前端开发之走进Vue.js

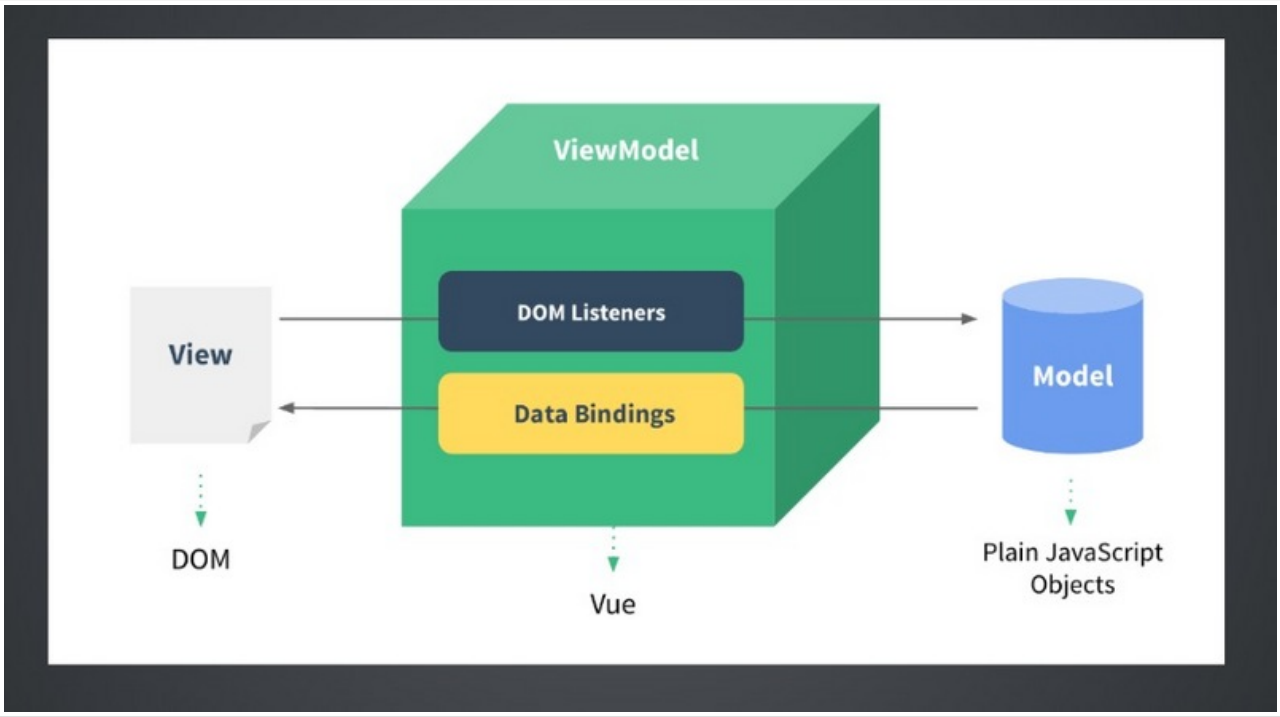
vue.js 前端 前端框架 劳卜 1 天前发布

Vue.js作为目前最热门最具前景的前端框架之一，其提供了一种帮助我们快速构建并开发前端项目的新的思维模式。本文旨在帮助大家认识Vue.js，了解Vue.js的开发流程，并进一步理解如何通过Vue.js来构建一个中大型的前端项目，同时做好相应的部署与优化工作。

文章将以PPT图片附加文字介绍的形式展开，不会涉及知识点的具体代码，点到为止。有兴趣的同学可以查看相应的文档进行了解。

Vue.js简介

从上图的介绍中我们不难发现Vue.js是一款轻量级的以数据驱动的前端JS框架，其和jQuery最大的不同点在于jQuery通过操作DOM来改变页面的显示，而Vue通过操作数据来实现页面的更新与展示。下面便是Vue数据驱动的概念模型：



Vue.js主要负责的是上图绿色正方体ViewModel的部分，其在View层（即DOM层）与Model层（即JS逻辑层）之间通过ViewModel绑定了DOM Listeners与Data Bingings两个相当于监听器的东西。

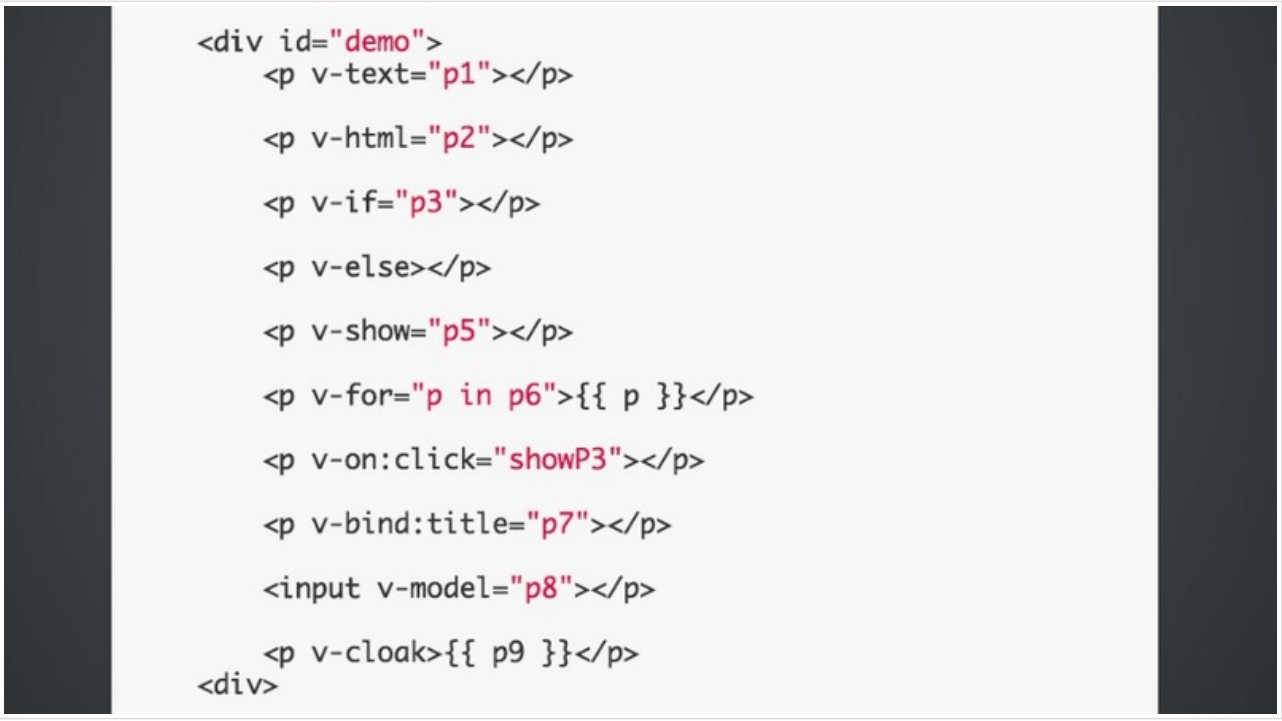
当View层的视图发生改变时，Vue会通过DOM Listeners来监听并改变Model层的数据。相反，当Model层的数据发生改变时，其也会通过Data Bingings来监听并改变View层的展示。这样便实现了一个双向数据绑定的功能，也是Vue.js数据驱动的原理所在。

Vue实例

在一个html文件中，我们直接可以通过script标签引入Vue.js，然后就可以在页面里写Vue.js代码了。上图中我们通过new Vue()构建了一个Vue的实例，在实例中，可以包含挂在元素（el），数据（data），模板（template），方法（methods）与生命周期钩子（created等）等选项。不同的实例选项拥有不同的功能，如：

（1）**el**表明我们的Vue需要操作哪一个元素下的区域，'#demo'表示操作id为demo的元素下区域。
（2）**data**表示Vue 实例的数据对象，data 的属性能够响应数据的变化。
（3）**created**表示实例生命周期中创建完成的那一步，当实例已经创建完成之后将调用其方法。

Vue常用指令



在Vue项目的开发中，我们使用的最多的应该就属Vue的指令了。通过Vue提供的常用指令，我们可以淋漓尽致地发挥Vue数据驱动的强大功能。以下便是图中常用指令的简单介绍：

（1）**v-text**: 用于更新绑定元素中的内容，类似于jQuery的text()方法
（2）**v-html**: 用于更新绑定元素中的html内容，类似于jQuery的html()方法
（3）**v-if**: 用于根据表达式的值的真假条件渲染元素，如果上图P3为false则不会渲染P标签
（4）**v-show**: 用于根据表达式的值的真假条件显示隐藏元素，切换元素的 display CSS 属性
（5）**v-for**: 用于遍历数据渲染元素或模板，如图中P6为[1,2,3]则会渲染3个P标签，内容依次为1，2，3
（6）**v-on**: 用于在元素上绑定事件，图中在P标签上绑定了showP3的点击事件

关于更多的Vue指令可以查看Vue2.0文档，地址：https://vuefe.cn/api/#指令

Vue.js技术栈

以上我们讲到可以直接在一个html页面里通过引入Vue.js来直接写Vue代码，但是这样的方式并不常用。因为如果我们的项目比较大，项目中会存在很多页面，一旦每个页面都引入一个Vue.js或者声明一个Vue实例，这样非常不利于后期的维护和代码的公用，也会存在实例名冲突的情况，所以我们需要用到Vue提供的技术栈来构建强大的前端项目。

除了Vue.js我们还需要用到：

（1）**vue-cli**：Vue的脚手架工具，用于自动生成Vue项目的目录及文件。
（2）**vue-router**： Vue提供的前端路由工具，利用其我们实现页面的路由控制，局部刷新及按需加载，构建单页应用，实现前后端分离。
（3）**vuex**：Vue提供的状态管理工具，用于同一管理我们项目中各种数据的交互和重用，存储我们需要用到数据对象。
（4）**ES6**：Javascript的新版本，ECMAScript6的简称。利用ES6我们可以简化我们的JS代码，同时利用其提供的强大功能来快速实现JS逻辑。
（5）**NPM**：node.js的包管理工具，用于同一管理我们前端项目中需要用到的包、插件、工具、命令等，便于开发和维护。
（6）**webpack**：一款强大的文件打包工具，可以将我们的前端项目文件同一打包压缩至js中，并且可以通过vue-loader等加载器实现语法转化与加载。
（7）**Babel**：一款将ES6代码转化为浏览器兼容的ES5代码的插件

利用以上等技术，我们便可以开始构建我们的Vue项目了。

构建大型应用

在构建我们的中大型Vue项目中，我们主要介绍如何利用vue-cli来自动生成我们项目的前端目录及文件，了解Vue中一切皆组件的概念及父子组件的通信问题，讲解在Vue项目中我们如何使用第三方插件，最后利用webpack来打包及部署我们的项目。

vue-cli构建

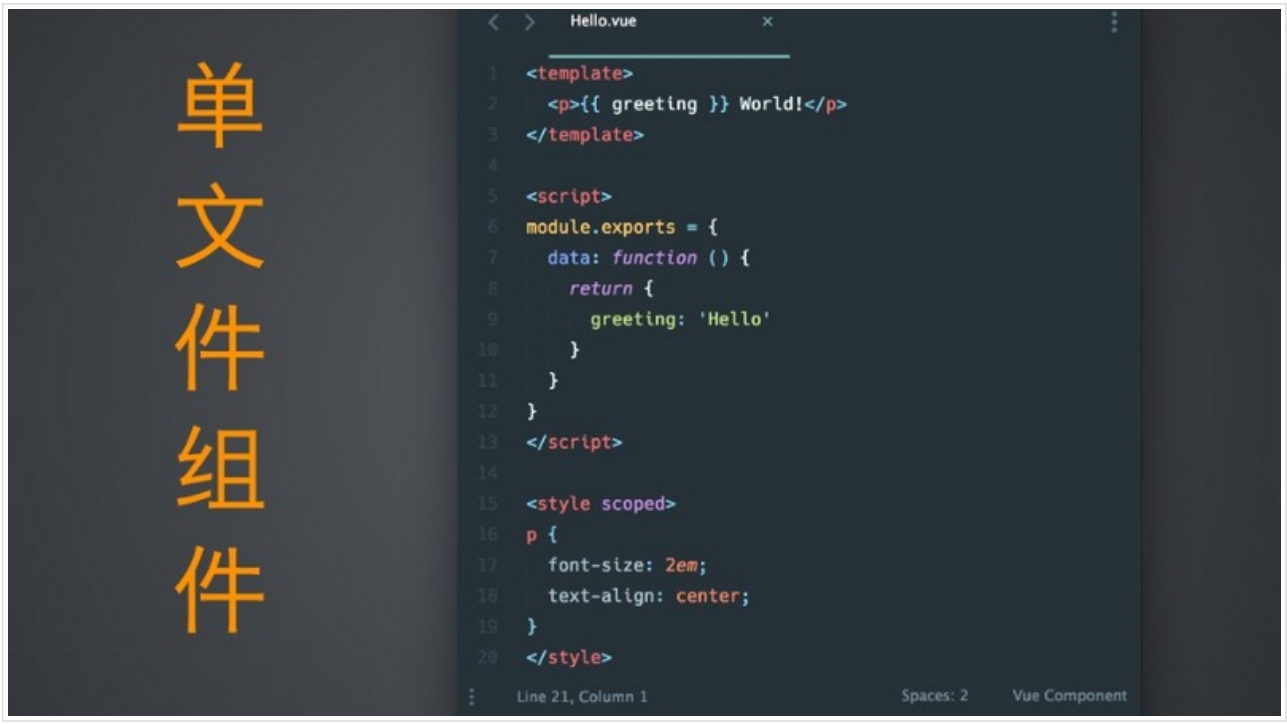
在使用vue-cli之前我们需要安装node.js，利用其提供的npm命令来安装vue-cli。安装node.js只需去其官网下载软件并安装即可，地址为：https://nodejs.org/en/

安装完成之后我们打开电脑的cmd命令行工具依次输入上图中的命令：

（1）**npm install -g vue-cli**：全局安装vue-cli
（2）**vue init webpack my-project**: 利用vue-cli在目录地址生成一个基于webpack的名为'my-project'的Vue项目文件及目录
（3）**cd my-project**：打开刚刚创建的文件夹
（4）**npm intall**：安装项目所依赖的包文件
（5）**npm run dev**：利用本地node服务器在浏览器中打开并浏览项目页面

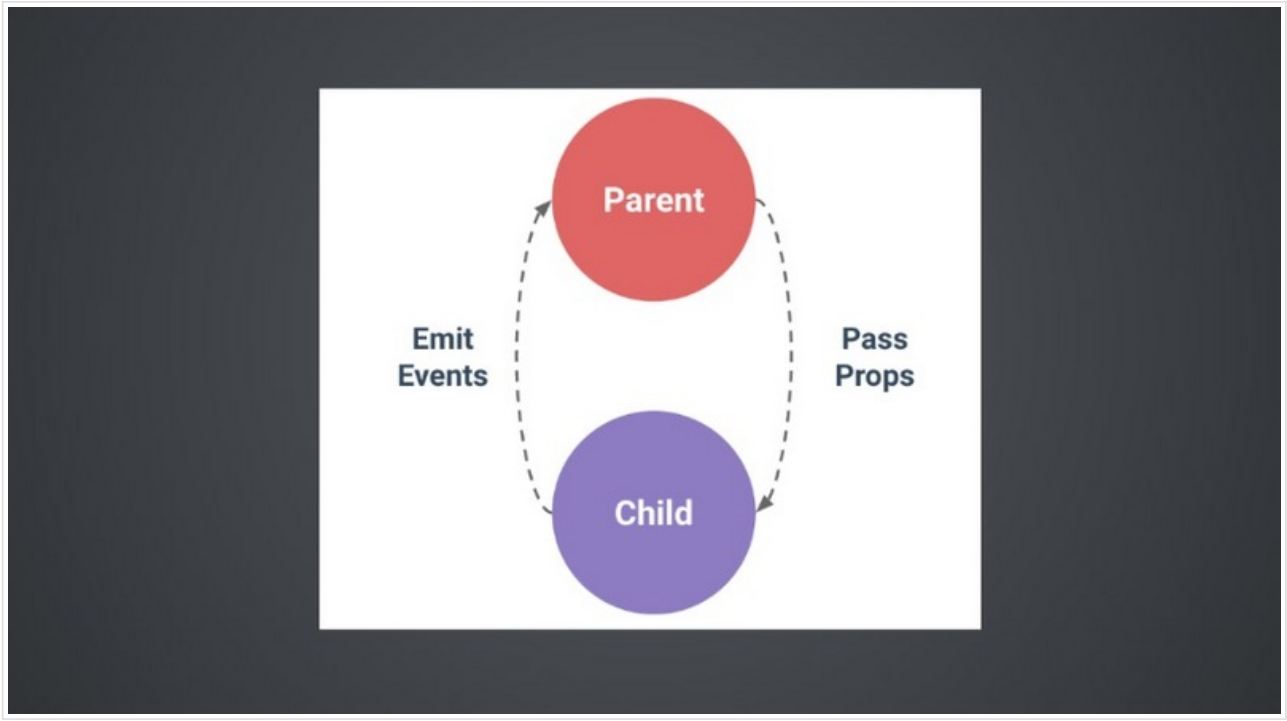
这样我们的一个基于webpack的vue项目目录就构建好了。

单文件组件



在刚刚构建好的vue项目中，我们会发现一个App.vue和Hello.vue的文件，那么像这样的以.vue后缀结尾的文件便是我们Vue项目中常见的单文件组件。单文件组件包含了一个功能或模块的html、js及css。在.vue文件中，我们可以在template标签中写html，在script标签中写js，在style标签中写css。这样一个功能或模块就是一个.vue组件，对于组件公用和后期的维护也非常便捷。

父子组件通信



那么像这样在以单文件组件为核心的项目开发中，我们一定会想到一个问题，就是.vue父子组件之间是如何交换数据来实现通信的呢？在Vue2.0中提供了props来实现父组件向子组件传递数据，通过$emit来实现子组件向父组件传递数据。当然如果是较为复杂和普遍的数据交互，建议大家使用vuex来同一管理数据。详情请见：https://vuefe.cn/guide/compon...使用Props传递数据

插件使用

接下来我们介绍下在基于webpack的vue项目中我们是如何使用插件的，主要有两种情况：

（一）全局使用

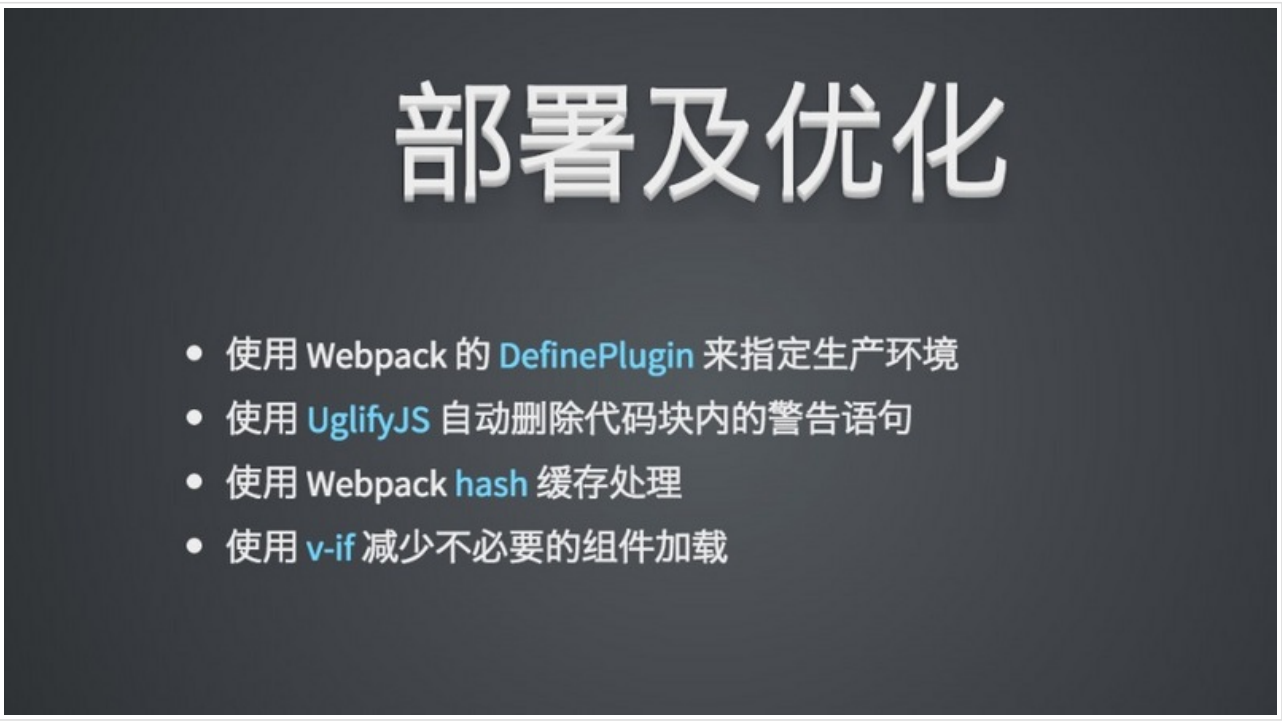
（1）**在index.html引入**：这样的方式不推荐使用，因为存在先后加载顺序的问题，有些插件不支持这一方式。
（2）**通过webpack配置文件引入**：主要通过plugin配置项的webpack.ProvidePlugin()方法实现，不过只适合支持CommonJs规范并提供一个全局变量的插件，如jQuery中的$。
（3）**通过import + Vue.use()引入**：这种方式需要在全局.vue文件中import需要加载的插件，然后通过Vue.use('插件变量名')来实现，不过此方法只支持遵循Vue.js插件编写规范的插件使用，如vue-resourece。

（二）单文件使用

（1）**通过import直接引入**：这种方式可以在需要调用插件的.vue文件中使用，不过需要注意和实例的创建顺序问题，或者也可以通过require引入。

（2）**import + components注册**：此方式为Vue组件的使用方式，可以在一个组件中注册并使用一个子组件。

部署及优化



当我们搞定整个Vue项目的前端编码阶段后，我们需要对我们的前端项目文件进行部署和优化工作，主要的几个方式如下：

（1）**使用webpack的DefinePlugin指定生产环境**：通过plugin中的DefinePlugin配置，我们可以声明'process.env'属性为'development'(开发环境)或者'production'(生产环境)，结合npm配置文件package.json中scripts的命令来切换环境模式十分方便。

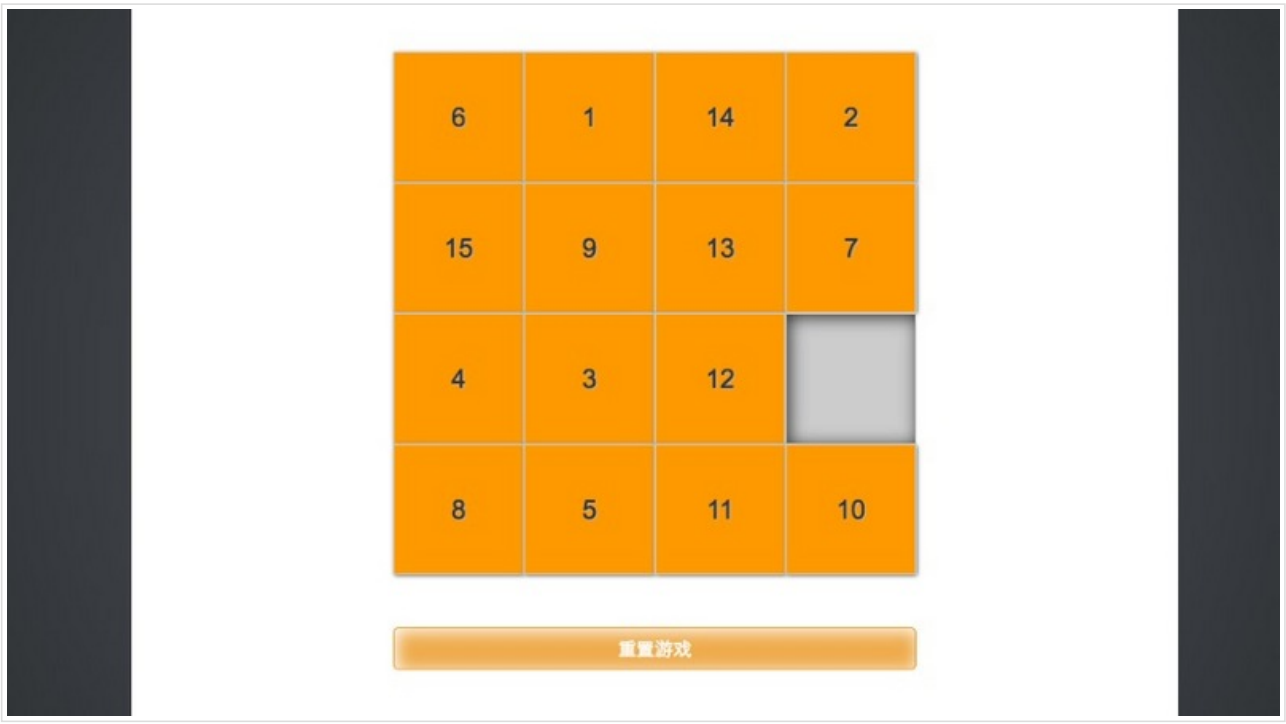
（2）**使用UglifyJs自动删除代码块内的警告语句**：一般在生产环境的webpack配置文件中使用，通过new webpack.optimize.UglifyJsPlugin()来进行配置，删除警告语句可以缩减文件的体积。

（3）**使用Webpack hash处理缓存**：当我们需要对发布到线上的文件进行修改时，重新编译的文件名如果和之前版本的相同会引起浏览器无法识别而加载缓存文件的问题。这样我们需要自动的生成带hash值的文件名来阻止缓存。详见：https://segmentfault.com/a/11...

（4）**使用v-if减少不必要的组件加载**：v-if指令其实很有用处，它可以让我们项目中暂时不需要的组件不进行渲染，等需要用到的时候在渲染，比如某个弹窗组件等。这样我们可以减少页面首次加载的时间和文件量。

除了以上几点的优化，还有很多优化选择，有兴趣的童鞋可以好好地了解下webpack的API文档，毕竟webpack的功能十分强大。

数据驱动实例



这是我之前利用Vue.js数据驱动的原理写的一个拼图游戏，希望能够供大家进一步了解Vue数据驱动的理念。
演示地址：拼图游戏
代码地址：拼图代码

总结

本文以PPT图片附加文字介绍的形式简单介绍了Vue.js的知识点及开发流程，并将前端自动化、组件化、工程化的理念贯穿其中，由浅入深地阐述了Vue.js“简单却不失优雅，小巧而不发大匠”的独特魅力。

1 天前发布　更多

你可能感兴趣的文章

Vue.js 的一些资源索引　75 收藏，9.1k 浏览
Vue.js + LeanCloud（node.js）前后端分离开发样板　31 收藏，1.9k 浏览
结合Vue.js的前端压缩图片方案　11 浏览

讨论区

使用评论询问更多信息或提出修改意见，请不要在评论里回答问题

提交评论

评论支持部分 Markdown 语法： `**bold**` `_italic_` `[link](http://example.com)` `> 引用` `` `code` `` `- 列表` 。
同时，被你 @ 的用户也会收到通知

本文隶属于专栏

前端呼啦圈

汇聚前端知识，普及前端技术。公众号：前端呼啦圈（Love-FED）

劳卜
作者

关注专栏

相关收藏夹 换一组

Vue.js
4 个条目 | 0 人关注

Vue.js
4 个条目 | 0 人关注

Vue.js
10 个条目 | 2 人关注

分享扩散：