Handwritten Digit Recognizer

Pattern Recognition

Jihirsha Golwala, Mallika Kulkarni, Ruchi Shah, Saraswathi Natarajan

San Jose State University

One Washington Square, San José, CA- 95192-0080, USA

I. INTRODUCTION

In a Single Page Application, the entire application runs as a single web page. The web page rendering is not server side but client side. So, page refreshes do not occur after the page loads. Instead, the presentation logic is loaded when the page loads and views are swapped based to user requests to give an illusion of page refresh. The client communicates with the server asynchronously via the XMLHTTPRequest Object. The data used between client and server communication is in JSON format
In a traditional server side web application design, every time a client needs fresh data, an HTTP request is made to the server. The server side controller handles the request sends back the view after making the necessary changes to the data. The overall SPA architecture is almost the same as the traditional server side architecture, with a three major changes

1. **No page refreshes**
   In SPA, views are not complete web pages. All the tools for creating and displaying views are downloaded at the time of initial page load. When a new view is needed, it is loaded with the help of Javascript acting on the DOM. No page refresh ever happens.

2. **Presentation logic resides on the client side**
   HTML and data is combined on the client side rather than on the server

side. The data is typically fetched vial AJAX calls and Javascript on the browser combines the data with the HTML. Now-a-days third party

3. **Difference in server side rendering**
   The involvement of the server can vary for different calls, as against traditional architecture where rendering occurs only via the server. The approaches include partial server side rendering or no server side rendering at all in which case, the server just sends data via an XMLHTTPRequest Object in Json format.

4. **Fat Client**
   Since the responsibility for the presentation layer lies with the client, the client becomes a fat client with some business logic.

II. SPA Architecture

The architecture of an SPA is almost the same as the architecture of a traditional web application.

Fig2: SPA Architecture

Components of an SPA include

1. **Model**
   Is a JSON data structure that serves as data for an HTML page. It has support for getter / setter properties and property change signal.
2. **Events**
   Represents any event on the DOM. The javascript on the browser can then act on these events.
3. **View**
   A view is a part of the HTML page that pairs with the model to produce the illusion of a complete web page. It can listen to DOM events. In some frameworks, the Controller manages pairing between View and Model
4. **Routing**
   Navigation within an app via URLs
5. **Syncing**
   Persisting changes occurring through AJAX calls.

III. Important considerations while building an SPA

1. **Choosing a framework**
   There are multiple frameworks to choose from. Some are client side, some are server side. Meteor JS is probably the only full stack framework. Some

frameworks have been described in detail in the later sections.

2. **Choosing a client side template**
   Jade, HandleBars and Underscore are examples. Some frameworks have inbuilt support for certain frameworks.
3. **Modular development**
   This promotes decoupling, code reusability, ease of development and debugging.
4. **Package Management to resolve dependencies**
5. **UI Considerations**
   Error handling, localization, form handling, alerts and triggers, graphics, uniformity, custom elements.
6. **Browser Considerations so as to make the app compatible with all browsers**
7. **Performance and Analytics**
8. **Testing**

IV. 1. Technologies used for developing single page applications (SPA's):

1. **Javascript Frameworks**:
   Javascript is a dynamic programming language mostly used in the web browser allowing the client side scripts to interact with the user, alter the document being displayed, communicate asynchronously and control the browser. Several frameworks are developed using Javascript that contribute towards the development to highly functional single page applications. Some of the Javascript frameworks which are highly used for developing single page applications are AngularJS, EmberJS, Meteor JS, React JS.

   Angular JS: Angular JS often referred as Angular is an open source web application framework maintained by Google to address many challenges encountered while developing single page applications. It aims at simplifying the development and testing of such applications by providing a framework for client side development using model–view-controller and model-view-viewmodel architectures. It is mainly based on bidirectional UI data binding. Data binding is an automatic way of updating the view whenever the model changes, as well as updating the model whenever there is any change in the view. In this framework the controller and model state are maintained in the browser. Hence new pages are generated without interaction with the server.

   Ember JS: It is a client side Javascript framework based on model-view-controller

software architecture pattern. Developers can create scalable single page applications by incorporating best practices in to the framework that provides a rich object model, declarative 2 way data binding, computed properties and a router for managing the applications state.

Meteor JS: Meteor gives a radically simple way of developing real time mobile apps and web apps with a single code base. It interacts with both the client and server. Its client gui provides the look and feel and response of a thick client app using meteor's blaze framework or integrating with angularjs and react js.

2. AJAX

Ajax is a technique for creating dynamic and fast web pages. It allows the web pages to be updated asynchronously by exchanging small amount of data with server behind the scenes. This means it is possible to update the web page without reloading the whole page. AJAX is based on internet standards and uses combination of XMLHttpRequest and Javascript DOM.

3. Web Sockets:

Web Sockets are a part of HTML5 specifications that perform bi-directional stateful real time client server communication. Web Sockets are much more powerful than ajax in terms of performance and simplicity.

4. HTML5 Local Storage:

This is one of the useful feature of html5 which is useful while developing single page application. Local Storage helps storing the application data locally in the browser without causing any hindrance in the performance of website. It is more secure and its storage limit is around 5MB which is much more than cookies. Local storage is per origin. All pages from one origin can store and access the same data.

5. Browser Plugins

Browser Plugins: Asynchronous call to the server can also be achieved using browser plugins such as Silverlight, Flash or Java Applets.

V. Design Approaches for developing Single Page Applications (SPA's):

1. **Data transport (XML, JSON and AJAX):** Server requests typically result into either raw data or a newly rendered HTML page. Javascript on the client side updates a partial area of the DOM.

When raw data is returned, often a client side Javascript process is used to translate the raw data into HTML, which is then used to update a partial area of the DOM.

2. Thin server architecture:

A SPA moves the logic from the server to the client. This results in results in the role of the web server evolving into a pure data API or a Web service. This architectural shift in which the complexity has been moved from the server to the client, with ultimately reducing the overall system's complexity can be coined as 'Thin server architecture'.

3. Thick stateful server architecture:

The necessary state of the client page is kept in the server. Hence, in this manner when the request is done to the server, the server sends the appropriate HTML and/or JavaScript with the appropriate changes to bring the client to the desired state, along with updating the state of the server. Most of the computation is done on the server and the HTML page is also rendered on the server. This approach needs more server side processing but the advantage of this approach is simple development model.

4. Thick stateless server architecture:

The client page sends the data representing its current state to the server usually using AJAX requests. Using this data, the server is able to reconstruct the client state of the part of the page that needs to be updated and can generate the necessary data or code, which is returned to the client to bring it to a new state, usually modifying the page DOM tree according to the client action which motivated the request. This approach requires more data to be sent to the server and may require more computation resources per request to partially or fully reconstruct the client page state in the server. As there is no per-client page data kept on the server, this approach is more scalable and, therefore, AJAX requests can be dispatched to different server nodes with no need for session data sharing or server affinity.

VI. Advantages of single page:

Single page web applications provide users a rich user interface, giving the users a feel of using an application rather than a web site.

As all the data is loaded upfront on the initial page load, single page web application tend to be more responsive.

SPA provides a fluid transition between the different page states and maintaining the state becomes easier because there is only one single point of entry.

Above all, SPA has a better Google Page rank because these applications have higher content density.

Unlike traditional web applications, Single page web applications support offline processing. When the user uses the application in offline mode, all changes can be synchronized later.

SPA also improves performance by dramatically reducing the load on the server, as most if the validation and processing is done on the client side and only very less data transfer is required. This also makes caching a lot easier.

VII. Challenges in single page web application

1. Google AdSense

"Google AdSense is a program run by Google that allows publishers in the Google Network of content sites to serve advertisements, that are targeted to site content and audience." In general, AdSense does not work well with AJAX based web applications and as SPA design does not refresh the web page as well, a SPA based application can result in low impression and low revenue.

2. Google Analytics

The Google Analytics tool relies heavily on new web pages loading in the browser. Since there is only a single page load occurring in SPA, the browser never triggers a page load and hence the analytics package cannot figure out who is doing what on the web page.

3. Search engine optimization

According to webopedia, "Search engine optimization is a methodology of strategies, techniques and tactics used to increase the amount of visitors to a website by obtaining a high-ranking placement in the search results page of a search engine (SERP) -- including Google, Bing, Yahoo and other search engines." Traditionally, search engines do not support JavaScript execution on crawlers. This posed as a problem for the websites wishing to adopt SPA model. But recently Google started crawling URLs containing hash fragments starting with "#!" to overcome this.

4. Browser history

In traditional web pages, the browser navigates the user to the previous web page

loaded, when the user clicks the Back button of the browser. SPA breaks this, because only a single page is loaded. Supposing that a user scrolls down a SPA page and accidentally clicks the "Back" button of the browser (expecting to view the initial parts of the page), the browser takes the user to a different web page, creating a very bad user experience. To achieve the desired result, the SPA should be capable of resurrecting the same screen state from information contained within the URL hash.

5. Speed of initial load

In SPA, the first page load is expected to download the entire framework and all needed application code for rendering the web page. This results in slower page load.

6. Client side must enable JavaScript

If the client browser does not support JavaScript or if for some reason the user has disables JavaScript, then SPA does not work.

7. Client server code partitioning

In a single page application, if there is a network loss and the user has to reload the page, then the user is taken to the start of the page, resulting in a bad user experience.

9. Security

Although security is not an issue unique to single page applications, SPAs tend to be less secure as compared to traditional pages due to cross site scripting (XSS).

CONCLUSION

In this paper we have discussed the rise of single page AJAX applications. We have also discussed about the architecture and the considerations when developing a single page application. The technology stack for developing a SPA was also analyzed. Finally we have emphasized the advantages and disadvantages of using SPA.

REFERENCES

1. SPA Design & Architecture, Scott, Emmitt, Jr. (Manning)
2. https://en.wikipedia.org/wiki/Single-
3. http://itsnat.sourceforge.net/php/spim/s
4. http://www.techferry.com/articles/single
5. http://code.tutsplus.com/tutorials
6. https://en.wikipedia.org/wiki/Single-page_application
7. http://stackoverflow.com/questions/21862054/single-page-application-advantages-and-disadvantages

8. http://www.slideshare.net/hsplmkting/benefits-of-developing-single-page-web-applications-using-angular-js
9. http://programmers.stackexchange.com/questions/144717/advantages-and-disadvantages-of-building-a-single-page-web-application