Comparasion of Ajax Table Loading

Experiment Design for Computer Science

Meng LI(201620728)

1. Introduction

There is no production or result about my research now, so I want to talk about a framework of me. This framework can be found in my github (https://github.com/lm2343635/Mengular), which includes 2 parts: JavaScript Ajax Loading and Java Template Engine. In this report, I will compare two loading method in the function of JavaScript ajax table loading (I will call it ajax table loading from now). At first, I will explain the meaning of ajax table loading.

For traditional web development, we use PHP, Java, ASP .NET and other programming language for background service program, they have to prepare their own template file such as jsp file in Java Web development. To create a table, code would like this:

```
<tbody>
<%
for(Item item in items) {
    %>
    <tr>
        <td><%=item.attribute_1%></td>
        <td><%=item.attribute_2%></td>
        ...
        <td><%=item.attribute_n%></td>
    </tr>
    <%
}
%>
</tbody>
```

This style is same in PHP and other background service programming, which is difficult to read, especially for front end Engineer who only care about browser side. This style is not comply with MVC(Model-View-Controller) standard. And we can image a situation that a table has many rows, for instance, more than 10000 rows. It is no doubt that we could not use such style to show this 10000 rows in a html documents because it is too scroll to create this html document and transfer it to browser. If we can load 100 rows at first, and when we scroll the page down, more 100 rows are loaded dynamically by ajax, things will be better. In fact, peoples are doing like this method. Web applications of Google photos will not load all your photos, when you scroll it down, more photos will be loaded automatically.

The question is how to create dom element dynamically after loading table data asynchronously There is a function called `$.append()` in jQuery framwork which can append a child element to a parent element, the core question is how to create this child element. We can call a function `document.createElement()` provided by native Javascript API or the equal function in jQuery like `$("<tr>")`. However, this method is just suitabled for such simple situations. With the increment of child element's complexity, a lots of html document is written in javascript which is not convenient to read and rewrite. Thus, we hope a template egine for front end development just like the same thing in background service programming.

2. Goal

At first, we prepare a template as which has been introduced in part 1.

```
<tbody>
  <tr id="${id}$" class="mengular-template example-table-template">
      <td>${attribute_1}$</td>
        <td>${attribute_1}$</td>
        ...
        <td>${attribute_1}$</td>
  </tr>
</tbody>
```

The row document `<tr>` is the part that we want to load asynchronously. It has two classes incuding `mengular-template` which is defined in `mengular.css` in order to instruct the element is a row template, and `example-table-template` which is assigned by ourself for finding template element. Then we can user the core function `$.mengular(template, data)`(it is a jQuery style function) in Mengular framework to load `items` which is an arry from server asynchronously like this: $("#example-table").(".example-table-template", items); or $()