**今天学习的内容**

**1:复习昨天知识重点**

**2:作业**

**3:今天的目标**

**3.1:自定义指令**

**Vue 自带很多指令; v-for/v-else/v-model...**

**但是这些指令都是比较偏向于工具化，有些时候在实现具体业务时，发现**

**不够用，如何来自定义指令**

**3.2:语法**

**new Vue({**

**...**

**directives:{**

**change:{**

**bind:function(){},**

**update:function(){},**

**unbind:function(){}**

**}**

**}**

**});**

**在自定义指令时:3个处理函数**

**bind: 自定义指令绑定元素时执行函数(1)**

**update: 如果调用指令传入参数，参数发生变化(n)**

**unbind: 解除绑定(1)**

**示例:**

**directives:{**

**hello:{**

**bind:function(el,bings...){}**

**update:function(){}**

**unbind:function(){}**

**}**

**}**

**<h3 v-hello="count" >**

**注意事项:**

**建议在给指令命名小驼峰命名方式:比如 changeBackgroundColor**

**应用:v-change-background-color (烤串式书写法)**

**练习:实现一个自定义指令 10:55--11:05**

**应用 <h3 v-change-background-color="myBgcolor"></h3>**

**js changeBacgroundColor**

**在调用指令时，可以传递参数,根据参数值**

**修改调用该指令元素背景色**

**3.3:自定义的过滤器的创建和使用**

**过滤器:生活中：净水器，空气净化器**

**过滤器:实现数据筛选，过滤，格式化**

**vue 1.\* 内置很多过滤器 但是在vue2 所有没有，自定义过滤器**

**3.4:如何创建使用过滤器**

**new Vue({**

**filters:{**

**myCurency:function(myInput){**

**//处理数据**

**return 处理后的数据;**

**}**

***}***

**});**

**#使用**

**<any>{{表达式|过滤器}}<any>**

**示例**

**<h1>{{price|myCurency}}</h1>**

**练习:实现一个过滤器 myTextTransform**

**功能实现将一个字符串转换大写或者小写**

**取决于传递过滤器参数 true 转大写**

**false 转小写**

**3.5:组件与组件之间数据传递(重点-->难点)**

**3.6:组件(重点-->难点)**

**现在前端领域，对代码质量和数据非常高的要求，特别容易出问题**

**通过组件的方式来完成代码的管理编写**

**组件:是一个可以被反复使用的，带有指定的功能视图(View)**

**生活场景:类似"组件" "V8"**

**3.7:创建组件**

**(1)创建组件**

**Vue.component("组件名称",{组件内容,template:'<h3>模块</h3>'});**

**示例:**

**Vue.component("my-component",{**

**template:'<h3>模块</h3>'}**

**);**

**(2)组件的使用:组件的使用就像一个普通html标签一样**

**<my-component></my-component>**

**注意事项:**

**(1)组件的命名和使用建议使用烤串命名规则**

**(2)如果一个组件中渲染多个元素，将多个元素放到一个根标签，否则报错**

**3.8:创建复合组件**

**复合组件并不是一个新概念，就是一个组件，只不过该组件又调其它组件。**

**一个完整VUE项目，可以比做一颗组件树：根组件->层次组件->登录->注册->用户列表.....**

**练习:使用复合组件完成表单**

**label-name input-name**

**label-pwd input-pwd**

**button-login button-register**

**表单复合组件**

**3.9:组件的生命周期(\*\*\*\*\*)**

**分为4个阶段**

**create/mount(挂载)/update/destroy**

**每个阶段都有对应处理函数**

**create:beforeCreate created(1) {创建:初始化操作}**

**mount:beforeMount mounted(1) {挂载DOM树}**

**update:beforeUpdate updated(n) {数据更新->操作或逻辑判断}**

**destroy:beforeDestroy destroyed(1) {清理工作}**

**小结：Vue实例或者组件：都有哪些属性**

**el/data/methods/directives/filters/watch (监听属性)**

**16:26--16:36**

**练习一:创建一个组件，该组件渲染一个h1标签**

**要求:在组件挂载完之后，启动定时器，修改**

**h1标签的透明度值 0-1**

**<h1 v-bind:style="{opacity:opacityValue}">透明度</h1>**

**练习二:创建一个组件:my-game**

**猜数字大小**

**组件:一个input(幸运数字)(中奖数字)和一个p元素构成**

**当组件准备挂载时候，初始化一个中奖数:**

**在input取输入的时候，如果用户输入的数字小,**

**在p显示，输入的太小**

**如果输入数字大了，在p显示:输入太大，否则提示输入正确**

**练习三:创建一个组件:my-cal**

**{+-\*/}计算器**

**组件:一个input(num1) select(opt+-\*/) input(num2)/button**

**和一个p元素构成**

**当组件准备挂载时候，初始化一个num1/num2**

**点击button按钮时计算结果并且将结果显示在p显示**

**3.10:组件之间的通信1 父组件--->子组件**

**子组件-->父组件**

**兄弟组件**

**步骤:**

**(1)父组件在调用子组件传值**

**<child myValue="123"></child>**

**(2)子组件中获取父组件传来值**

**Vue.compoent("child",{**

**props:["myValue"], //声明父组件变量，在我这里可以使用**

**template:''**

**});**