SkillBakery Presents

MASTER BACKBONE.JS

WHAT IS BACKBONE.JS

- Backbone.js is a library/framework created by Jeremy Ashkenas
- It provides client side app structure
- Its kind of MVC [Model-View-Controller] framework
- Focuses on Separation of Concerns
- Backbone.js also makes use of Routers which is helpful in creating Single Page Applications

BACKBONE ENTITIES OR COMPONENTS

- Models to represent data
- Views to hook up models to the DOM
- Events allows publish or subscribe to an event
- Routers primarily used in SPAs
- Collections comprises of a set of Models
- History
- Sync used to read or save a model to the server

INSTALLING BACKBONE.JS

- You can get Backbone.js from http://backbonejs.org/
- It depends on _(underscore) library

WHY USE BACKBONE.JS

- Consider developing a simple Todo application
- which allows creating Todo items
- We can get the data from server something like $.getJSON("/getTodoData",function(data) { } making use of Ajax and making things more responsive

Server might return a simple JSON like
{task:'Create BackBone Course',status:'in-progress',id:1}

Till this point things are pretty simple and manageable but what if we start adding more functionality to our app such as marking off a todo item,add completion date, Reorder or sort todo items.
This will make our plain old javascript application more complex and we might end up with spaghetti code where

methods can be disorganized
We might lose the data structure –
as soon as the complexity of the application grows like if we want to save all the completed to do items, or update their deadlines it will be difficult to keep track of all the changes that we will be making to our data model and hence we might loose the data structure
To resolve these issues BackBone.js can be used

WHY USE BACKBONE.JS CONTINUED…

- BackBone.JS solves the issues by providing structure to client side code
- It provides models to represent the data
- It provides views to hook up models to the DOM
- It synchronizes data to/from server

MODELS

- Quick Recap
- Fetching Data From the Server
- Destroying Data
- Get JSON From Model

MODELS CONTINUED…

- Default Values
- Parsing non-standard JSON into your Models
- Instantiating Models
- Changing Attribute Names

MODELS CONTINUED…

- Sending JSON Back to the server
- Overriding JSON Method
- Specifying the ID attribute

EVENTS IN MODELS

- How to listen for an event on a Model
- How to trigger event
- Removing an Event
- Avoiding Triggering of Events
- Special Events - These gets triggered on their own
- Change - When an attribute is modified
- change:<attr> When <attr> is modified
- destroy : when a model is destroyed
- sync: Whenever successfully synced
- error : When model save or validation fails
- all : Any triggered event

MODEL VALIDATION

- How to validate models
- Making use of isValid() function
- Making use of validationError property

MODEL INHERITANCE

- Making use of extend to create subclasses
- Calling parent methods from child classes

VIEWS

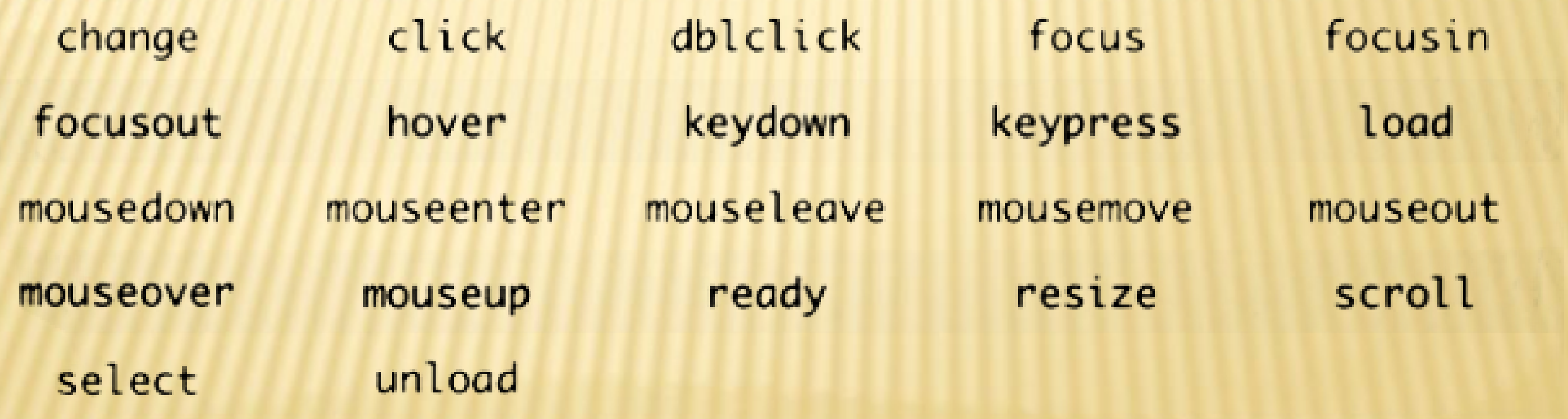
- Creating Views
- Changing Top Level element
- Adding id & class attribute to a View
- Accessing the content of View

MAKING USE OF TEMPLATES IN VIEWS

- Using Underscore Template Feature
- We can also use other template libraries like
- Handlebar
- Mustache.js or any other templating library

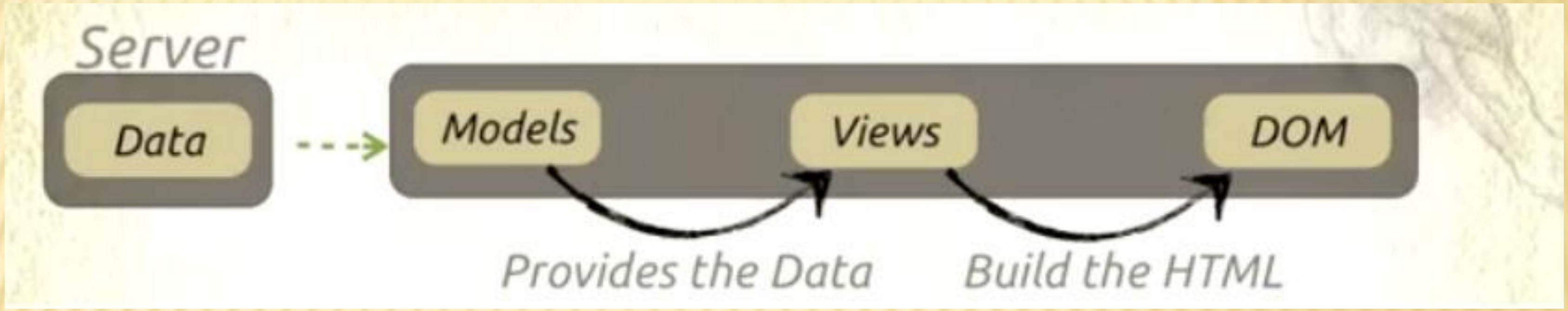
ADDING VIEW EVENTS

- Defining Events inside of a view
- A View can have more than one event
- Events supported by Backbone.js

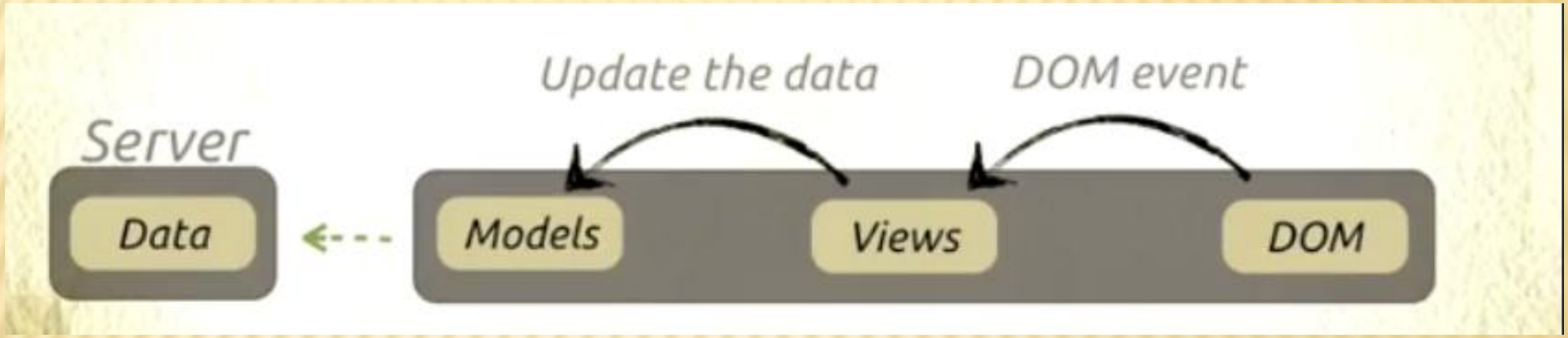


change	click	dblclick	focus	focusin
focusout	hover	keydown	keypress	load
mousedown	mouseenter	mouseleave	mousemove	mouseout
mouseover	mouseup	ready	resize	scroll
select	unload			

UPDATING MODEL WHEN VIEW CHANGES



UPDATING VIEW WHEN MODEL CHANGES



COLLECTIONS

- Creating Collections
- Add a Model Instance
- Get Number of Models in a Collection
- Get Model Instance at a given Index
- Get Model by ID
- Removing a Model Instance

COLLECTIONS

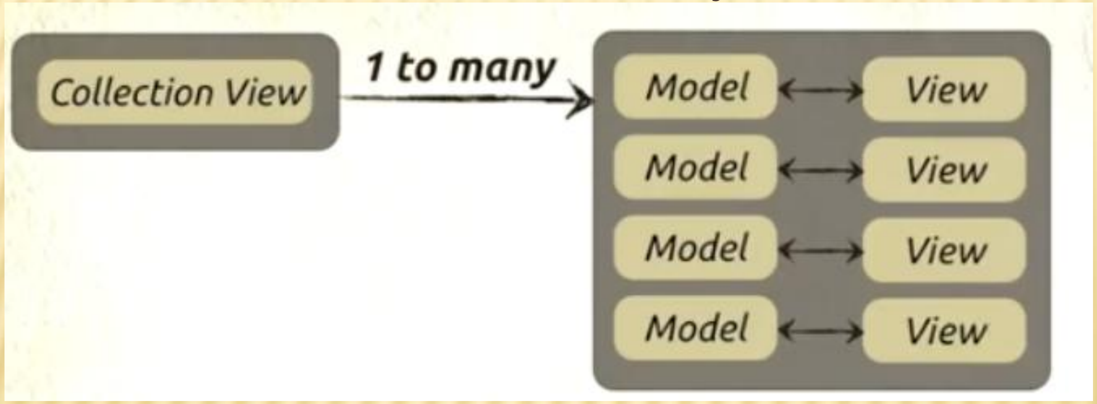
- Exploring add method
- Using Push Method to add Model Instance
- Making use of where and findWhere methods
- Making use of filter method for customized search
- Iterating Collections – Making use of each method

CUSTOMIZING COLLECTIONS

- Paging
- Sorting
- Aggregates

COLLECTIONS VIEWS

- Collection + Views = Collection Views
- Our Model and Views have one to one relationship
- i.e One Model corresponds to one view
- With Collection Views as the collection has many models
- so it has 1 to many relationship
- i.e One Collection View has many model and those models have views



- That's why a Collection view doesn't render any of it's own HTML.
- It delegates that responsibility to the model views

ROUTER AND HISTORY

- Router is responsible for mapping URLs to actions
- Defining and using Routers
- Route Matchers
- There are many more route matchers like

1. search/:query	search/backbone params	query='backbone'
2. search/:query/p:page	search/backbone/4	query='backbone',page=4
3. search/:name-:mode	search/backbone-r	name='backbone',mode='r'
4. search/*filepath	search/file/backbone.js	filepath='file/backbone.js'

- Making use of History

ROUTER AND HISTORY

- Optional Routes
- Using Router navigate method
- Optional Parameter
- Regex Criteria
- Common route

FORMS

- Creating a Form for TodoItem
- Edit a TodoItem using Form

ORGANIZING OUR APPLICATION

- Class Naming
- Problems

- 1. Global namespace - which can lead to naming collision
- 2. we have been appending name of the Type when creating a class
- 3. Managing it also becomes a problem

INSTALLING GRUNT ON WIN & MAC

- Please ensure you have node.js installed
- The solution presented works with Grunt version 0.4.1 or greater
- On Windows [Type it on command prompt]
- npm install grunt-cli –g
- On Mac [Type it on Terminal]
- sudo npm install grunt-cli -g

INSTALLING GRUNT ON WIN & MAC…

- Now we need to install Grunt.js
- Create a directory called build inside your project root
- Inside that create a file called package.json with following contents
- { "name": "Backbone-CodeDemo",
- "version": "0.1.0",
- "devDependencies": {
- "grunt": "~0.4.1",
- "grunt-contrib-concat": "~0.1.3",
- "grunt-contrib-jst": "~0.5.0"
- }
- }
- https://github.com/gruntjs/grunt-contrib-jst

INSTALLING GRUNT ON WIN & MAC…

- Run npm install now
- On Mac run sudo npm install
- Now create a file called Gruntfile.js inside our build directory
- Once you have the contents as shown in the Gruntfile.js execute grunt command

HOW THIS COURSE IS ORGANIZED

- This course begins with Backbone.js introduction and then it covers the core
- components of Backbone.js like

- Models,
- Views,
- Collections,
- Collection Views
- Routers & History
- Forms

- Each Topic is explained with examples and the code is provided for each section so that you can follow along. While discussing the topics a problem scenario is presented and the we discuss how to solve that issue at hand by making use of Backbone.js

HOW THIS COURSE IS ORGANIZED

- In Section - 8 We explore how we can better organize our code

- In Section - 9
- We develop a Backbone.js application which helps us in understanding the topics which we covered in Section 1 - 8

- This way we get to see how we can make use of Backbone.js in real applications

- In Section - 10 - We explore how we can externalize our templates, here we cover
- how we can asynchronously load our templates from external files using require.js and we also explore how we can pre-compile our templates by making use of Grunt tasks

- I hope you will find this course helpful. So lets start learning Backbone.js