一. 了不起的vue

1. 官方介绍 （https://cn.vuejs.org/v2/guide/）

Vue (读音 /vjuː/，类似于 view) 是一套用于构建用户界面的渐进式框架。与其它大型框架不同的是，Vue 被设计为可以自底向上逐层应用。Vue 的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合。另一方面，当与现代化的工具链以及各种支持类库结合使用时，Vue 也完全能够为复杂的单页应用提供驱动。

2. 渐进式

框架做分层设计，每层都可选，不同层可以灵活接入其他方案。而当你都想用官方的实现时，会发现也早已准备好，各层之间包括配套工具都能比接入其他方案更便捷地协同工作。

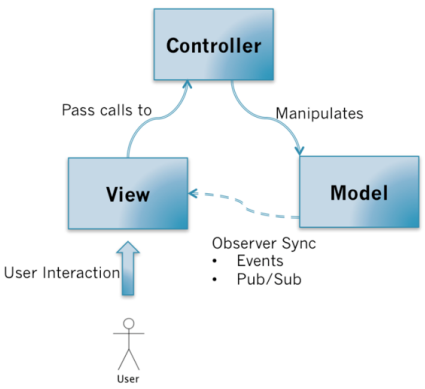
一个个放入,放多少就做多少。

3. MV*模式（MVC/MVP/MVVM）

- **"MVC": model view controller**

用户的对View操作以后，View捕获到这个操作，会把处理的权利交移给Controller（Pass calls）；Controller会对来自View数据进行预处理、决定调用哪个Model的接口；然后由Model执行相关的业务逻辑（数据请求）；当Model变更了以后，会通过**观察者模式（Observer Pattern）**通知View；View通过**观察者模式**收到Model变更的消息以后，会向Model请求最新的数据，然后重新更新界面。

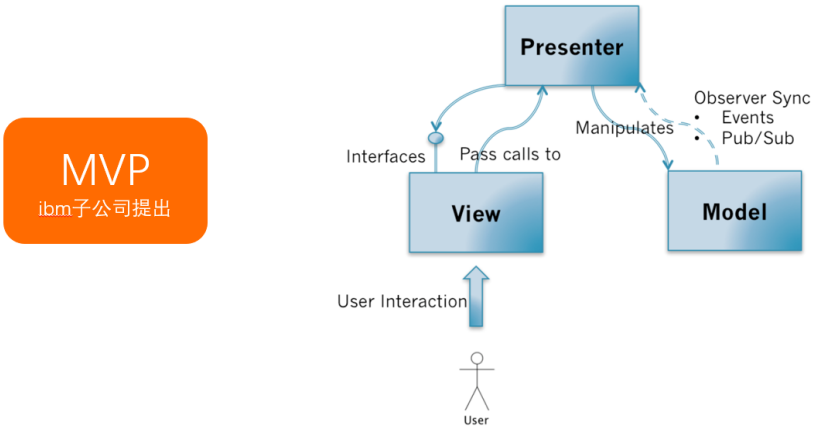
把业务逻辑和展示逻辑分离，模块化程度高。但由于View是强依赖特定的Model的，所以View无法组件化，无法复用



- **"MVP"：model view presenter**

和MVC模式一样，用户对View的操作都会从View交移给Presenter。Presenter会执行相应的应用程序逻辑，并且对Model进行相应的操作；而这时候Model执行完业务逻辑以后，也是通过观察者模式把自己变更的消息传递出去，但是是传给Presenter而不是View。Presenter获取到Model变更的消息以后，**通过View提供的接口更新界面。**

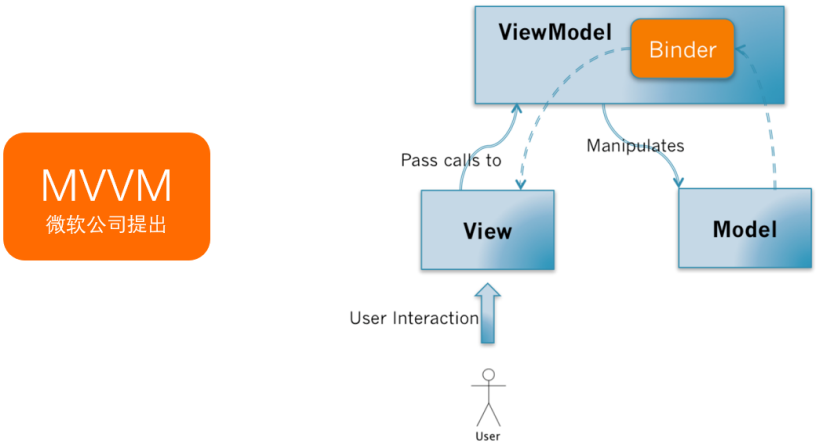
View不依赖Model，View可以进行组件化。但Model->View的手动同步逻辑　麻烦，维护困难



- **"MVVM"：model view viewmodel**

MVVM的调用关系和MVP一样。但是，在ViewModel当中会有一个叫Binder，或者是Data-binding engine的东西。你只需要在View的模版语法当中，指令式地声明View上的显示的内容是和Model的哪一块数据绑定的。当ViewModel对进行Model更新的时候，Binder会自动把数据更新到View上去，当用户对View进行操作（例如表单输入），Binder也会自动把数据更新到Model上去。这种方式称为：Two-way data-binding，双向数据绑定。**可以简单而不恰当地理解为一个模版引擎，但是会根据数据变更实时渲染。**

解决了MVP大量的手动View和Model同步的问题，提供双向绑定机制。提高了代码的可维护性。对于大型的图形应用程序，视图状态较多，ViewModel的构建和维护的成本都会比较高。



二. Vue 心跳体验

- 直接下载并用 <script> 标签引入，Vue 会被注册为一个全局变量。

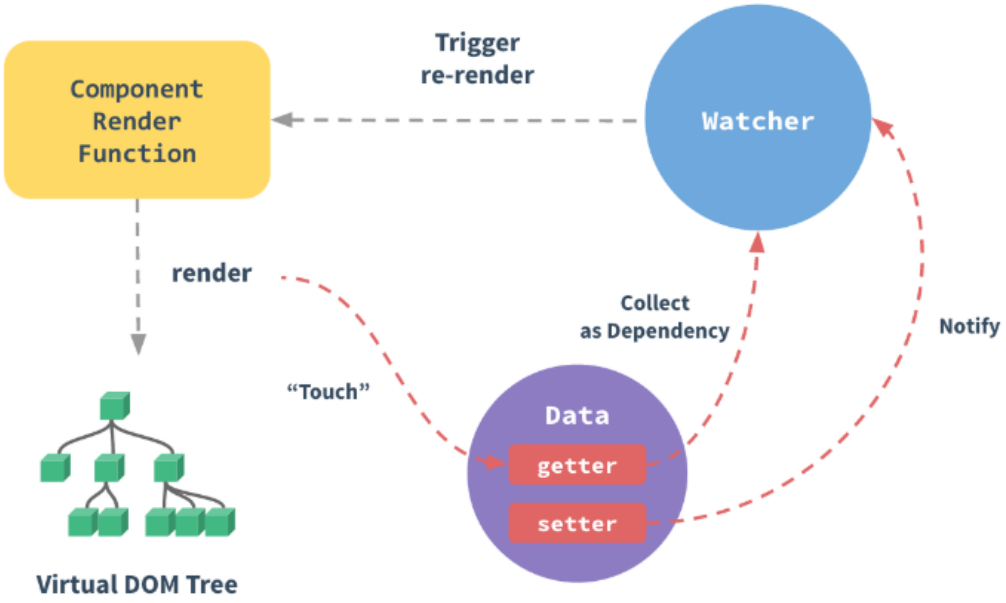
```
<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
```

- 命令行工具vue cli

  Vue 提供了一个官方的 CLI，为单页面应用 (SPA) 快速搭建繁杂的脚手架。

```
npm install -g @vue/cli
```

三. 真相只有一个-数据绑定原理



> https://cn.vuejs.org/v2/guide/reactivity.html

当你把一个普通的 JavaScript 对象传入 Vue 实例作为 data 选项，Vue 将遍历此对象所有的属性，并使用 **Object.defineProperty** 把这些属性全部转为 getter/setter。Object.defineProperty 是 ES5 中一个无法 shim 的特性，这也就是 Vue 不支持 IE8 以及更低版本浏览器的原因。

每个组件实例都对应一个 watcher 实例，它会在组件渲染的过程中把“接触”过的数据属性记录为依赖。之后当依赖项的 setter 触发时，会通知 watcher，从而使它关联的组件重新渲染。

注意：vue3 的 变化

Object.defineProperty有以下缺点。

1、无法监听es6的Set、Map 变化；

2、无法监听Class类型的数据；

3、属性的新加或者删除也无法监听；

4、数组元素的增加和删除也无法监听。

针对Object.defineProperty的缺点，ES6 Proxy都能够完美得解决，它唯一的缺点就是，对IE不友好,所以vue3在检测到如果是使用IE的情况下（没错，IE11都不支持Proxy），会自动降级为Object.defineProperty的数据监听系统。