



# Ground Up

Syed Awase



## 2.INTRODUCTION



# Why use **Angular**?

## What are the Key factors in building a line of business application?

## How is a line of business application built with angular?



# Responsive web application for all devices





# AngularJS

- A client-side JavaScript framework for building interactive web applications
- Brings simple/clean back to complex web applications
- Originally developed by Google
- Now open source
  - <http://angularjs.org>

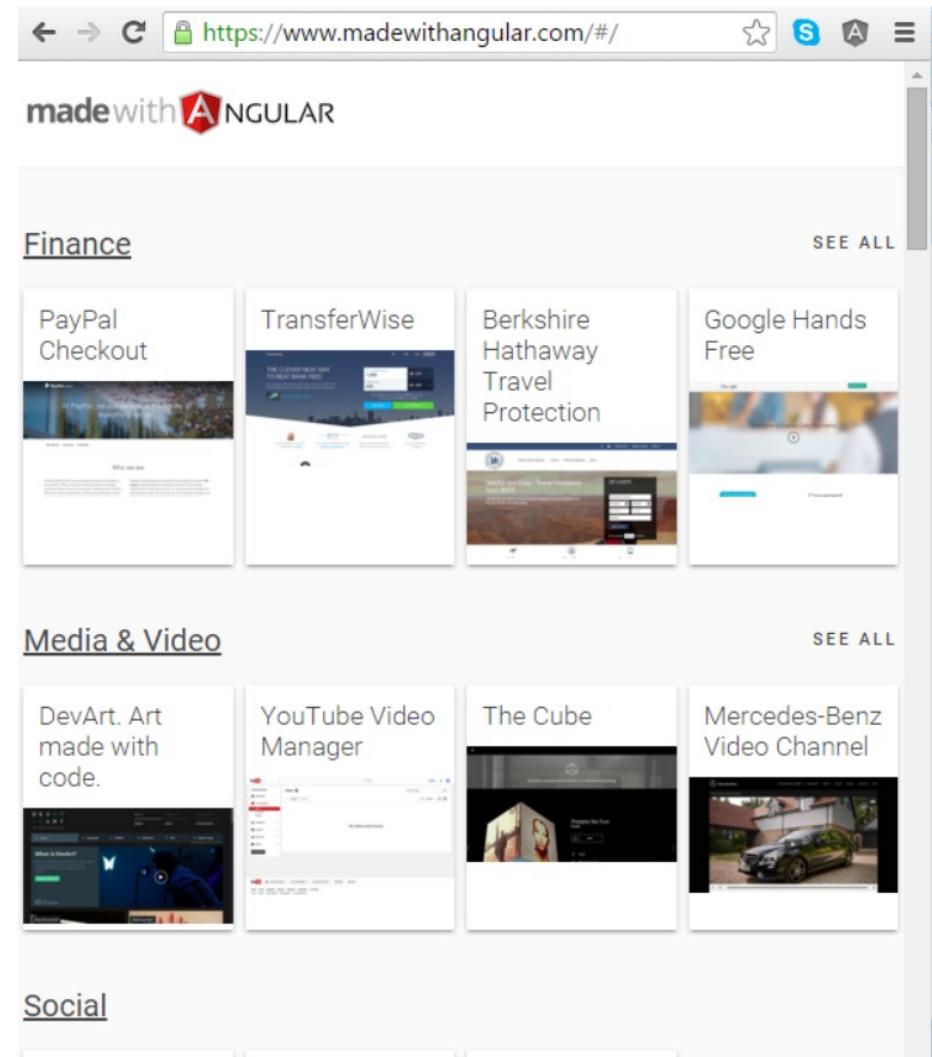
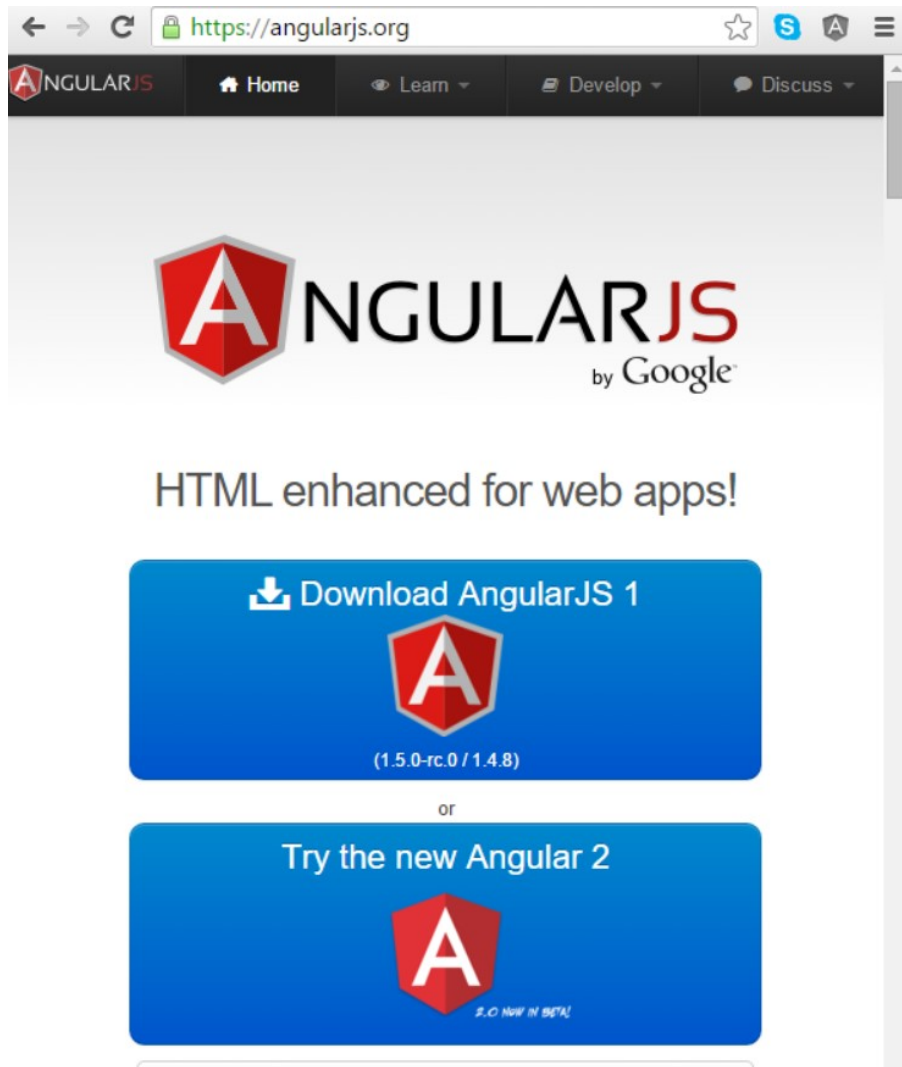


# Why AngularJS?

- Makes your HTML expressive -extends the functionality of html
- Angular is modular – a modular approach to building web application to manageable units.
- Clean controllable and testable.
- Rule-based navigation/State based navigation
- Powerful data binding (2 way data binding)
- Testable framework – focused on separation of concerns – End to End Testing and Unit Testing
- Open source and wide support



# Angularjs.org



<https://madewithangular.com>



# Two Common Types of Javascript Libraries



## DOM Manipulation Libraries Jquery

- Focus on HTML View
- Works on existing HTML & CSS
- Jquery supports animation, hiding and revealing of elements
- Manipulation of DOM elements
- Unidirectional data flow



## Model View Controller AngularJS / Backbone

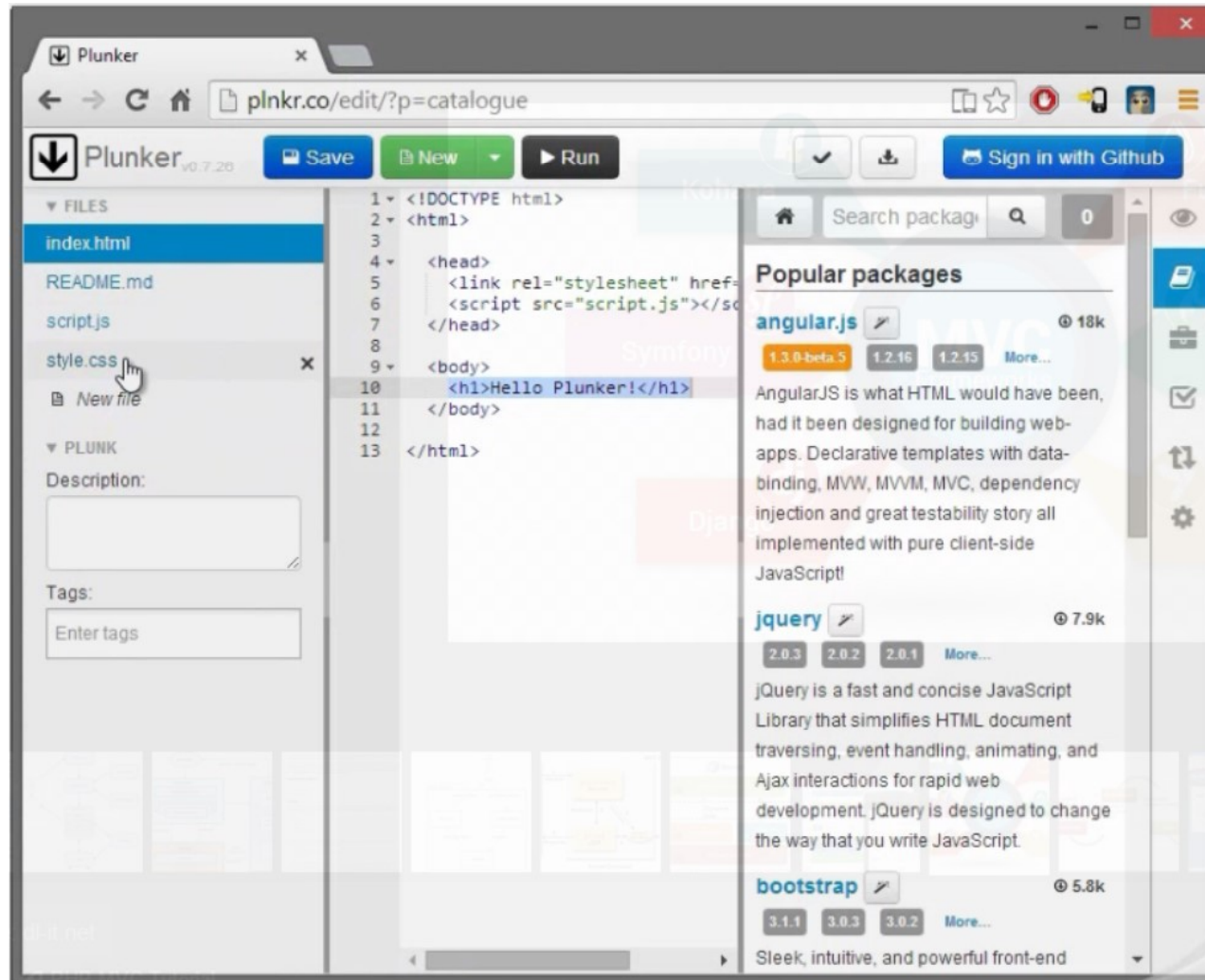
- Data first and builds view from the data
- Maintains pipeline between model, data and controller
- **Strongly encourages single master data**
- **Updating and sync – 2 way data flow.**





# Plunker

**Web-based development environment for javascript/angularjs built with angularjs**





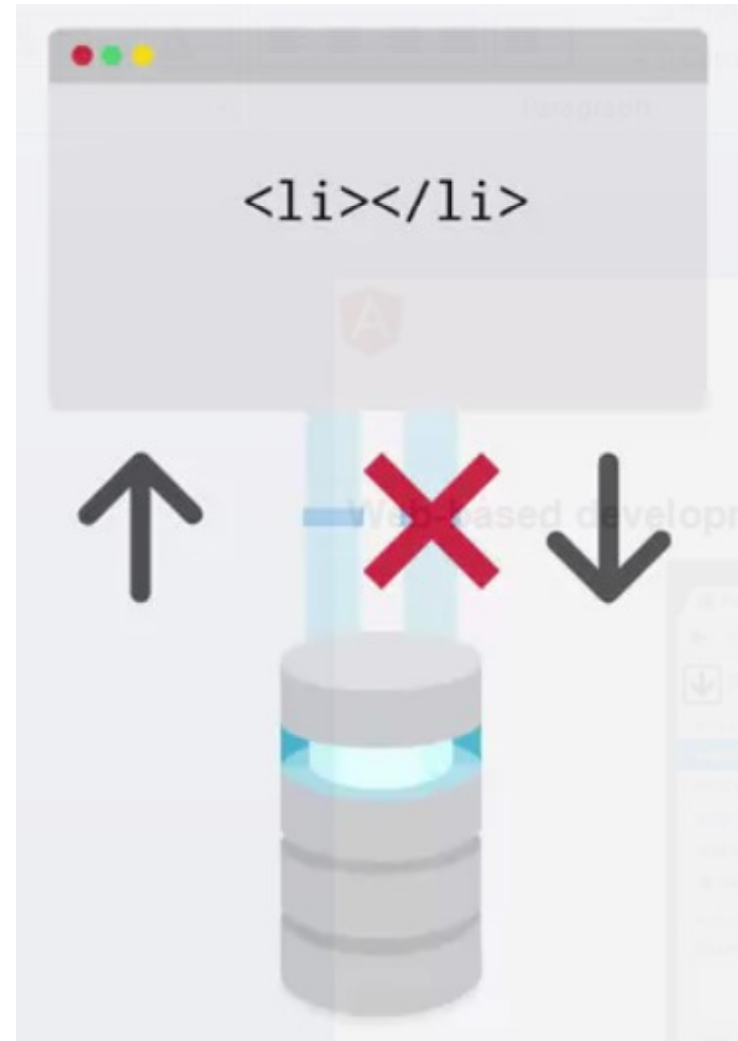
# DOM Manipulation Example

## JQUERY

### DOM Manipulation Example

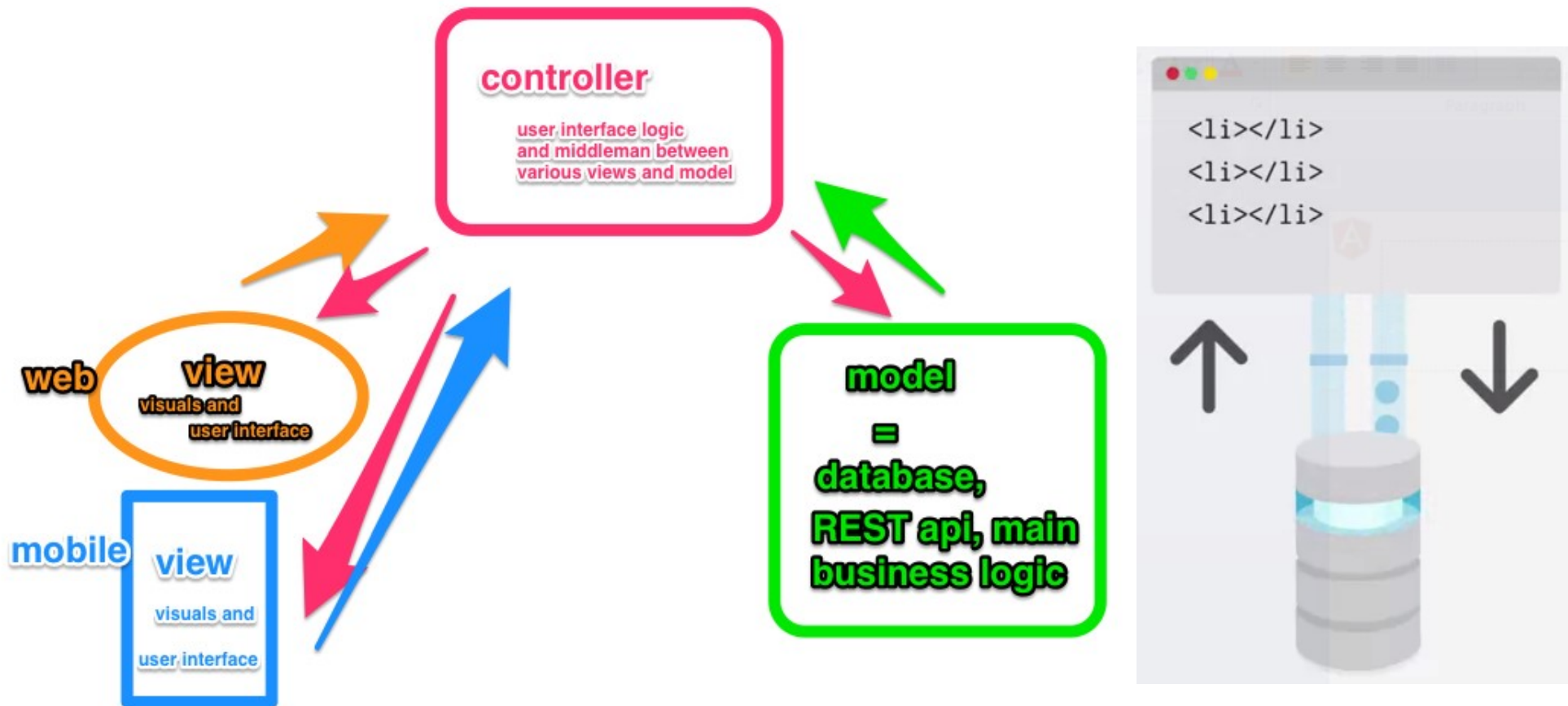
- Example S2-1-DomManipulation

- Focus on HTML View
- Works on existing HTML & CSS
- JQuery supports animation, hiding and revealing of elements
- Manipulation of DOM elements
- Unidirectional data flow



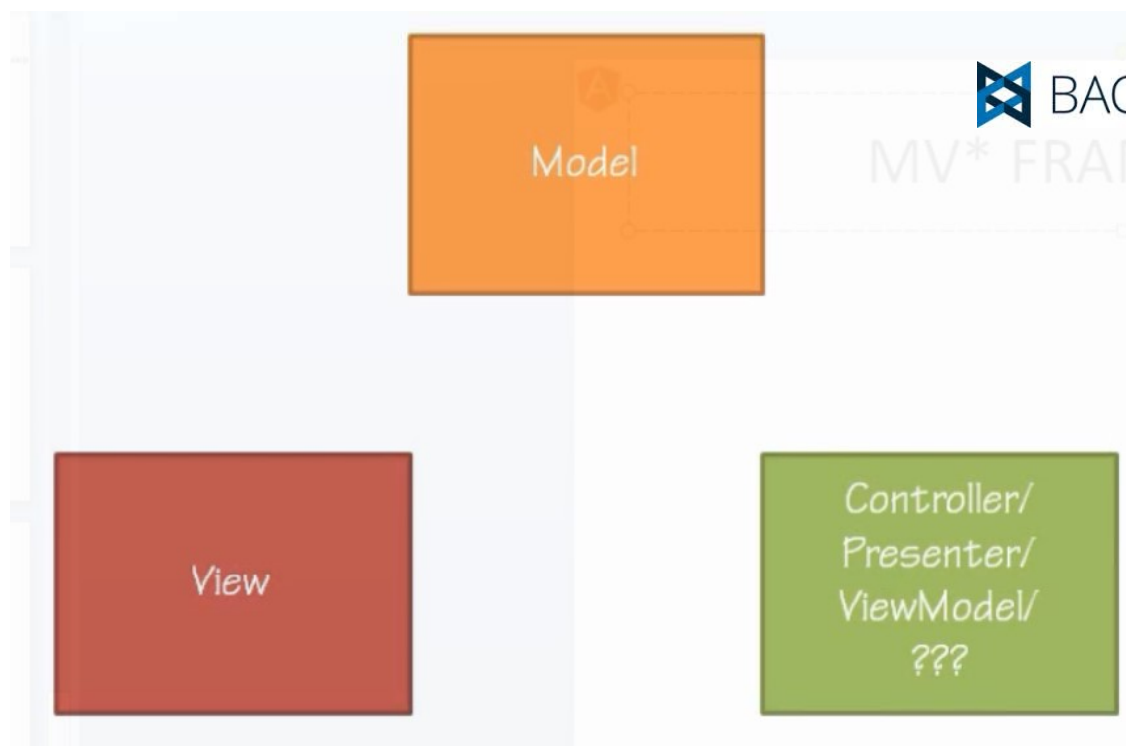
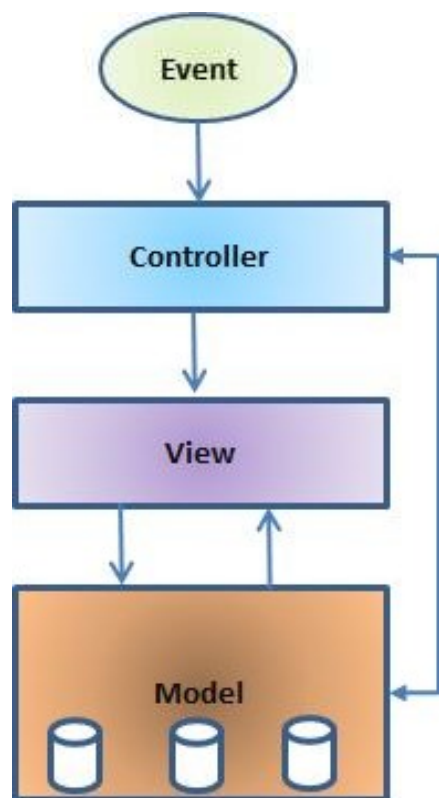


# MV\*/MVC FRAMEWORK





# MV\* FRAMEWORK



1978 at Xerox PARC, Trygve Reenskau



# MV\* FRAMEWORK ENABLES

- Handling your data
- Listening to specific user interactions
- Presenting data (or changes) back to the user
- Providing a pipeline between your data and the views the user sees, as well as the various input points that user to interact with and/or change that data.



# AngularJS FEATURES

MV\*

OPEN SOURCE

COMPREHENSIVE

Partial templates, 2 way data binding, Routing, SPA

EXTENDED  
HTML  
VOCABULARY

TESTABLE

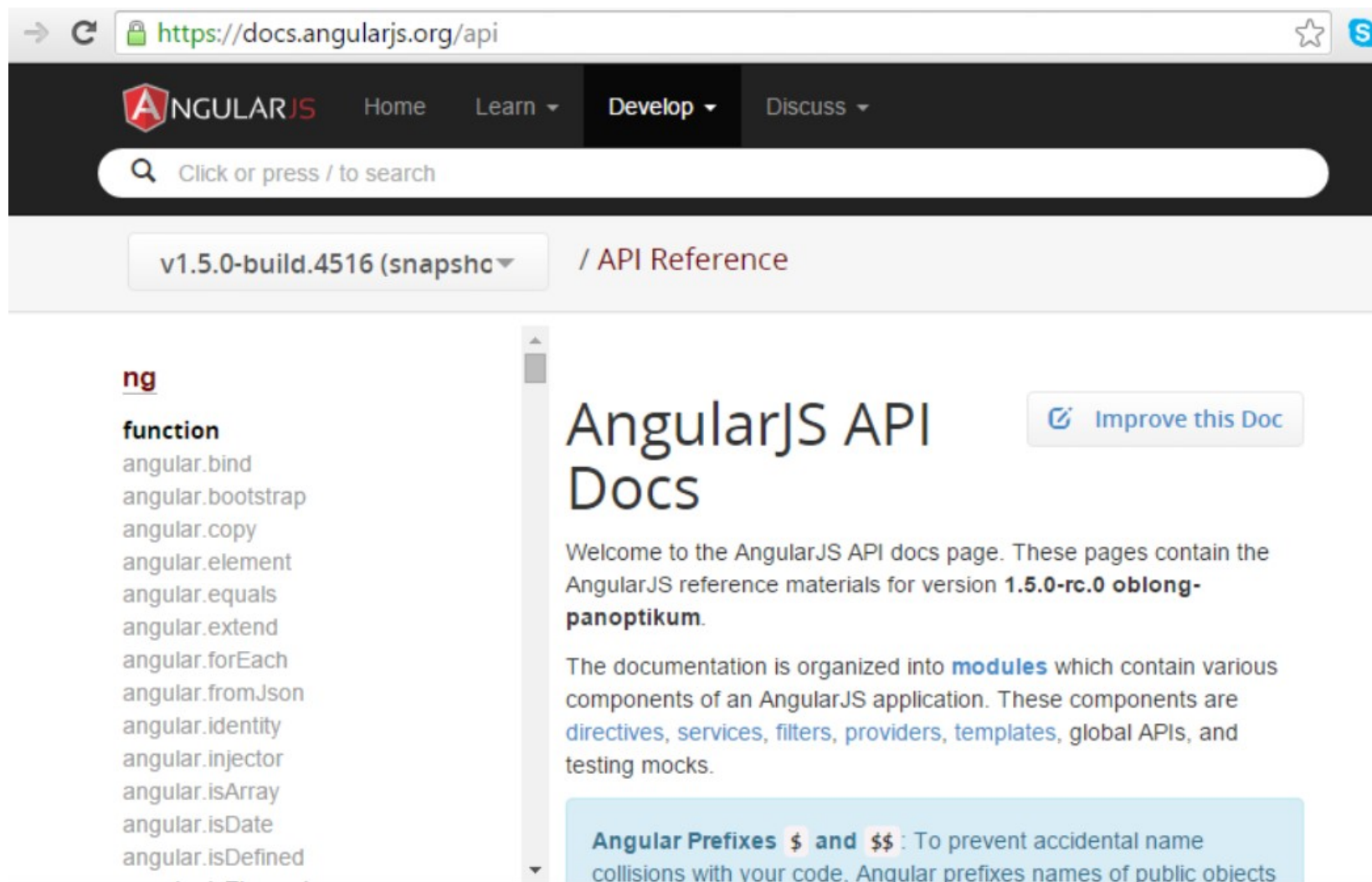
Karma, Unit Test, E2E Testing

FORWARD THINKING

Web components, Object. Observe



# Angular API REFERENCE



The screenshot shows the AngularJS API Reference page in a web browser. The address bar displays `https://docs.angularjs.org/api`. The navigation bar includes the AngularJS logo, "Home", "Learn", "Develop", and "Discuss" links. A search bar is present with the placeholder text "Click or press / to search". Below the navigation bar, a version selector shows "v1.5.0-build.4516 (snapshot)" and the current page is identified as "/ API Reference".

The main content area is titled "AngularJS API Docs" and includes a button to "Improve this Doc". The text welcomes users to the API docs page, stating that these pages contain reference materials for version **1.5.0-rc.0 oblong-panoptikum**. It explains that the documentation is organized into **modules** which contain various components of an AngularJS application, including **directives**, **services**, **filters**, **providers**, **templates**, global APIs, and testing mocks.

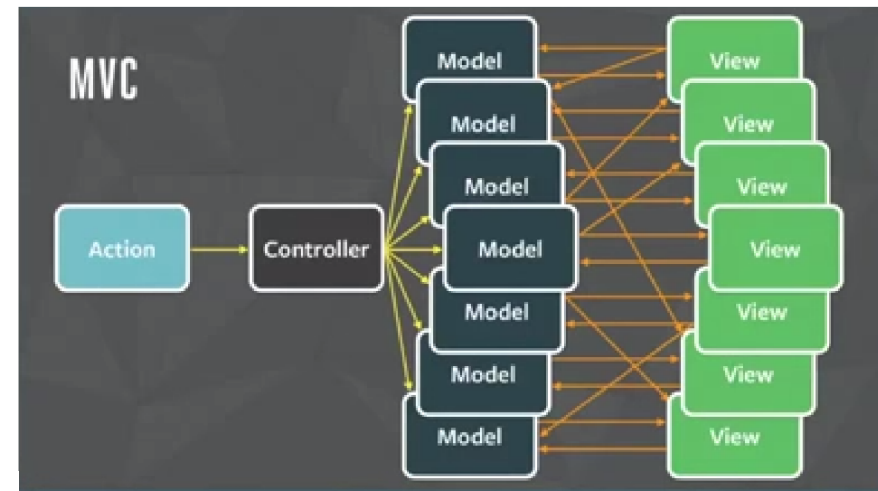
A blue box at the bottom contains the text: **Angular Prefixes \$ and \$\$**: To prevent accidental name collisions with your code, Angular prefixes names of public objects.

On the left side, there is a sidebar with the prefix **ng** and a list of functions: `angular.bind`, `angular.bootstrap`, `angular.copy`, `angular.element`, `angular.equals`, `angular.extend`, `angular.forEach`, `angular.fromJson`, `angular.identity`, `angular.injector`, `angular.isArray`, `angular.isDate`, and `angular.isDefined`.



# When to use MV\* Frameworks

- Data driven application
- Generalizing views/handling view methods
- Handling progressive enhancements
- Reusable web components
- Heavy AJAX applications
- Consuming REST/Web API Services using JSON/XML
- Objectifying for better code organization
- Modularity







# When Not to use MV\* Framework

- Server cannot handle too many requests.
- Few interactions on each page
- Data does not change very often



# Angular Architecture

TWO WAY BINDING

DIRTY CHECKING

DEPENDENCY INJECTION



# Angular Components

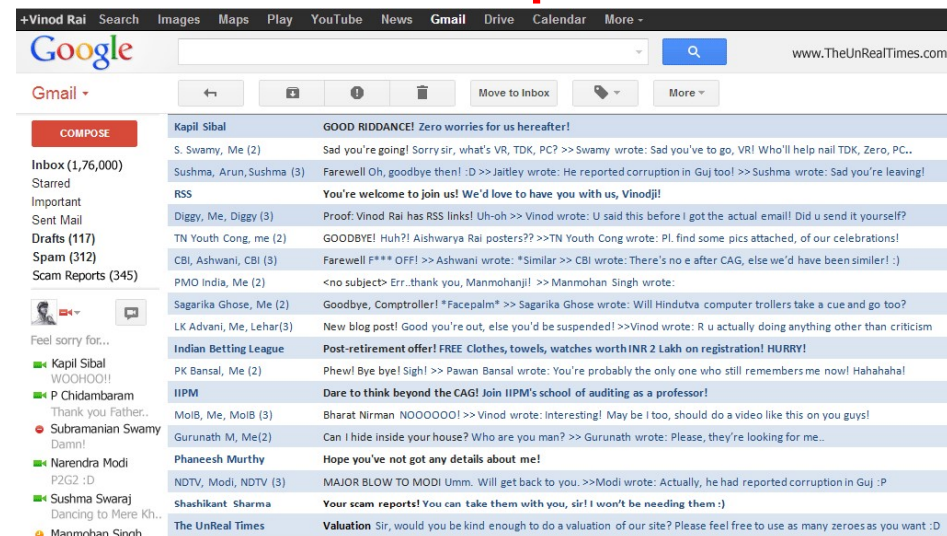
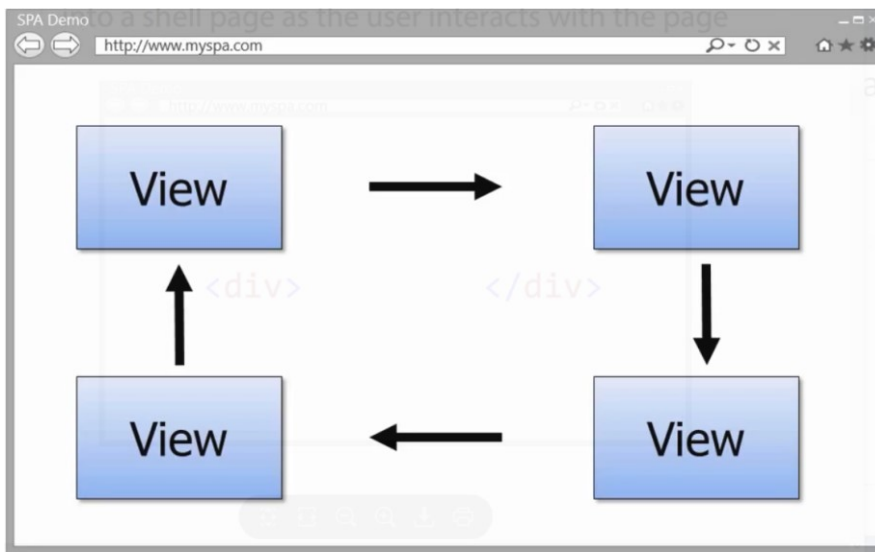




# Single Page Application Framework (SPA)

SPAs allow different views(screens) to be loaded into a shell page as the user interacts with page.

## SPA Example: GMAIL



SPAs maintain a history of views that have been displayed



# SPA

- SPAs rely on many different technologies:
  - DOM Manipulation
  - History tracking
  - Routing
  - AJAX
  - Data Binding
  - Etc..



Data Binding

MVC

Routing

Testing

jqLite

Templates

History

Factories



AngularJS is a full-featured  
SPA framework

ViewModel

Controllers

Views

Directives

Services

Dependency Injection

Validation

ONE CORE LIBRARY TO ACHIEVE ALL



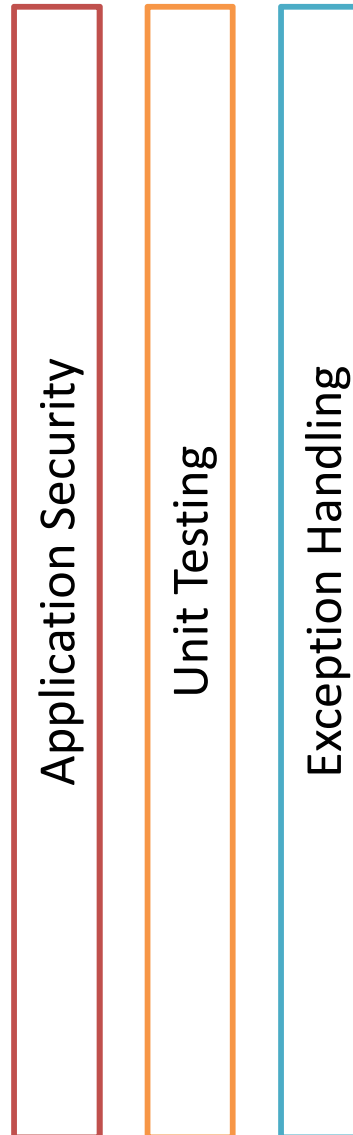
# What are the key factors in building a line of business application?

- Data is an asset
- Applications are data driven
- Amount of data is significant (volume)
- Number of data fields is significant
- Huge amount of data display/ analytical data visualization
- Data integrity is critical – validation at client side
- Data visualization / Data analytics



# How do we build Line of Business Applications

- Business Requirements
  - Application features
  - Design Considerations
  - Architecture
- Layout and Navigation
  - Page Layout
  - Style and theme
  - Navigation
- Data Access
  - Retrieving data
  - Saving data
- Data Entry Forms
  - Form layout
  - Validation
  - Submitting data
- Business Logic
  - Services
- Data Visualization/Reporting







# Other top data binding frameworks

- KnockoutJS
- Ember
- JSViews
- Can.js
- ReactJS
- ExpressJS



# Section 2 -QUIZ

- Angular thinks of HTML as if it had been designed to do what?
  - Build applications instead of documents
- What kinds of tests does angular support?
  - Unit tests and end to end tests
- When should we use MV\* frameworks
  - Data driven applications, progressive enhancements
- Name one of the ways that angular is forward thinking
  - Support for Web Components/Object.observe



## Section2 -Quiz

- What is the central component in an angular application?
  - The Controller
- Directives are part of which component?
  - Views
- In which component should you put your complex business logic?
  - Services