2022年6月21日 日报 day02

1.模板语法

模板的作用域：模板拥有自己的作用域，只能使用 data 传入的数据以及模板定义文件中定义的 模块。

注：不能在引用模板时 在里面定义view

- 方式一：使用局部模板 全局在页面中无法创建

  使用局部模板 并传入模板所需的数据

  技巧： es6 语法 ...：扩展运算符 拆包

```
<template is='temp1' data="{{...obj}}"></template>
```

- 方式二：使用全局模板

  1.导入 import src（绝对或相对）

  2.使用 template is

```
<import src="/template/tem2"></import>
<template is='tem2'></template>
```

注：src 填写资源路径 1.本地路径 2.网络路径 路径：相对、绝对路径

- 应用文件有两种方式

```
1.import src :只能引用template模板里的内容 有自己的作用域 不能深层次的调用 （A import B，B import C,A 不会 import到 C）
2.include src : 可以引用目标文件中非template和wxs外的所有代码 ，相当于拷贝 ，可以深层次的调用
```

2.wxs 的页面赋值问题

读取数据可以采用 this.data.num ，但赋值尽量不要采用这种方式， 因为this.data.num形式赋值，它只是改变了数据层,没有改变视图层。

```
1.通过this.setData({对象}) 去同时更新视图层和数据层 频繁的使用setData会引起视图的阻塞。
2.如果是赋值非视图层的数据 不需要放在data对象中 而使用 自定义数据的方式
this.myData=。。。或者 const 常量的方式
3. 只有和视图层关系密切的数据才需要放在data中
```

3.模块化的语法

模块化：将某些功能，封装成一个独立的js文件，以模块的形式去使用

- **注：**

  模块导入导出的语法名称叫做：commonjs

  模块 放在utils中

  导出：js模块中，想要提供给其他js文件使用的内用，可以使用语句导出

  导入：在其他的js模块中，引入第三方的模块文件

- **语法**

  导出的语法：

  module.exports=something

  module.exports.property(属性)= something

  导入的语法:

  const a=require("路径名")

```
var obj=require("../../utils/demo")
console.log(obj)
```

4.wxml中常用的组件

```
<!--pages/day2-4/day2-4.wxml-->
<text>wxml中常用的组件</text>

<view>视图容器</view>
<!-- view视图容器，类似于html中div的块级元素 独占一行

-->

<text>文体容器</text>
<text>文体容器</text>
<!-- text 组件,只能嵌套text 行内元素 不会独占一行 一行排列放不下才换行-->

<button>按钮组件</button>
```

```
<!-- 表单组件中的一种 按钮组件 -->

<navigator>导航组件</navigator>
<!-- 页面中的导航组件 类似于 html中的a标签 -->

<image src="/static/images/主页.png">图片组件</image>
<!-- 图片组件 支持 JPG、PNG、SVG、WEBP、GIF 等格式-->

<input type="text" placeholder="输入框组件"/>
<!-- 输入框。该组件是原生组件 -->
```

5.绑定事件 事件函数调用

5.1事件:

事件是视图层到逻辑层的通讯方式。

事件可以将用户的行为反馈到逻辑层进行处理。

事件可以绑定在组件上，当达到触发事件，就会执行逻辑层中对应的事件处理函数。

事件对象可以携带额外信息，如 id, dataset, touches

事件三要素：事件源、事件类型、事件处理函数

5.2事件对象:

每一个事件都会包含一个事件对象event，在事件处理函数提供一个形参 该形参就会接收到该事件对象event

event对象中有多个属性，其中

- target属性：表示触发该事件的源组件
- currentTarget属性：表示事件绑定的当前组件

注:这两个属性都有dataset属性：当前组件上由data-开头的自定义属性组成的集合

```
event.currentTarget.dataset.hi
event:
    changedTouches: [{…}]
    currentTarget: {id: "", offsetLeft: 68, offsetTop: 21, dataset:
{…}}
    detail: {x: 189.328125, y: 43.65625}
    mark: {}
    mut: false
    target: {id: "", offsetLeft: 68, offsetTop: 21, dataset: {…}}
    timeStamp: 21965
    touches: [{…}]
    type: "tap"
    _userTap: true
    __proto__: Object
```

5.3 事件处理函数参数：

要给事件处理函数传参数 使用**data-属性名**的方式进行传参

6.细节 双向绑定

如果需要在用户输入的同时改变 `this.data.value` ，需要借助简易双向绑定机制。此时，可以在对应项目之前加入 `model:` 前缀

```
<!-- 通过size属性可以改变按钮的大小 -->
<button size="mini">小按钮</button>

<!-- input的双向绑定 -->
<input type="text" model:value="{{num}}"/>
```

这样，如果输入框的值被改变了， `this.data.value` 也会同时改变。同时， WXML 中所有绑定了 `value` 的位置也会被一同更新， 数据监听器 也会被正常触发。

7.选择器

类选择器、id选择器、元素选择器、::before、::after、::hover、 等 使用方法和php

选择器	样例	样例描述
.class	`.intro`	选择所有拥有 class="intro" 的组件
#id	`#firstname`	选择拥有 id="firstname" 的组件
element	`view`	选择所有 view 组件
element, element	`view, checkbox`	选择所有文档的 view 组件和所有的 checkbox 组件
::after	`view::after`	在 view 组件后边插入内容
::before	`view::before`	在 view 组件前边插入内容

建议：

实训不应该有休息时间，以前实训都没有休息时间