Netid: cz32
Name: Chenlai Zhang

Introduction to Backbone.js

Abstract

In this paper, I will give an introduction to why we need backbone.js, how backbone.js work. I will also compare backbone.js with react.js, and have an overall analysis of both the advantages and disadvantages of the backbone.js towards react.js. From the compare and analysis, we will discover that for simple javascript applications, we prefer to use react.js, while for heavy javascript applications, we prefer the other.

Introduction

Backbone.js is a very light framework or front-end library that enables us to organize our Javascript codes in MVC style. This framework is so small that read the whole structure in a few hours because it has less than one thousand lines of javascript codes. Though it is small, it may supply structures for heavy javascript applications.

Explanation

We need to use backbone sometimes because it solves some problems. When we write javascript codes, we usually store the data either in global variables or DOM data attributes. However, data stored in global variables can easily affect each other, while data stored in DOM data attributes are difficult to handle because data can only be stored in string type using this method. Backbone nicely solves this problem by using a kind of objects called Model. [3] Model objects retrieves and populates data. It properly represent the data and the associated data handling methods. The purpose of backbone is to add structure into Javascript codes. [1] We know that javascript applications can easily become very messy and very quickly. We often do not know where to store the data, where to put the methods, and how to tie methods together. Backbone helps us a lot by providing different objects to handle these questions. Model objects and Collection objects will represent data and collections of data. View objects will update corresponding DOM when data change. Events enable different components in this system listen to each other.

Besides the powerful Model objects, backbone.js also provides us with other necessary components, View objects and Collection objects. Each View object is implemented as a render function which represents and redraws part of the view in a project. View objects is associated with particular parts of DOM, and also associated with specific Model objects or Collection objects. When the Model object is updated, it triggers an event, and then the view tied to the object will re-render itself according to the event. Likewise, when you change a view object, it will update the corresponding model object. [1] Collection object stores a list of Model objects, and when these Model objects change, like any Model object updating information, adding to or removing from the Collection object, or changing the orders, the Collection object will trigger an event, and then update the corresponding View objects according to the corresponding event. This behavior provides a clear separation of concerns that make the framework in order.

From the previous introduction, we may find that different components in this framework are well decoupled from each other. They communicates through events: View objects call render functions when corresponding Model objects or Collection objects trigger events, and Model objects will also update its status when the bounded View objects trigger events. All these objects and APIs are connected to existing application over a RESTful Json interface. [2]

Compare and Analysis

Compared to react.js, backbone has many advantages. First, its components are highly decoupled from each other and well organized, thus the codes will never get messed even if the codebase becomes very large. Second, we do not have to store data in DOM any more, and we now can store data in Model objects instead, in which we can handle those data more reasonably. [2] Third, the code of backbone.js is very simple and well documented, and there are only a few simple concepts as Model objects to understand, so it is easy to read. It is very wise for a front end starter to learn backbone.js first. Fourth, it is very lightweight thus very fast. The framework is highly compact and minimalistic because according to its MVC pattern, it contains only some basic components that are necessary to the application. Since it is so small and simple, it is a great foundation to build our own framework on. Therefore, the developer must have enough experience or do lots of research before you are able to fully utilize all the available libraries and plugins which are compatible with the framework. [3]

Although it has so many advantages, this framework also has some drawbacks compared to react.js. First, it provides some basic components to construct the structure rather than provide the structure. Since the develop needs to decide many details on how to structure his application, there are a lot of work to do. Second, we need to carefully consider memory management and view lifecycle management. If not, state changes will easily lead to memory leaks. [2] Although many functions not available from backbone itself can be realized by third-party plugins, we need to make plenty of choices on several alternative plugins when we create an application. It takes a lot of time to do enough research before choose the best one for the certain project, which violates the saving-time purpose of the backbone framework. Besides these, it also lacks the support for two-way data binding, so we have to implement a lot of midware to update the corresponding components while the associated components change. And View objects in this framework have direct access to the DOM, which makes it hard to do unit-test, thus makes the framework more fragile and reduces its reusability. [2] Besides, the common practice to look up

DOM elements in this framework is by using CSS selectors. Even if we do a few changes to CSS, it may disable the CSS selector and destroy the whole application.

Conclusion

By analysis of both advantages and disadvantages of the framework, we have to admit that it is very useful in some applications in structuring our javascript codes into MVC style. Decoupled components in this framework and event-driven communication in the system help us never end up in a mess. Its features also increase the code extensibility since we can extend your functionality simply by creating some new objects and having these objects listen to the corresponding objects. Using this framework enables us to make our code more compact, clean and maintainable. From the comparison, we can also conclude that react.js is more suitable for simple javascript applications, while backbone.js is more suitable for heavy javascript applications.

References

[1] Backbone.js documentation

http://backbonejs.org/

[2] Why do you need Backbone.js?

https://cdnjs.com/libraries/backbone.js/tutorials/why-would-you-use-backbone

[3] Why Backbone.js?

https://roost.bocoup.com/blog/why-backbone/