VueJs 相关文献

Vue.js (pronounced /vjuː/, like view) is a library for building interactive web interfaces. The goal of Vue.js is to provide the benefits of reactive data binding and composable view components with an API that is as simple as possible.

Vue.js是一个构建数据驱动的 web 界面的库。Vue.js 的目标是通过尽可能简单的 API 实现响应的数据绑定和组合的视图组件。

Vue.js itself is not a full-blown framework - it is focused on the view layer only. It is therefore very easy to pick up and to integrate with other libraries or existing projects. On the other hand, when used in combination with proper tooling and supporting libraries, Vue.js is also perfectly capable of powering sophisticated Single-Page Applications.

Vue.js 自身不是一个全能框架——它只聚焦于视图层。因此它非常容易学习，非常容易与其它库或已有项目整合。另一方面，在与相关工具和支持库一起使用时，Vue.js 也能完美地驱动复杂的单页应用。

If you are an experienced frontend developer and want to know how Vue.js compares to other libraries/frameworks, check out the Comparison with Other Frameworks; if you are more interested about how Vue.js approaches larger-scale applications, check out the section on Building Larger-Scale Applications.

如果你是有经验的前端开发者，想知道 Vue.js 与其它库/框架的区别，查看对比其它框架；如果你对使用 Vue.js 开发大型应用更感兴趣，查看构建大型应用。

Reactive Data Binding

数据绑定原理

At the core of Vue.js is a reactive data-binding system that makes it extremely simple to keep your data and the DOM in sync. When using jQuery to manually manipulate the DOM, the code we write is often imperative, repetitive and error-prone. Vue.js embraces the concept of data-driven view. In plain words, it means we use special syntax in our normal HTML templates to “bind” the DOM to the underlying data. Once the bindings are created, the DOM will then be kept in sync with the data. Whenever you modify the data, the DOM updates accordingly. As a result, most of our application logic is now directly manipulating data, rather than messing around with DOM updates. This makes our code easier to write, easier to reason about and easier to maintain.

Vue.js 的核心是一个响应的数据绑定系统，它让数据与 DOM 保持同步非常简单。在使用 jQuery 手工操作 DOM 时，我们的代码常常是命令式的、重复的与易错的。Vue.js 拥抱数据驱动的视图概念。通俗地讲，它意味着我们在普通 HTML 模板中使用特殊的语法将 DOM “绑定”到底层数据。一旦创建了绑定，DOM 将与数据保持同步。每当修改了数据，DOM 便相应地更新。这样我们应用中的逻辑就几乎都是直接修改数据了，不必与 DOM 更新搅在一起。这让我们的代码更容易撰写、理解与维护。

Webpack相关文献

webpack is a module bundler for modern JavaScript applications. It is incredibly configurable, but to get started you only need to understand Four Core Concepts: entry, output, loaders, and plugins.

webpack 是一个现代的 JavaScript 应用程序的模块打包器(module bundler)。它是高度可配置的，但是，在开始前你需要先理解四个核心概念：入口(entry)、输出(output)、loader、插件(plugins)。

Webpack is a popular module bundler, a tool for bundling application source code in convenient chunks and for loading that code from a server into a browser.

Webpack是一个流行的模块绑定器，用于将应用程序源代码捆绑在方便的块中 并将该代码从服务器加载到浏览器中的工具。

Webpack is a powerful module bundler. A bundle is a JavaScript file that incorporates assets that belong together and should be served to the client in a response to a single file request. A bundle can include JavaScript, CSS styles, HTML, and almost any other kind of file.

Webpack打包机是一种强大的模块。一捆是一个JavaScript文件,包含资产属于彼此,应该在响应客户端请求一个文件。包可以包含JavaScript、CSS样式的HTML,几乎任何其他类型的文件。

Webpack roams over your application source code, looking for import statements, building a dependency graph, and emitting one or more bundles. With plugins and rules, Webpack can preprocess and minify different non-JavaScript files such as TypeScript, SASS, and LESS files.

You determine what Webpack does and how it does it with a JavaScript configuration file, webpack.config.js.

Webpack出没在你的应用程序源代码,寻找导入语句,构建一个依赖图,和释放一个或多个包。通过插件和规则,Webpack可以进行预处理和贬低打印稿等不同的javascript文件,SASS,和更少的文件。

您可以确定Webpack的功能以及JavaScript配置文件的运行方webpack.config.js。