Backbone Views

- Responsible for the HTML that the user "sees"
- Extends *Backbone.View* base class
- Will listen for events in the model and render designated sections in the HTML

Views Creation

- Created using *Backbone.View.extend()*
- Provides an *initialize* function
- A *model* and an attachment *element* can be passed during creation
- A single model or a collection can be passed during creation

Binding to the DOM

- A view can be bound to HTML using the *el* attribute
- The *el* element:
  - Can reference an existing DOM element by selecting it using standard CSS selector
  - Defaults to *div* when nothing is passed during element creation
  - Can be changed after definition using *.setElement(<selector>)* view method

Rendering Content

- Content is rendered using the render function. It is defined in view creation.
- The *render* function can deal with a single model or a collection depending on the view definition
- Content can be removed entirely from the DOM using the *.remove()* view method
- The *$el* property is provided as a shorthand for jQuery *$(view.el)*.
  - It can be used to use jQuery methods against the view, like finding a nested element using *$el.find(<selector>)*
- The *$(<selector>)* is a shorthand for *$el.find(<selector>)* like *$("#mydiv")*

View Events

- Used to make the view interactive
- Defined in the *events* section of the view definition
- Use jQuery "on" function to provide callbacks for DOM events within the view
- Specified using the format **`{"event <selector>": "callback function"}`**
- If no selector is specified, the event will be bound to the root *el* element

Backbone Templates

- Used to separate logic (JavaScript) from presentation (HTML)
- JSON objects are inserted into the HTML, which contains conditional, placeholders, and other helpers
- You can used Underscore as well as other templating engines like Mustache and Handlebars

Underscore

- Contained inside a special *script* tag having type set to "text/template"
- The script tag should have an *id* for easy manipulation by JavaScript
- The template contains standard HTML, and it can contain JavaScript code enclosed in <% %>
- Template rendering is done as follows:
  - The script element's html is passed to the _.template() function, e.g. **`template = _.template($("script-id").html())`**. Where *template* is the return value.
  - *template* from the above function is actually another function that accepts the model collection. It is passed on like this:
    `result = template({"library":self.collection.toJSON()})`
    Where library is the collection name used inside the script tag. It returns the rendered HTML
  - Finally the resulting HTML is added to the placeholder element as follows:
    `self.$el.append(result)`
- For a neater code, the template part can be placed in the *view* definition in the *template* section

Handlebars

- Downloaded freely from www.handlebarsjs.org and placed after backbone.js script reference
- Uses a special script tag like Underscore, but the type set to text/x-handlebars-template. An id must be set
- Does not accept JavaScript code within the template. Uses built in helpers instead
- Accepts data in JSON format
- The dot notation can be used in the JSON object the same way it is used in JavaScript
- Can be interchanged with Mustache templating engine as both use the same style

Handlebars Compile

- The output of *html()* is passed to *Handlebars.compile()*
- The rest of the render function code remains the same

Handlebars Expressions

- Displays variables inside {{ }} tags
- Accepts comments in {{! }} or {{!-- —}}
- Uses *{{#expression}}* and *{{/expression}}* for block level code
- *#each* is used for looping
- *#if* and *#unless* are used for conditionals. *#unless* is the inverse of *#if*

Which one to choose?

- Depends largely on the type of project
- Underscore has the advantage of being already required so no need for new script files. Suitable for very simple templates
- Handlebars is more suitable for larger projects with many different views
- Mustache and Handlebars can be easily interchanged as both use largely the same syntax