior based on interactions

Scenarios consist of **commands** and **expectations**

beforeEach runs before each **it** block

afterEach runs after each **it** block

ADVANCED ANGULARJS DIRECTIVES

- Isolated Scope
- Compile
- Transclusion
- 3rd Party Integration

ISOLATED SCOPE

Isolated scope does not prototypically inherit from its parent

This prevents your directive from accidentally modifying data in the parent scope

This in a sense defines the API for the directive

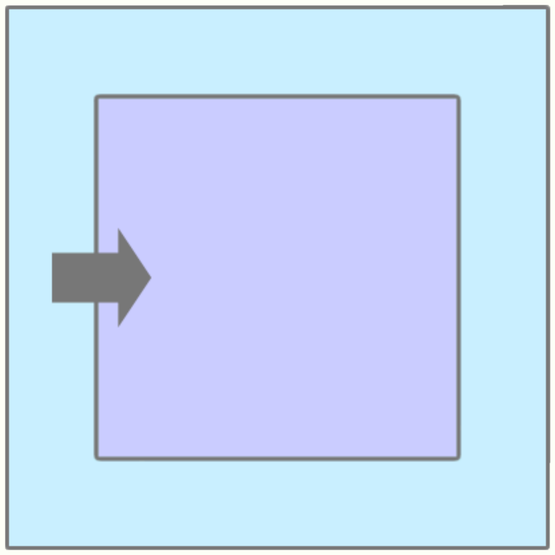
ATTRIBUTE ISOLATED SCOPE

Defined with an @ symbol

Binds a local scope property to the value of a DOM attribute

The binding is uni-directional from parent to directive

The result is always a string because DOM attributes are strings



“I’ll let you know when this has changed.”

“Cool!”

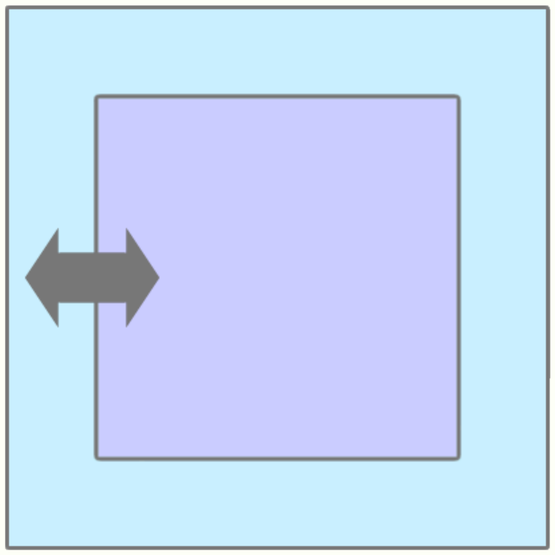
BINDING ISOLATED SCOPE

Defined with an = symbol

Bi-directional binding between parent and directive

You can define the binding as optional via =?

Optimal for dealing with objects and collections



“Let’s keep each other in the loop”

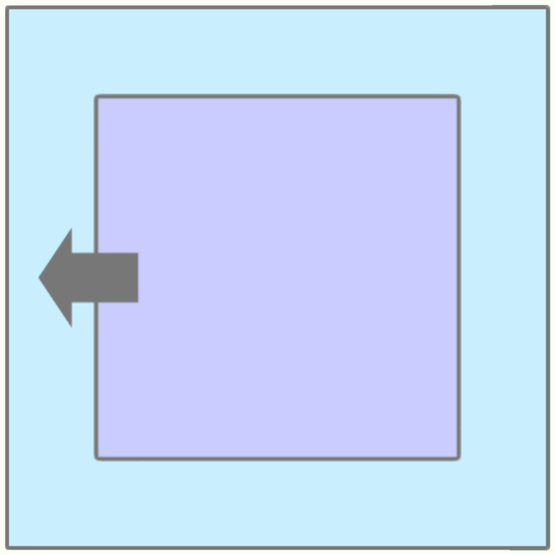
“Bromance”

EXPRESSION ISOLATED SCOPE

Defined using an & symbol

Allows you to execute an expression on the parent scope

To pass variables from child to parent expressions you must use an object map



“Let me know if you need me to do anything.”

“Go get me a juice box!”

TRANSCLUSION

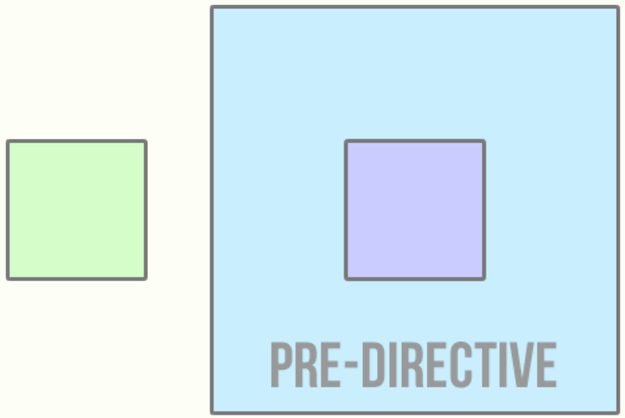
Compiles the contents of the element and makes it available to the directive

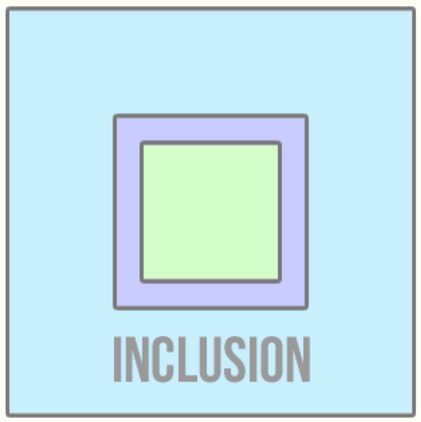
The transcluded contents are bound to the parent scope

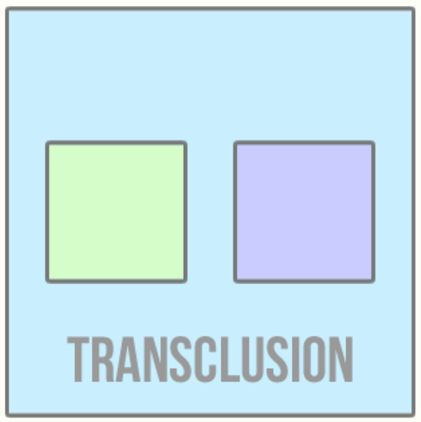
The directive contents are bound to an isolate scope

The two scopes are siblings

This is good for decorating an existing element with a directive without destroying it


PRE-DIRECTIVE


INCLUSION


TRANSCLUSION

COMPILE

Responsible for transforming the template DOM

MORE 3RD PARTY INTEGRATION

Content pending.

ANGULARJS ANIMATIONS

- The Animation Naming Convention
- CSS Transitions
- CSS Animations
- Javascript Animations

THE ANIMATION NAMING CONVENTION

The pattern is [class][event][destination] and stack on each other

.repeat-item.ng-enter.ng-enter-active

.my-elm.ng-hide-add.ng-hide-add-active

.my-animation.CLASS-add.CLASS-add-active

CSS TRANSITIONS

The **easiest** and **fastest** way to attach animations

Support by every browser except for IE9 and below

As long as the matching CSS class is present then AngularJS will pick up the animation

CSS ANIMATIONS

More extensive than transitions

Supported by every browser except IE9 and below

Does not require the destination class styling aka **-active** class

JAVASCRIPT ANIMATIONS

Allows for more control over your animations

Define your animations with the **animation** service

Naming convention is still class based

Make sure to call the **done()** function when the animation is over