Introduction to BackboneJS

ROHAN RAJORE

History of Web

- Up until 2005, web sites were pretty static, with all the real business logic implemented in the server side, using languages such as PHP, Java, and .NET.
- During 2005, Ajax (Asynchronous JavaScript and XML) gained popularity and changed how web sites would be used forever.
- Representational State Transfer (REST) provides an architecture for client-server communication over HTTP. All Ajax requests are made using RESTful services, and when creating Backbone applications, you will invariably be consuming such services in your data model.
- Another key milestone in JavaScript's maturity was the release of John Resig's jQuery in 2006, a framework that acknowledged the need for a more controlled approach to writing JavaScript for web applications.
- Now that Ajax had proved itself and the JavaScript ecosystem was providing more robust libraries and frameworks, single-page applications were easier to implement.

Introduction to BackoneJS

- Released in late 2010
- Backbone was created by Jeremy Ashkenas, who also wrote CoffeeScript.
- Enables the creation of SPA
- JS library to create large frontend applications
- Provides MVC on the client side
- Unopionated
- Extensive community support and extensions
- NewyorkTimes, Airbnb, SoundCloud, FourSquare etc

Model View Controller

- Model View Controller (MVC) is a pattern that separates the three main areas of any code base that involves a user interface. The pattern uses three key terms.
- *Model*: The *model* consists of all the data you want to represent in your application.
- *View*: The *view* is the visual representation of the model and can be thought of as the presentation layer. In our web applications, views are the HTML/DOM elements that are presented to the user within browser.
- *Controller*: The *controller* deals with the input from the user and updates the state of the model, essentially acting as the glue for the entire structure. As the user needs to change data in the model, they will use the controller as an intermediary for this.

How Backbone Supports Model View *

- Backbone has four core concepts: Backbone.Model, Backbone.View, Backbone.Router, and Backbone.Collection.
- Backbone.Model and Backbone.Collection as both dealing with the model side of things.
- Backbone.View represents the presentation of data but could also be seen as taking on some of the responsibilities of a controller. The real rendering of the view will be done in the HTML, and the view in Backbone really just provides the capability to pass on the model information through to the HTML.
- Backbone.Router will simply map URLs to functions, but in a way it can also participate in controller duties.

Why to choose Backbone?

- *Backbone is a library, not a framework*
- *JavaScript applications need structure*
- *Rich documentation and a large user community*
- *It scales well, and credible companies are using it*
- *Your code base is a jQuery mess*

Three Reasons Backbone Might Not Be Right for You

- *Proof-of-concept applications*
- *You’re not comfortable with lower-level JavaScript*
- *You are creating a small web page*

Prime components in BackboneJS

- Router
- Model
- Collections
- Views & Templates