正课:

1. 自定义指令——了解

2. Axios

3. \*\*\*\*\*组件化开发

1. 自定义指令:

如何:

1. 创建指令

Vue.directive("指令名",{

inserted(elem){//指令所在的元素被加载到DOM树上后自动执行指令

//elem会自动获得指令所在的当前元素

对elem执行DOM操作！

}

})

2. 使用指令: <ANY v-指令名>

2. Axios  
 什么是: 专门发送http请求的函数库

为什么: ajax

1. 自己封装函数:

2. jQuery: $.ajax() 小题大做

3. Vue有一个组件resource，已不推荐使用

4. Axios 官方推荐

何时: 只要在vue中发送ajax请求都用axios

如何:

1. 引入: axios.min.js

2. 发送2种请求:

get请求:

axios.get("url",{

params:{

参数:值

}

}).then(res=>{

获得服务端返回的结果: res.data

})

post请求:

axios.post("url","变量=值&变量=值&...").then(res=>{...})

3. 组件化开发:

什么是组件: 拥有专属的HTML，CSS，JS和数据的页面独立区域

什么是组件化: 今后所有页面都是由多个组件组成的

每定义一个网页，都要先划分组件结构，再分别编写每个组件

为什么:

1. 重用

2. 便于大项目的分工协作

3. 松散耦合

如何:

1. 创建组件:

每个组件包含三部分:

HTML: 独立的HTML片段:

<template id="tplxxx"></template>

<template>是HTML5中新增的专门保存一段隐藏的HTML片段的元素——专门为框架保存HTML模板

JS: <script>

Vue.component("组件名",{

//和new Vue()极其相似

template:"#tplxxx", //代替了el，每创建一个组件的副本，就自动复制一次模板中的html片段

data:function(){//每创建一个组件的副本，就调用一次data()，为当前组件创建专属的模型数据data  
 return {

count:1

}

},

//其余和new Vue()完全一样

})

CSS: <style> 暂时不关心

2. 使用组件:

组件名其实就是一个标签名

只要在主内容中，添加一个<组件名></组件名>，运行时Vue就会用组件的template代替<组件名>标签所在位置

组件分类:

1. new Vue({ ——根组件

el:"#app",

data:{}

})

2. Vue.component("组件名",{——全局组件: 可在任何位置随意使用的组件

template:"#tplxxx",

data:function(){ return {} }

})

3. 子组件: 仅限于在某个父组件内才能使用的组件

如何: 3步:

1. 只要将Vue.component降级为普通对象

强调: 变量名最好是将来组件标签名的驼峰方式

2. 在父组件中添加components属性: {

子组件变量名, 子组件变量名, //ES6简写

}

Vue会将驼峰命名的组件名，转为小写，并用-分隔

比如: todoAdd -> todo-add

todoList -> todo-list

组件间传递数据:

父子间:

1. 父->子:

为什么: Vue中父组件的数据模型是专属的，子组件默认不能使用父组件的数据

解决: 2步:

1. 子组件:

var 子组件={

template:"xxx",

props:["变量", ... ] //声明一个内外两用的变量

//对外: 父元素可为其绑定数据

//对内: 相当于data:{ 变量 }

}

<template id="tpl子组件">

{{变量}} <ANY :属性="变量"

</template>

2. 父组件:

<template id="tpl父组件">

<子组件 :变量="模型变量"></子组件>

父组件将自己的模型变量的值传递给子组件的变量属性，子组件就获得了父组件的数据

2. 子->父:

1. 父组件: 在子组件身上提前定义自定义事件和处理函数:

<template id="tpl父组件">

<子组件 @自定义事件="父的处理函数"//不要加()

//当有人触发自定义事件时，自动执行父组件中的处理函数，修改父组件自己的数据

父组件{

...

methods:{

处理函数(形参) {//形参会收到子传来的参数  
 //操作父组件中的数据

}

}

}

2. 子组件:

任何时候都可以:

this.$emit("父给子定义的事件名", 实参)

意为触发父给子提前定义的事件，并自动引发执行父的处理函数。