Routers

- A way to organize multiple views into a structure that can be navigated
- A *route* refers to a specific section of the web application denoted by a hash (#).
  Example: http://localhost/#books
- A *router* refers to the code that manages the *route*. It listens to changes in the URL

Router Creation

- Extends the *Router* base class
- The routes are defined in the *routes* section
- To start monitoring routes, *Backbone.history.start()* function is called
- Two initial routes should be defined at least:
  - The homepage route routes: *{"":"handler"}*
  - The default route. It is used as a fallback if the requested route could not be found: {"*default":"defaultRoute"}
- Order is important: The homepage and the default route should be the last in the route list

Route Parameters

- Provides more functionality to the route, e.g. http://localhost/#books/1
- Defined in Backbone as *route/:name* where *name* is the parameter passed to the route handler function
- Any number of parameters can be passed to the route and, hence, to the handler function
- A parameter can be made optional by placing it inside brackets. In this case, the handler must check for the presence of the optional parameter
- A parameter can be made more friendly be prefixing it with text. This text is specified in the route. Example: routes *{"books/book:bookNumber"}*, which will match http://localhost/index.html#books/book3 for example
- An arbitrary number of parameters can be passes by prefixing the parameter with a *. Example: *{"books/:*path}"* which will match http://localhost/books/book1/page34 for example. Where *path* will evaluate to *book1/page34*

Route Events

- Fires when the URL on the page changes to a defined route
- Defined using
  router.on("route:routeHandler: function(page){}")
  Where *router* is the router object
- Must be defined before *Backbone.history.start()*
- Can be called anywhere in the application
- Useful when other parts of the application need to listen to specific route changes

Manual Navigation

- You can change the URL of the page to a specific,defined route by invoking *router.navigate(<route>)* where router is the router object
- The defined route will NOT be invoked unless you specify *{trigger:true}* as the second function parameter
- You can prevent the route from being added to the browser's history by appending *replace:true* to the second function parameter object

Routers as Controllers

- To make a *router* act as a *controller,* one way to do this is the following:
  - Create a generic model that will be shared between the router and the view
  - Attach this model to the view
  - Any changes that happen to this model will be done by the router object. For example, a URL change will change the ID of the current model
  - The *router*, then, calls the view's *render* method to update the HTML according to the new model data
  - Since the *view* already has a reference to the current model, it will use this model data in its *render* function
- In other MVC frameworks, there is a specific object that is responsible for this type of work, called *Controller*. But in Backbone, the *router* can assume this role.