Backbone.js & Grails

Jean Barmash
@jbarmash
jean@ctoschool.org

Rich UI Strategies

- Server-Side Frameworks / Components
  - GWT, ZK, Vaadin
- UI Toolkits
  - EXT-js, YUI, Dojo
- PJAX
  - Combination of server-side / client-techniques
- DOM Manipulation
  - JQuery, Prototype

JQuery Example

```
$.getJSON('/todo', function(data){
```

→ receives

```
{ id:1, description: 'Learn Grails', status: 'complete' }
```

→ HTML / DOM

```
<input type='checkbox' data-id='1'>Learn Grails </input>
```

Problems:

Connection to original data is lost.

No abstraction for the data.

Difficult to manage multiple views of data

Client-Side MVCs

- Backbone.js
- Spine.js
- Sencha
- Angular.js
- Ember
-
- Shine for Single-Page Apps, esp with multiple views and complicated interaction model.

Server MVC vs Client MVC

- Grails
  - M – Domain Objects, GORM
  - V – Views, GSP, Templates
  - C – Orchestrate Generate Views, Process Requests, Glue Everything Together
- Client-MVC (Backbone)
  - M – Model, Communicate over REST
  - V – View, Templates
  - C - ???, Browser Event Handling ?

Backbone is a Micro-Framework

- Less than 1K LOC
- Some building blocks for organizing code
- Not a full-fledged framework
- Extensible & Pluggable
-
- Like Ratpack vs Grails or Sinatra vs Rails
-
- Good Discussion – Backbone vs Ember
  - http://smus.com/backbone-and-ember/

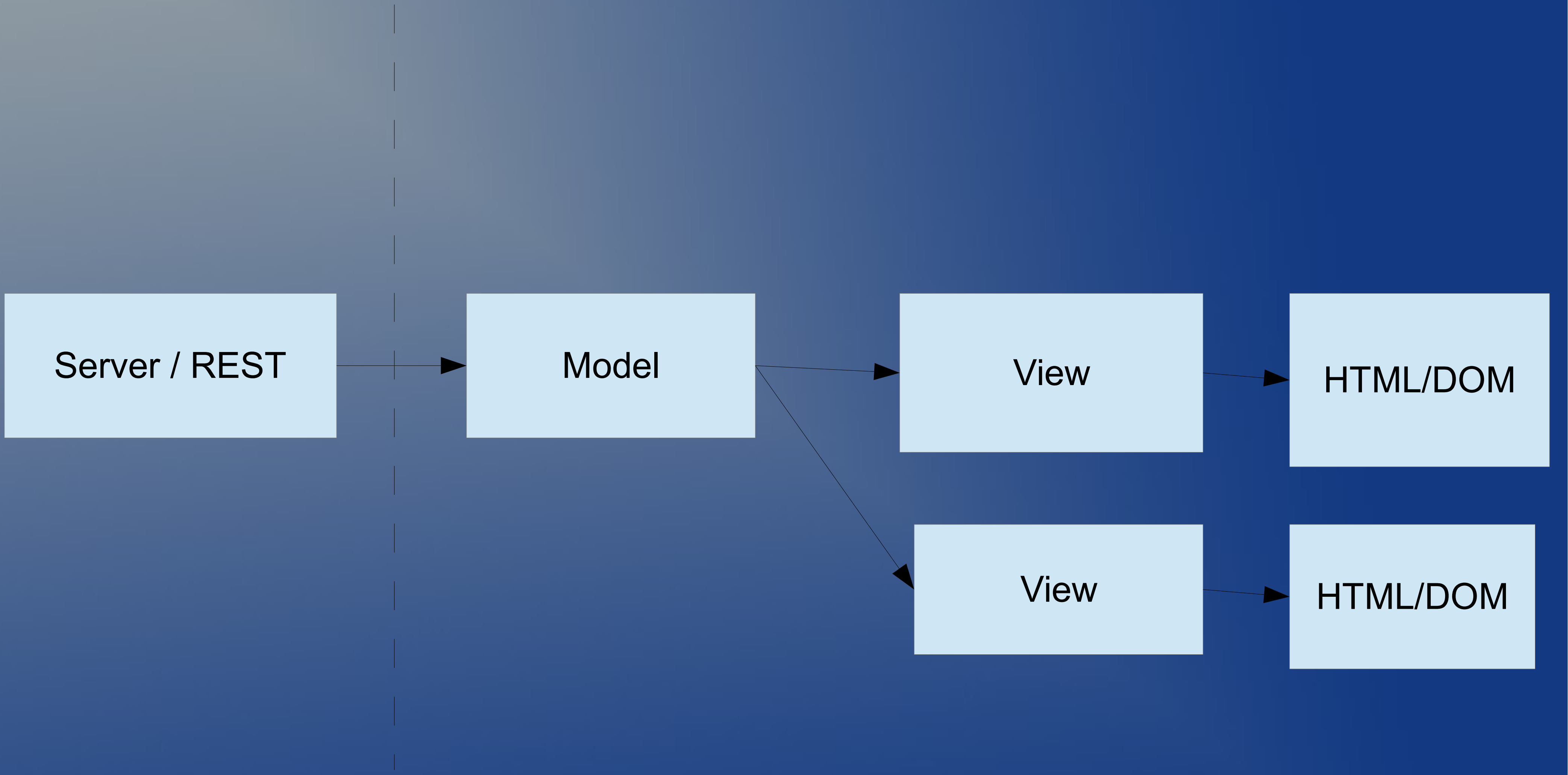
Backbone.js Components

- Backbone.js (5.6Kb zipped, v 0.9.2).
- Underscore.js (only hard dependency)
- json2.js, and either jQuery (> 1.4.2) or Zepto.
  - Lots more functionality
  - RESTful persistence
  - History support via Backbone.Router
  - DOM manipulation with Backbone.View
- Backbone.sync

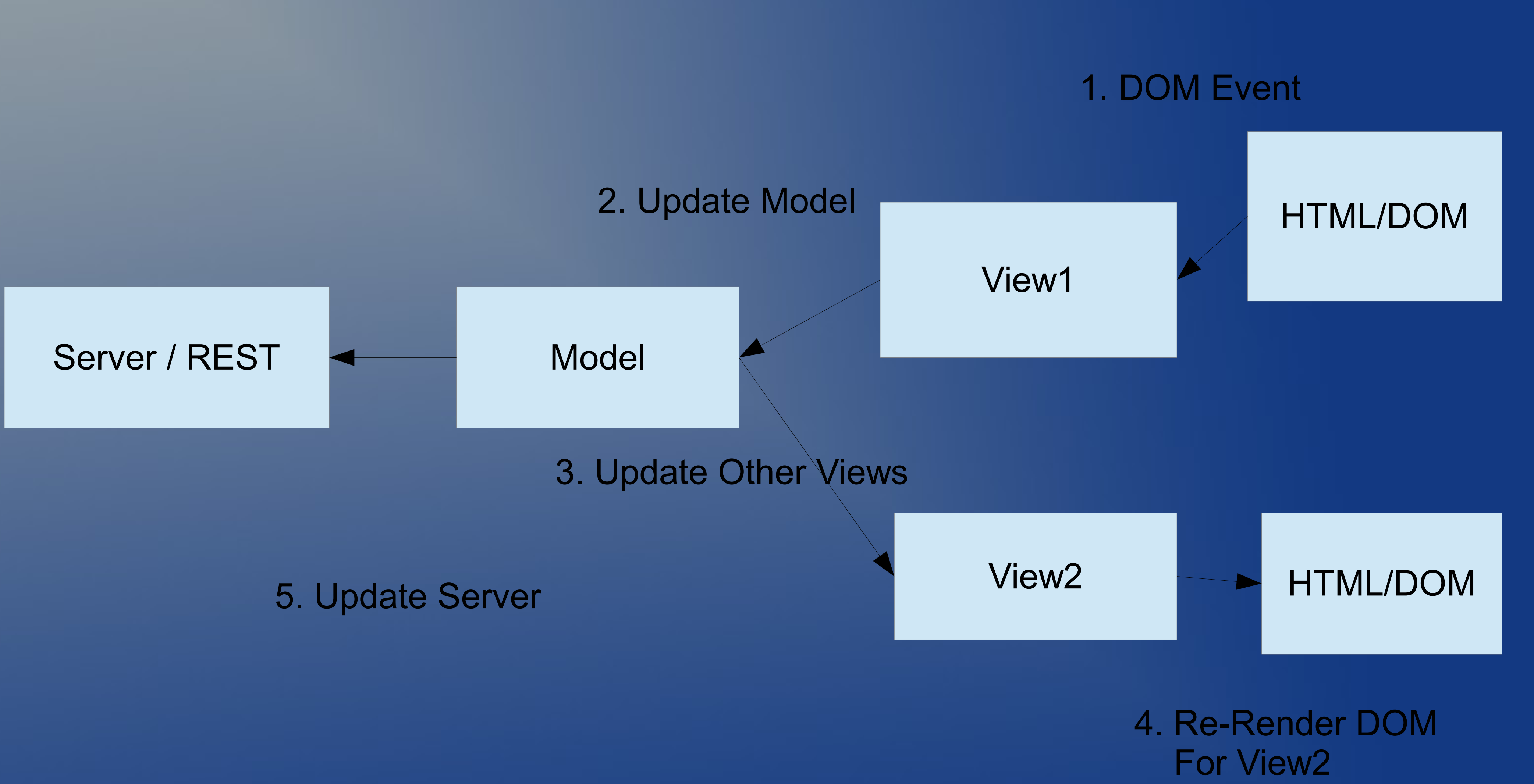
Backbone.js

- Client-Side App Structure
- Objects to model data
- Automatic Synchronization to/from server
- Modular & Extensible
- Views display Models as HTML/DOM
- Events Sent between Views and Models

Flow - Rendering



Flow – User Actions


1. DOM Event

2. Update Model

3. Update Other Views

4. Re-Render DOM
For View2

5. Update Server

Backbone Concepts

- Model
- Collection
- View
- Events
- Routes / History

Models

- window.Todo = Backbone.Model.extend({});
- No strict schema definition like in Grails
- Persistence Configuration
- Attributes
- Defaults
- Functions
- Events – change, sync, destroy, change:attr
- Collections
  - List of Models

Model Example

```
window.Todo = Backbone.Model.extend({
 urlRoot:'todos',
  // Default attributes for a todo item.
  defaults: function() {
    return {
      done:  false,
      order: Todos.nextOrder()
    };
  },
  toggle: function() {
    this.save({done: !this.get("done")});
  }
});
```

Manipulating Model

```
var item = new window.Todo();
item.set({text:'Hello Gr8Conf!'});  //fire change
item.save();    //POST /todos with JSON
item.get('id');  → 3
item.destroy();  //DELETE /todos/3
item.toJSON();
item.on('change:text', function() {
    alert('Change happened!');
});
```

Model – Backbone Methods

- initialize() - constructor
- fetch() - retrieve from server
- save() - save to server (post / put)
- set(JSON)
- get(attribute)
- validate()
- destroy()

Collection – Backbone Methods

- Add(models)
- Remove(models)
- get(id)
- GetByCid()
- at(index)
- Comparator()
- Where()
- Fetch()
- Reset()
- create()

Collection – Underscore.js Methods

- forEach(each)
- map (like groovy collect)
- reduce
- find
- filter
- every
- groupBy
- indexOf

Views

- 1 - Responsible for Rendering Data

```
var TodoListView = Backbone.View.extend({
  tagName : 'li', //div is default
  render: function() {
     var html = "<span style="font-color='red'>" + this.model.get('description')
+ "</font>";
     $this.$el.html(html);
  }
}
```

Views – Listen for Events & Model Changes

```
var TodoListView = Backbone.View.extend({
  initialize: function () {
      this.model.on('change', this.alertUser, this);
   }
   alertUser() {
     alert ("Something Changed!");
   }
   events: {
       "click span" : "makeEditable"
        "keypress .todo-input"      : "updateOnEnter"
    }
    makeEditable: function() { }
}
```

Collection Views

- Views for Collections
- Delegates rendering responsibility to contained model views
- Also respond to events

DEMO

- http://github.com/jbarmash/grails-backbone-todo

Resources

- Main Backbone Page - http://backbonejs.org/
- Backbone and Ember - Microframework vs Framework
  - http://smus.com/backbone-and-ember/
- Turorials
  - http://backbonetutorials.com/

  Step By Step from JQuery to Backbone
  - https://github.com/kjbekkelund/writings/blob/master/published/understanding-backbone.md/
- Code School - Anatomy of Backbone.js
  - http://www.codeschool.com/courses/anatomy-of-backbonejs
- Backbone - Hacker's Guide
  - http://dailyjs.com/2012/07/19/mvstar-2/
- Tutorial On Building Single Page Backbone.js app
  - http://andyet.net/blog/2010/oct/29/building-a-single-page-app-with-backbonejs-undersc/
- NYC -
- Groovy / Grails Meetup http://meetup.com/grails
- CTO School – www.ctoschool.org