一. 了不起的vue

1.官方介绍

Vue (读音 /vju:/，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。

1. 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。

1. 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。

四. 模板语法

1. 插值

1. 文本 {{}}
2. 纯HTML v-html
   防止XSS,CSRF

   (1) 前端过滤

   (2) 后台转义(< > < >)

   (3) 给cookie 加上属性 http

   1. click // 复制pdf需要注意编码问题
3. 表达式

1. 指令：是带有 v- 前缀的特殊属性

- v-bind 动态绑定属性
- v-if 动态创建/删除
- v-show 动态显示/隐藏
- v-on:click 绑定事件
- v-for 遍历
- v-model 双向绑定表单

1. 缩写

- v-bind:src => :src
- v-on:click => @click

2. class 与 style

2. class与style

(1)绑定HTML Class

- 对象语法
- 数组语法

(2)绑定内联样式

- 对象语法
- 数组语法

//需要将 font-size =>fontSize

3.条件渲染

(1)v-if

(2)v-else v-else-if

(3)template v-if ,包装元素template 不会被创建

(4)v-show

4.列表渲染

(1)v-for (特殊 v-for=”n in 10”)

- in
- of

//没有区别

(2)key:

- 跟踪每个节点的身份，从而重用和重新排序现有元素
- 理想的 key 值是每项都有的且唯一的 id。data.id

(3)数组更新检测

- a. 使用以下方法操作数组，可以检测变动

push() pop() shift() unshift() splice() sort() reverse()

- b. filter(), concat() 和 slice() ,map(),新数组替换旧数组
- c. 不能检测以下变动的数组

vm.items[indexOfItem] = newValue

解决

```
(1) Vue.set(example1.items, indexOfItem, newValue)
(2) splice
```

(4)应用:显示过滤结果

5. 事件处理

(1)监听事件-直接触发代码

(2)方法事件处理器-写函数名 handleClick

(3)内联处理器方法-执行函数表达式 handleClick($event) $event 事件对象

(4)事件修饰符 https://cn.vuejs.org/v2/guide/events.html

- .stop
- .prevent
- .capture
- .self
- .once
- .passive

```
每次事件产生，浏览器都会去查询一下是否有preventDefault阻止该次事

件的默认动作。我们加上**passive就是为了告诉浏览器，不用查询了，我们没用**

**preventDefault阻止默认动作。**

这里一般用在滚动监听，@scroll，@touchmove 。因为滚动监听过程中，移动每个像素都会产生一次
事件，每次都使用内核线程查询prevent会使滑动卡顿，我们通过passive将内核线程查询跳过，可以
```

事件，每次都使用内核线性查询prevent会使滑动卡顿。我们通过passive将内核线性查询跳过，可以大大提升滑动的流畅度。

(5)按键修饰符

6.表单控件绑定/双向数据绑定

v-model

(1)基本用法

- 购物车

(2)修饰符

- .lazy :失去焦点同步一次
- .number :格式化数字
- .trim : 去除首尾空格

7. 计算属性

复杂逻辑,模板难以维护

(1) 基础例子

(2) 计算缓存 VS methods

- 计算属性是基于它们的依赖进行缓存的。
- 计算属性只有在它的相关依赖发生改变时才会重新求值

8. Mixins

混入 (mixins) 是一种分发 Vue 组件中可复用功能的非常灵活的方式。

混入对象可以包含任意组件选项。

当组件使用混入对象时，所有混入对象的选项将被混入该组件本身的选项。

https://cn.vuejs.org/v2/guide/mixins.html#%E5%9F%BA%E7%A1%80

五、数据请求

五. 数据请求

1. fetch

why

- XMLHttpRequest 是一个设计粗糙的 API，配置和调用方式非常混乱，
- 而且基于事件的异步模型写起来不友好。
- 兼容性不好

polyfill

https://github.com/camsong/fetch-ie8

```
//get
fetch("**").then(res=>res.json()).then(res=>{console.log(res)})
fetch("**").then(res=>res.text()).then(res=>{console.log(res)})
//post
fetch("**",{
method:'post',
headers: {
"Content-Type": "application/x-www-form-urlencoded"
},
body: "name=kerwin&age=100"
}).then(res=>res.json()).then(res=>{console.log(res)});
fetch("/users",{

method:'post',
// credentials: 'include',
headers: {
"Content-Type": "application/json"
},
body: JSON.stringify({
name:"kerwin",
age:100
})
}).then(res=>res.json()).then(res=>{console.log(res)});
```

注意

1. Fetch 请求默认是不带 cookie 的，需要设置 fetch(url, {credentials: 'include'})

2. axios

```
axios.get("")
axios.post("")
axios.put("")
axios.delete("")

axios({
url:"",
headers:{
'X-Client-Info': '{"a":"3000","ch":"1002","v":"1.0.0","e":"1"}',
'X-Host': 'mall.cfg.common-banner'
}
}).then(res=>{
console.log(res.data);
})

返回的数据会被包装

{
*:*
data:真实后端数据
}
```

六. 组件

1. 虚拟dom与diff算法 key的作用

liupeijun1988@163.com\bb98034c03b74a3d8f19a06e4a3708b3\clipboard.png" alt="img">

（1）把树按照层级分解

liupeijun1988@163.com\419390c1d60a47aaaa695db077e3a1d2\clipboard.png" alt="img">

(2) 同key值对比

liupeijun1988@163.com\137f7d19e2494a95bcf2cd36a515aaa4\clipboard.png" alt="img">

(3) 同组件对比

liupeijun1988@163.com\69da1f74c3db4ec6b6b83f66206b17e1\clipboard.png" alt="img">

2. 为什么组件化

扩展 HTML 元素，封装可重用的代码

3. 组件注册方式

- a.全局组件

liupeijun1988@163.com\75d1f9cec4794e9494910cf22122abf5\clipboard.png" alt="img">

- b.局部组件

liupeijun1988@163.com\89ea9db58812463f97f93e2f04c34015\clipboard.png" alt="img">

4. 组件编写方式与Vue实例的区别

*自定义组件需要有一个root element

*父子组件的data是无法共享

*组件可以有data，methods,computed….,但是data 必须是一个函数

5. 组件通信

i. 父子组件传值 (props down, events up)

ii. 属性验证

props:{name:Number} Number,String,Boolean,Array,Object,Function,null(不限制类型)

iii. 事件机制

a.使用 $on(eventName) 监听事件

b.使用 $emit(eventName) 触发事件

iv. Ref

this.$refs.mytext

v. 事件总线

var bus = new Vue();

* mounted生命周期中进行监听

6. 动态组件

- <component> 元素，动态地绑定多个组件到它的 is 属性
- <keep-alive> 保留状态，避免重新渲染

7. slot插槽（内容分发）

- 混合父组件的内容与子组件自己的模板—>内容分发
- 父组件模板的内容在父组件作用域内编译；子组件模板的内容在子组件作用域内编译。

  a. 单个slot

  b. 具名slot

liupeijun1988@163.com\fe99c38d952b4766a5b5bf18695a412d\clipboard.png" alt="img">

注意 v-slot 只能添加在 template 上，文本节点也可以当具名插槽内容插入

8. transition过渡

Vue 在插入、更新或者移除 DOM 时，提供多种不同方式的应用过渡效果。

(1)单元素/组件过渡

* css过渡

* css动画

* 结合animate.css动画库

liupeijun1988@163.com\956f7b793fdd46448e755d355210152b\clipboard.png" alt="img">

(2) 多个元素过渡(设置key)

*当有相同标签名的元素切换时，需要通过 key 特性设置唯一的值来标记以让 Vue 区分它们，否则 Vue 为了效率只会替换相同标签内部的内容。

mode:in-out ; out-in

(3)多个组件过渡

(4)列表过渡(设置key)

*<transition-group>不同于 transition， 它会以一个真实元素呈现：默认为一个 。你也可以通过 tag 特性更换为其他元素。

* 提供唯一的 key 属性值

9. 生命周期

i. 生命周期各个阶段

https://cn.vuejs.org/v2/guide/instance.html#%E7%94%9F%E5%91%BD%E5%91%A8%E6%9C%9F%E5%9B%BE%E7%A4%BA

ii. 生命周期钩子函数的触发条件与作用

liupeijun1988@163.com\8e97398ea3ea4d3d8ffd01536a006f61\lifecycle.png" alt="img">

10. swiper学习

https://www.swiper.com.cn/

11. 自定义组件的封装

自定义封装swiper组件（基于swiper）

注意： 防止swipe初始化过早

七. 指令

1 自定义指令

1. 自定义指令

(1)自定义指令介绍 directives - 对普通 DOM 元素进行底层操作

liupeijun1988@163.com\d5dfdfd222574c11be96ba4cb7145a8e\clipboard.png" alt="img">

(2)钩子函数

* 参数 el,binding,vnode,oldvnode

* bind,inserted,update,componentUpdated,unbind

liupeijun1988@163.com\7e5e65c19dd941a29695022a0d65112b\clipboard.png" alt="img">

(3)函数简写

(4)自定义指令-轮播

- inserted 插入最后一个元素时初始化swiper

2. nextTick

liupeijun1988@163.com\d2d2762e336a4f789f36d073a065cbac\clipboard.png" alt="img">

- this.$nextTick()

八.过滤器

https://cn.vuejs.org/v2/guide/filters.html

liupeijun1988@163.com\808563af15b041f28ac06c85333a06cb\clipboard.png" alt="img">

全局写法

liupeijun1988@163.com\3b4424fa8ba346739e45d74165d22ee9\clipboard.png" alt="img">

局部写法

liupeijun1988@163.com\14dab0a7ffb146858262ba44640c75e7\clipboard.png" alt="img">

可以串联

liupeijun1988@163.com\d4bfa56294a043588dd385bc5efaae29\clipboard.png" alt="img">

过滤器是 JavaScript 函数，因此可以接收参数

liupeijun1988@163.com\6328519ad02543f2b9fdea48d5a39ab4\clipboard.png" alt="img">

这里，filterA 被定义为接收三个参数的过滤器函数。其中 message 的值作为第一个参数，普通字符串 ‘arg1’ 作为第二个参数，表达式 arg2 的值作为第三个参数。

九. 单文件组件

1. 写法

https://cn.vuejs.org/v2/guide/single-file-components.html

```
a.
 <template>
 html代码
 </template>
 <script>
 js代码
 </script>
 <style>
 css代码
 </style>
b.
 <template>
 html代码
 </template>
 <script src="相对路径的外部的js"></script>
 <style src="相对路径的外部的css"></style>
```

2. vue-cli3.0的使用

npm install -g @vue/cli (一次安装)

vue create myapp

*npm run serve 开发环境构建

*npm run build 生产环境构建

*npm run lint 代码检测工具

style标签 加上scoped属性，css局部生效 style标签 加上lang=”scss”，支持scss

3. Vue.config.js的配置

(1) proxy代理

https://cli.vuejs.org/zh/config/#%E5%85%A8%E5%B1%80-cli-%E9%85%8D%E7%BD%AE

```
devServer: {
port:8000, //随便改端口号
proxy: {
'/api': {
target: 'https://*.*.com',
host: '*.*.com',
changeOrigin:true
}
}
}
```

(2) alias别名配置

@ is an alias to /src

(3) vue.config.js 中配置 publicPath: ‘./ ‘

(4)关闭eslint

vue.config.js lintOnSave: false

.eslintrc 删除 ‘@vue/standard’ （对于某个规则关闭， no-new:”off” ）

liupeijun1988@163.com\94c0c3d557ad4638acaf6fcb4768dd28\clipboard.png" alt="img">

4. 利用vue cli进行组件化开发

迁移todolist、swiper案例到vue cli中

十. 路由开发

1. SPA概念

1. SPA概念

liupeijun1988@163.com\0c6746b048a84d3ba01172d1e30f3c5f\1013869-20180413214226425-113139192.png" alt="img">

2. vue-router

- 开始

liupeijun1988@163.com\938ef24fee454a07a7b6e6a14c2516d0\clipboard.png" alt="img">

- 动态路由匹配
- 嵌套路由
- 编程式导航（js跳转） vs 声明式导航<router-link>
- 命名路由
- 重定向和别名

liupeijun1988@163.com\b10203dd25754493a1f22e5e4135cc64\clipboard.png" alt="img">

liupeijun1988@163.com\a4a04aecc36342498075a5bfea23f255\clipboard.png" alt="img">

- HTML5 History模式

vue支持两种模式

a. hash #/home

b. history /home

- 路由守卫&路由拦截
- 全局拦截
- 单个拦截
- 路由懒加载

liupeijun1988@163.com\88deb2001ae44c3e822d16157bdf4202\clipboard.png" alt="img">

3. 路由原理

(1)hash路由 ==> location.hash 切换

window.onhashchange 监听路径的切换

(2)history路由==> history.pushState 切换

window.onpopstate 监听路径的切换

4. 项目

(1) 启动案例项目开发

(2) 利用vue-router搭建项目SPA结构

十一. 状态管理 Vuex

1. Vuex使用

Vuex是一个专为 Vue.js 应用程序开发的状态管理模式。它采用集中式存储管理应用的所有组件的状态，并以相应的规则保证状态以一种可预测的方式发生变化.

liupeijun1988@163.com\c03e3f15711941ff87e1fc4e365e8ce6\clipboard.png" alt="img">

(1)state:单一状态树 ,每个应用将仅仅包含一个 store 实例。

*this.$store.state.状态名字

*…mapState([“title”])

(2)getters:可以从store 中的 state 中派生出一些状态，getters的返回值会根据它的依赖被缓存起来，且只有当它的依赖值发生了改变才会被重新计算。

*可以认为是 store 的计算属性

*this.$store.getters.计算属性名字

*…mapGetters([“getFilms”])

(3)mutations：更改 Vuex 的 store 中的状态的唯一方法是提交 mutation。

*常量的设计风格

[SOME_MUTATION] (state) {

// mutate state

}

*必须是同步函数

*this.$store.commit(“type”,”payload”);

(4)actions:

*Action 提交的是 mutation，而不是直接变更状态。

*Action 可以包含任意异步操作。

*this.$store.dispatch(“type”,”payload”)

(5)

```
const store = new Vuex.Store({
state: {
count: 0
},
mutations: {
increment (state ,payload) {

}
},
actions: {
increment (context, payload) {
context.commit('increment')
}
}
})
```

(6) 模块分割

liupeijun1988@163.com\71557e32d2cb45d6b68f2077fe5bf06a\clipboard.png" alt="img">

liupeijun1988@163.com\3f7c26dd1136417182cfa8dd2c67756f\clipboard.png" alt="img">

2. 注意

(1)应用层级的状态应该集中到单个 store 对象中。

(2)提交 mutation 是更改状态的唯一方法，并且这个过程是同步的。

(3)异步逻辑都应该封装到 action 里面。

3. vue chrome devtools

liupeijun1988@163.com\bf6c53ecbae947f5809cc8b9ebd9b5df\clipboard.png" alt="img">

4. vuex在项目中的使用

1. 1. 复杂非父子通信**

1. 异步数据快照**

5. vuex持久化

https://github.com/robinvdvleuten/vuex-persistedstate

liupeijun1988@163.com\768b0048e85445418dc6b1ae0e889ffe\clipboard.png" alt="img">

十二. 组件库

1. 使用第三方插件

https://github.com/vuejs/awesome-vue#components--libraries

集合了来自社区贡献的数以千计的插件和库。

2. 使用第三方UI框架

1.饿了么UED团队推出的vue 前端框架

(1) PC框架：(element UI)

http://element.eleme.io/#/

https://github.com/ElemeFE/element

- 从0开始的话，可以通过以下命令

liupeijun1988@163.com\22bb0f6031644b3188ad938dc3b98458\clipboard.png" alt="img">

- 项目已经写了， 通过 cnpm i —save element-ui

liupeijun1988@163.com\7c706fedf1804a219d73f6d955279209\clipboard.png" alt="img">

(2) 移动端框架：(mint UI) 好久不更新维护了

https://mint-ui.github.io/docs/#/

https://github.com/ElemeFE/mint-ui

2. 有赞技术团队推出的vue移动端框架

移动端框架：(vant)

https://youzan.github.io/vant/#/zh-CN/

(1) 安装

liupeijun1988@163.com\ead41f8c36c04797b870ab07d9743df6\clipboard.png" alt="img">

liupeijun1988@163.com\3372611b50244ef88f050993c532456d\clipboard.png" alt="img">

(2) 正在加载

liupeijun1988@163.com\91dadaabe2c342b2b1bfc839d3f79dbd\clipboard.png" alt="img">

（3）无限滚动

liupeijun1988@163.com\19476b0cd4904c3a996791ffcc3ca63e\clipboard.png" alt="img">

(4) IndexBar 索引栏

liupeijun1988@163.com\e53d6e4ef6b94d0096cabbfa83c86bff\clipboard.png" alt="img">

十三. 项目实战

1. betterScroll

主要完成的功能需要包含Better-Scroll实现页面中拖动滚动、拉

动属性等功能

https://ustbhuangyi.github.io/better-scroll/doc/zh-hans/

(1) 初始化

```
//html
 <div class="kerwin" >
 <ul >
 <li v-for="item in datalist">{{item}}</li>
 </ul>
 <div class="loading-wrapper"></div>
 </div>

//js
import BScroll from 'better-scroll'
this.$nextTick(()=>{
 var myscroll = new BScroll('.kerwin',
 {
 pullDownRefresh: {
 threshold: 50,
 stop: 20
 },
 scrollbar: {
 fade: true,
 interactive: false // 1.8.0 新增
 },
 pullUpLoad: {
 threshold: 50
 }
 })
})

//css
.kerwin{
 height: 300px; //设置高度
 overflow:hidden; //溢出隐藏
 position: relative; //修正滚动条位置
 }
```

(2) 下拉刷新

```
myscroll.on('pullingDown',()=>{
 console.log("下拉了")
 setTimeout(() => {
 myscroll.finishPullDown() // 自动调用 .refresh()
 }, 1000)
 })
```

(3) 上拉加载

```
 myscroll.on('pullingUp',()=>{
 console.log("到底了")
 setTimeout(() => {
 myscroll.finishPullUp() // 自动调用 .refresh()
 }, 1000)
 })
```

2. 移动端事件相关

(1)click事件300ms延迟

liupeijun1988@163.com\1f89510253a949449efee2ba17795bbc\clipboard.png" alt="img">

解决:

- 设置meta viewport
- fastclick

liupeijun1988@163.com\277a70b144db4de2b7e1dc516bed5a0d\clipboard.png" alt="img">

(2)Hammer.js

HammerJS是一个优秀的、轻量级的触屏设备手势库

- hammer.js https://github.com/hammerjs/hammer.js

liupeijun1988@163.com\199b1bab21b64e95bafcc2163283608b\clipboard.png" alt="img">

- vue touch https://github.com/vuejs/vue-touch/tree/next

liupeijun1988@163.com\70558793daed49e0ae670bc70acb8893\clipboard.png" alt="img">

3. Git 复习

(1). 复习git的基本使用

git pull 拉取并merge代码

git add .; //添加 暂存区

git commit -m ‘zhu shi’// 提交到本地仓库并加上注释

git push origin master //往远程仓库推送代码

(2). 多人协作方式

a) 分支的构建

git branch -a 查看所有的分支

git checkout -b aaa 创建新的分支aaa

git checkout aaa 切换到aaa分支

git push origin aaa 推送aaa 分支到远程仓库aaa分支

git push origin master:aaa 推送master 到远程的aaa 分支

git branch -d ** 删除一个分支

b) 冲突的产生与解决

两人同时修改同一个文件，一个人上传远程仓库成功， 另一个人再上传会失败。

（1）git pull, （拉取服务器的代码， 会造成自动合并失败，需要手动合并）

1. (2) 手动合并代码（小乌龟等可视化git工具 进行代码对比）

4.经典web服务器nginx介绍

Nginx是一款自由的、开源的、高性能的HTTP服务器和反向代理服务器；同时也是一个IMAP、POP3、SMTP代理服务器；Nginx可以作为一个HTTP服务器进行网站的发布处理，另外Nginx可以作为反向代理进行负载均衡的实现。

（1）正向代理

正向代理最大的特点是客户端非常明确要访问的服务器地址；服务器只清楚请求来自哪个代理服务器，而不清楚来自哪个具体的客户端；正向代理模式屏蔽或者隐藏了真实客户端信息。

liupeijun1988@163.com\857ec6bUc1e146be8c6dcbe58e974c00\clipboard.png alt= img >

(2) 反向代理

客户端是无感知代理的存在的，反向代理对外都是透明的，访问者并不知道自己访问的是一个代理。因为客户端不需要任何配置就可以访问。

反向代理，”它代理的是服务端”，主要用于服务器集群分布式部署的情况下，反向代理隐藏了服务器的信息。

liupeijun1988@163.com\5044a3f4f10c4694ba77f24c0147feb7\clipboard.png" alt="img">

(3) nginx的基础配置（代理等）

```
  nginx -c kerwin.conf 加载kerwin.conf 并启动服务器

  nginx -s
  {
  stop — fast shutdown
  reload — reloading the configuration file(每次修改完kerwin.conf后 ，都通过此方法 重新加载一次)

  }
 静态文件serve

 location / {
  root html;//是当前文件夹下有个html文件夹
  index index.html index.html
  }

 location /frontend {
  root html ;
  #只要加载localhost/frontend路径 那么就会从 html/frontend/路径提供文件服务
  }
 下面四种情况分别用http://localhost/proxy/test.html 进行访问。
// 注意 /proxy/ 后面的/ 需要加上

 (1) location /proxy/ {
  proxy_pass http://127.0.0.1:8000/;
  }

  会被代理到http://127.0.0.1:8000/test.html 这个url

```

```
(2)相对于第一种，最后少一个 /
location /proxy/ {
proxy_pass http://127.0.0.1:8000;
}
会被代理到http://127.0.0.1:8000/proxy/test.html 这个url

(3)location /proxy/ {
proxy_pass http://127.0.0.1:8000/xiaoming/;
}
会被代理到http://127.0.0.1:8000/xiaoming/test.html 这个url。

(4)相对于第三种，最后少一个 / ：

location /proxy/ {
proxy_pass http://127.0.0.1:8000/xiaoming;
}
会被代理到http://127.0.0.1:8000/xiaomingtest.html 这个url
```

(4). 将nginx部署在线上作为webserver

serve 静态资源， 并反向代理node服务