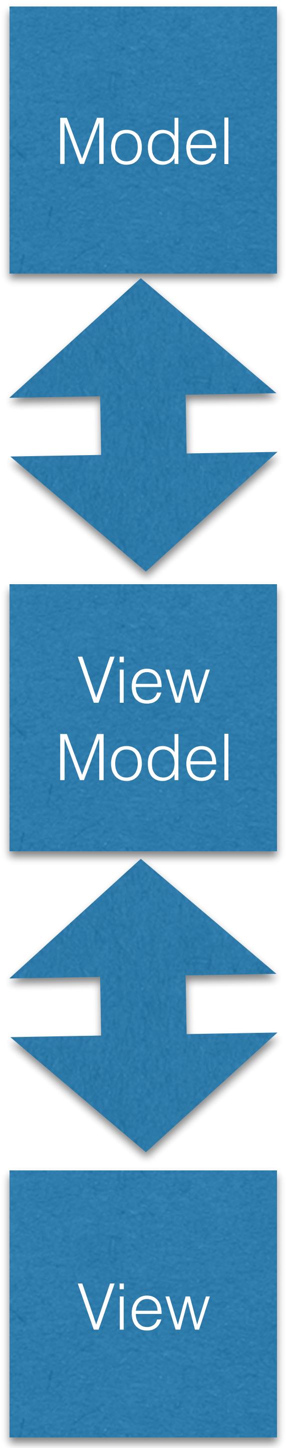
MVVM / MVVC



(Data)

(Stateful, holds logic and state of data)
(1 to 1 relationship with the view)

(Stateless, visuals)

MVVM / MVVC

MVP

MVC

Why JS ?

- widely used and popular, easy 2 write
- dynamic
- both client and server side
- vast application domain

C++

```
#include <iostream>
int main() {
    std::cout<<“Hello World”<<std::endl
}
```

Java

```
public class HelloWorld {
   public static void main (String[] args) {
          System.out.println(“Hi..”);
   }
}
```

JS

```
console.log(‘Hello World’);
```

JS Libraries

- provides methods (simple to complex functionalities) than an application can use

- speeds up development time
- cross browsing scripting and functionalities are taken care

x too reliant even if really not necessary

x effects site performance, slowing down load times

x adds dependences to ur projects

x tied to libraries coding standards

JQuery, Modernizr, Prototype, Moo Tools, Ember, Underscore, Vue ..

JS Frameworks

- provides structure, semantics, styles, layout, and functionality

- css, demo.htm, behaviour/ui element scripts, icons, web fonts ..

- pros and cons are same as that of JS libraries

Bootstrap, Foundation, Gumby, Skeleton, YAML, Angular, React ..

jQuery

- open source
- free
- AJAX
- dynamic content
- rich animations
- works across all modern browsers
- less verbose code than POJS
- CSS/JS syntax for common operations
- statement chaining for compact code
- plugins for extensibility

jQuery

- Environment setup
    - Brackets Editor
    - Chrome

jQuery selectors and filters

- work together to **access** the page content
    - selectors **selects** content/s, and filters **refine** the result set coming from a selector expression
- the results(array of objects) can be manipulated by other POJS or jQuery
    - this array is a collection of **jQuery objects** wrapped around each of the DOM elements that provide many **functions/properties** for operating on the content

jQuery selectors

- uses CSS like syntax to access page content

$(“tag1”)	selects all elements with name “tag1”
$(“#id1”)	selects the element with id attribute of “id1”
$(“.class1”)	selects all elements with class “class1”
$(“tag1.class1”)	selects all tag1 elements with class “class1”
$(“tag1#id1.class1”)	selects the tag1 element with id of “id1” and class “class1”
$(“*”)	selects all elements in the page
$(“tag1 > tag2”)	selects all tag2 that are immediate **children** of tag1
$(“tag1 tag2”)	selects all tag2 that are inside of tag1 , **descendent**
$(“tag1 + tag2”)	selects tag2 that is right next to tag1 , **adjacent**
$(“#id1 ~ tag1”)	selects all tag1 with id1 as its previous sibling , **next sibling**

jQuery filters

- uses CSS like syntax to access page content

:first, :last	selects the first / last instance of selector
:even, :odd	selects the even / odd items from the result set
:gt, :lt, :eq	selects items greater / less / equal to an index
:animated	selects all elements undergoing the animation process
:focus	selects element currently in focus
:not(expr)	selects elements not matching the expression

jQuery attribute filters

-checks if attributes are present and optionally if they have any values

$(“tag1[a1]”)	selects all tag1 elements with attribute a1
$(“tag1[a1=v1]”)	selects all tag1 elements with attribute a1 having value v1
$(“tag1[a1^=v1]”)	selects all tag1 elements with attribute a1 whose value starts with v1
$(“tag1[a1^=v1][a2*=v2]”)	selects all tag1 elements with attribute a1 whose value starts with v1, and another attribute a2 whose value contains the text v2

jQuery content filters

- filers the results of jQuery selectors by examining the content of the selectors

:contains	selects elements which contains specified text
:parent	selects elements with at least one child node (element or text)
:has	selects elements that has at least one element that mathches the selector
:first-child	selects elements which are immediate child of their parents
:last-of-type	selects elements which are the last of their type among siblings
:nth-child	selects elements which are nth child of their parent (1 based) (index or expression)

DOM Navigation with jQuery

.children()	selects elements which are children of the selector
.prev()	selects the previous element of the target element
.next()	selects the next element of the target element
.parent()	selects the parent element of the target element
.parents()	selects all the parent elements of the target element
.parentsUntil()	selects all the parent elements of the target element until the specified parent
.find()	for searching the DOM
.each()	for looping

Statement Chaining

- allows to call multiple functions in the same line of code for a single result set
- the functions will be executed in order (left to right)
- less verbose
- more readable

```
$(selector).f1().f2().f3().f4();
```