Vue介绍

1 Vue是什么

1.1 定义

- Vue 是一套用于构建用户界面的渐进式框架
- 使用Vue框架，可以完全在浏览器端渲染页面，服务端只提供数据
- 使用Vue框架可以非常方便的构建 单页面应用 (SPA)

1.2 关于作者

- 国人 尤雨溪
- 百科介绍： https://baike.baidu.com/item/%E5%B0%A4%E9%9B%A8%E6%BA%AA/2281470?fr=aladdin
- 微博： https://weibo.com/arttechdesign?is_hot=1#1528176039582

1.3 相关网站

- 官方网站： https://cn.vuejs.org/
- GitHub： https://github.com/vuejs/vue

2 前端框架

2.1 三足鼎立

- React
- Angular
- Vue

2.2 Angular、Vue、React的区别

Vue与React

- React与Vue 都采用虚拟DOM
- 核心功能都在核心库中，其他类似路由这样的功能则由其他库进行处理
- React的生态系统更庞大，由ReactNative来进行混合App开发; Vue更轻量
- React由独特的JSX语法; Vue是基于传统的Web计数进行扩展(HTML、CSS、JavaScript)，更容易学习

Vue与Angular

- Angular1和Angular2以后的版本 是完全不同的两个框架； 一般Angular1被称作Angular.js, Angular之后的版本被称作 Angular
- Vue与Angular的语法非常相似
- Vue没有像Angular一样深入开发，只保证了基本功能。 Angular过于笨重
- Vue的运行速度比Angular快得多
- Angular的脏检查机制带来诸多性能问题

2.3 MVVM

- Model
- View
- ViewModel

2.4 Vue的优点

- 不存在依赖
- 轻便（25k min）
- 适用范围广(大中小型项目、PC、移动端、混合开发)
- 本土框架,社区非常活跃
- 语法简单、学习成本低
- 双向数据绑定（所见即所得）

2.5 使用框架开展一个项目的时候，需要考虑哪些方面?

1.性能

如果一个网站在性能方面存在问题，它将会损失超过一半以上的用户。

对于框架性能，你可以在网上查询到各类测试，你可以了解框架的代码结构、逻辑处理，判断是否能够满足你对性能的需求。

2.扩展性

对于一个需要长期维护的项目而言，经常会有各种各样的功能添加进来，这时扩展性就显得尤为重要，如果你在前期选择了一款满足前期的框架，但后期你需要使用某个插件来完成某个功能，或者基于什么完成一个功能，这时候你发现网上并没有检索到相关内容，内心是否充满了心塞。

3.维护性

一个项目的生命周期不是三天两天，而前端的发展则是爆炸式的。在你选择框架的时候是否考虑过官方在后续的一段时间是否会一直对框架进行更新维护？如果不确定，是否已经有了官方放弃维护后的解决方案?

4.兼容性

这里的兼容性指的不是浏览器兼容，而是框架与其他框架及工具的兼容，使用这个框架对于你的开发环境是否有影响，对于你的开发 IDE 是否有影响。

Vue.js 适用具有以下性质的项目：

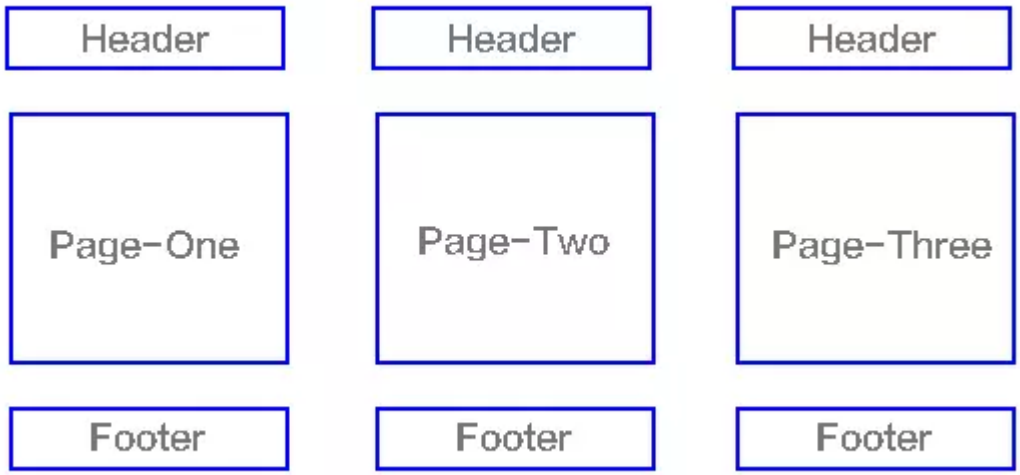
- 对浏览器兼容要求不高，不需要兼容至IE6-8；
- SPA开发；
- 对性能较高要求;
- 组件化。

总的来说，如果你是一个 MVVM 框架新手，那么 Vue.js 就是你最好的进阶工具，如果你是一个已经掌握了其他 MVVM 框架的老手，那你会发现 Vue.js 更加简单轻便。

3 多页面应用和单页面应用

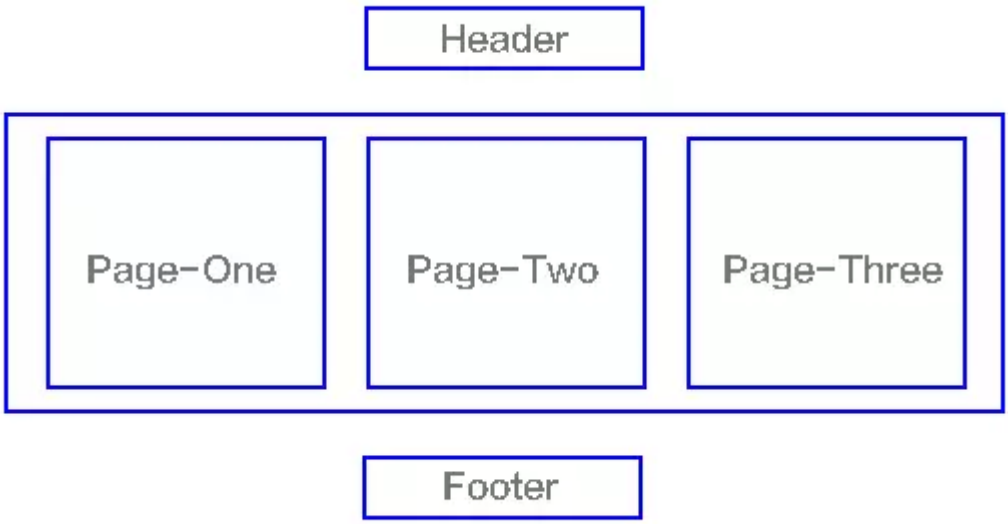
3.1 多页面应用（MultiPage Application，MPA)

多页面跳转刷新所有资源，每个公共资源(js、css等)需选择性重新加载，常用于 app 或 客户端等



3.2 单页面应用（SinglePage Web Application，SPA）

只有一张Web页面的应用，是一种从Web服务器加载的富客户端，单页面跳转仅刷新局部资源 ， 公共资源(js、css等)仅需加载一次，常用于PC端官网、购物等网站



3.3 两者对比

	单页面应用	多页面应用
组成	一个外壳页面和多个页面片段组成	多个完整页面构成
资源公用(css,js)	共用，只需在外壳部分加载	不共用，每个页面都需要加载
刷新方式	页面局部刷新或更改	整页刷新
url 模式	a.com/#/pagetwo a.com/#/pagetwo	a.com/pageone.html a.com/pagetwo.html
用户体验	页面片段间的切换快，用户体验良好	页面切换加载缓慢，流畅度不够，用户体验比较差
转场动画	容易实现	无法实现
数据传递	容易	依赖 url传参、或者cookie 、localStorage等
搜索引擎优化(SEO)	需要单独方案、实现较为困难、适用于追求高度支持搜索引擎的应用	实现方法简易
试用范围	高要求的体验度、追求界面流畅的应用	适用于追求高度支持搜索引擎的应用
开发成本	较高，常需借助专业的框架	较低，但页面重复代码多
维护成本	相对容易	相对复杂

4 Vue入门

4.1 安装

直接`<script>`引入

下载地址

- 开发环境版本 https://vuejs.org/js/vue.js 包含完整的警告和调试模式
- 生成环境版本 https://vuejs.org/js/vue.min.js 删除了警告、进行了压缩

CDN

```
https://cdn.jsdelivr.net/npm/vue@2.5.16/dist/vue.min.js
https://cdn.jsdelivr.net/npm/vue@2.5.16/dist/vue.js
# 以手动指定版本号
```

NPM

在用 Vue 构建大型应用时推荐使用 NPM 安装[1]。NPM 能很好地和诸如 webpack 或 Browserify 模块打包器配合使用。同时 Vue 也提供配套工具来开发单文件组件。

```
npm install vue
```

构建工具 (CLI)

```
npm install -g @vue/cli
vue create my-project
```

4.2 Vue基本演示

创建实例

```
var app = new Vue({
  el: '#app',

})
```

添加数据

```
var app = new Vue({
  el: '#app',
  data: {
    message: 'Hello Vue!'
  }
})
```

模板渲染（文本差值）

```
<div id="app">
  {{ message }}
</div>
或者
<div id="app" v-text="message">
</div>
```

绑定属性的值

```
<span v-bind:title="message">
    鼠标悬停几秒钟查看此处动态绑定的提示信息!
 </span>
```

双向数据绑定

```
 <p>{{ message }}</p>
 <input v-model="message">
```

事件

```
<div id="app">
  <p>{{ message }}</p>
  <button v-on:click="reverseMessage">逆转消息</button>
</div>
```

```
var app = new Vue({
  el: '#app-5',
  data: {
    message: 'Hello Vue.js!'
  },
  methods: {
    reverseMessage: function () {
      this.message = this.message.split('').reverse().join('')
    }
  }
})
```

循环

```
<ol>
  <li v-for="todo in todos">
      {{ todo.text }}
  </li>
</ol>
```

条件

```
<p v-if="seen">现在你看到我了</p>
```

Vue组件化

```
// 定义名为 todo-item 的新组件
Vue.component('todo-item', {
  template: '<li>这是个待办项</li>'
})
```

```
<ol>
  <!-- 创建一个 todo-item 组件的实例 -->
  <todo-item></todo-item>
</ol>
```