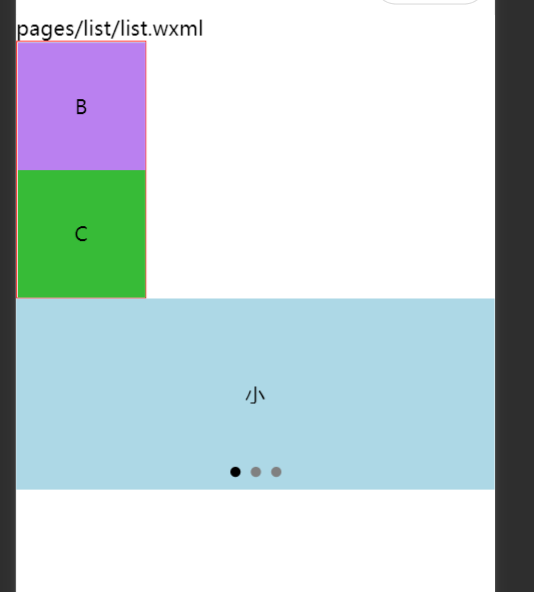
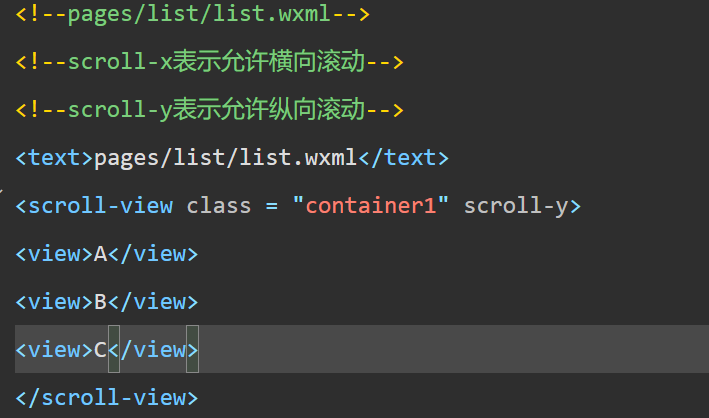
# 微信小程序开发学习

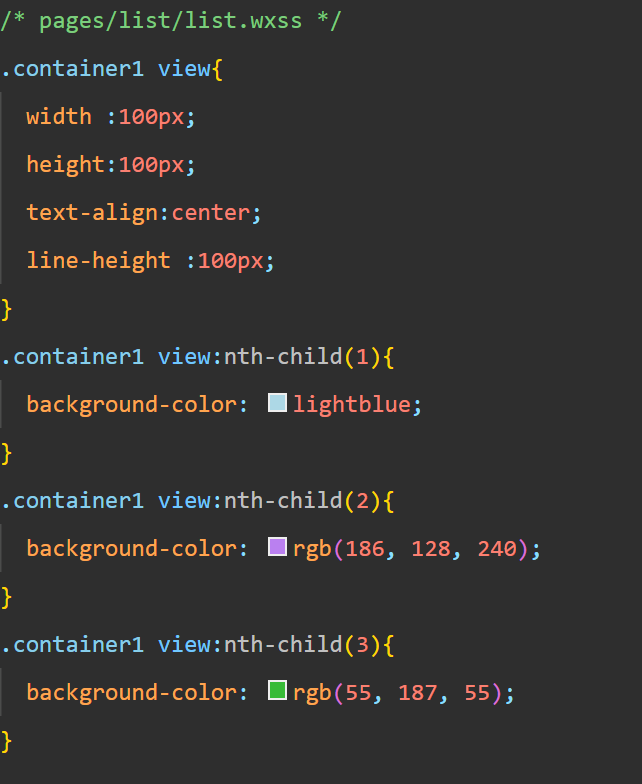
# 一、Part 1

## 安装：用到APPID

## 2、制作滚动的图形和轮播图







1. 轮播图

不要漏掉上面的：

<swiper class = "swiper-container "indicator-dots indicator-color="grey" autoplay interval="3000" circular>

<!--轮播图区域-->

<!--第一个-->

<swiper-item>

<view class="item ">Hello</view>

</swiper-item>

<swiper-item>

<view class="item ">你好</view>

</swiper-item>

<swiper-item>

<view class="item ">小程序第一界面</view>

</swiper-item>

</swiper>

wxss中进行渲染：

/\*轮播图样式\*/

.swiper-container

{

  height:150px;/\*轮播图容器高度\*/

}

.item {

  height: 100%;

  line-height: 150px;

  text-align: center;

}

swiper-item:nth-child(1) .item{

  background-color: lightblue;

}/\*.item前面加个空格才显示一些颜色\*/

swiper-item:nth-child(2) .item{

  background-color: rgb(173, 230, 181);

}

swiper-item:nth-child(3) .item{

  background-color: rgb(230, 173, 185);

}

  /\*.item前面加个空格\*/

Nth-child(1) .item{},注意空格

## 3、常用组件：

1. text 普通文本组件
2. rich-text 富文本组件，支持把HTML字符串渲染为WXML结构。



1. Button

<button type ="primary">点击进入</button>

1. Image组件

<image src="{{imgSrc}}"></image>

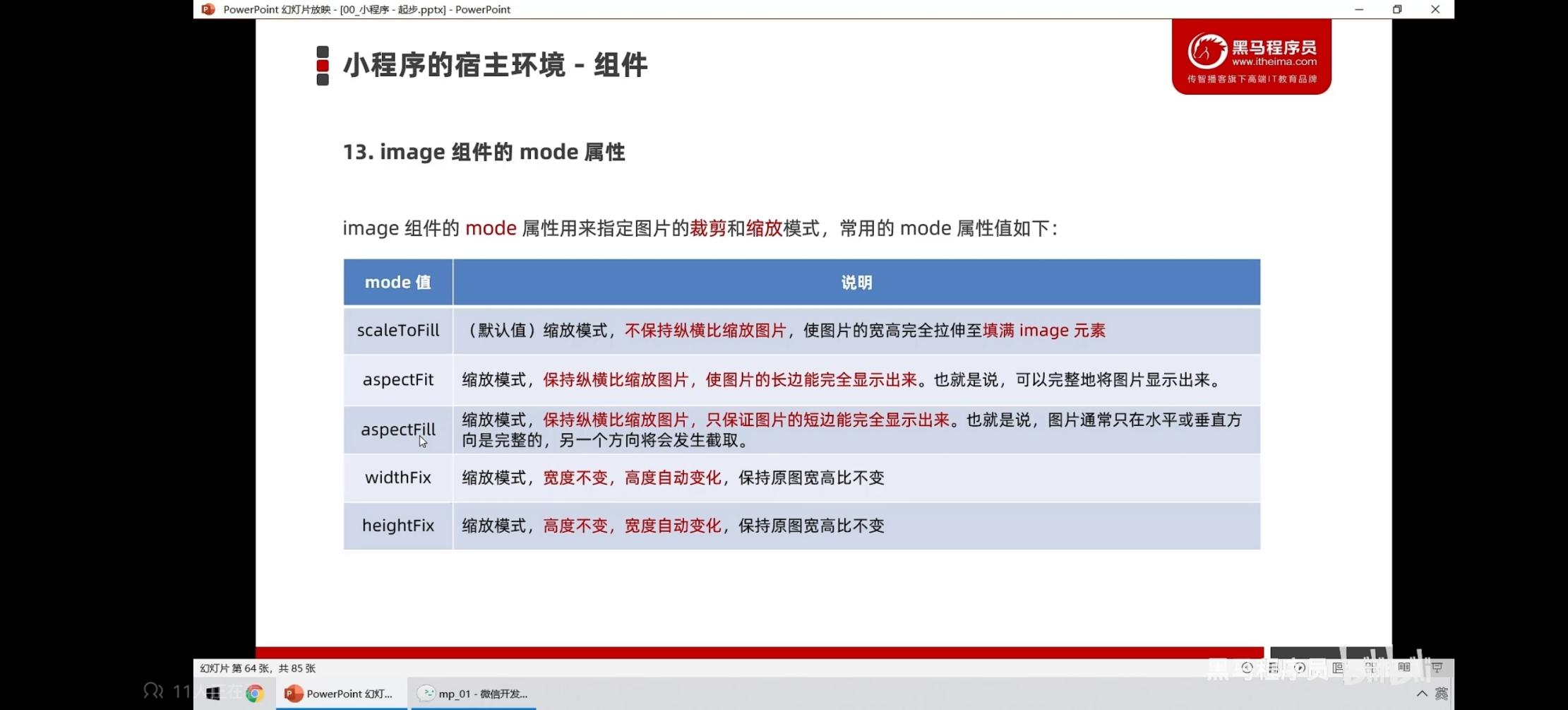
记得路径是这样的：

<image src="../../images/1.jpg" mode="aspectFill"></image>

../../images/1.jpg切记！



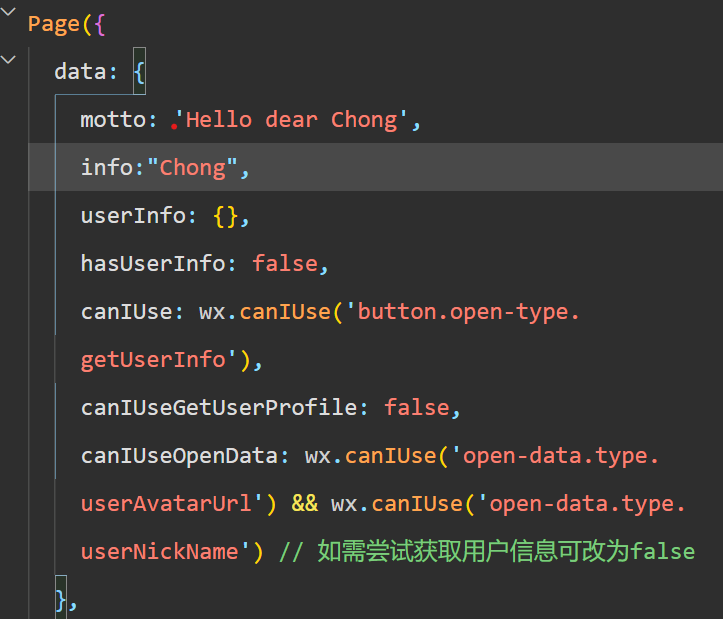
<image src="../../images/1.jpg" mode="heightFix"></image>



1. Navigator
2. part two

## WXML 模板语法

1. 基本原则：data中定义数据，WXML中使用数据
2. 定义数据：页面js文件中；



## mustache使用数据（双大括号）

<view>{{info}}</view>

绑定内容和属性，进行运算

## 事件绑定:渲染层到逻辑层

tap:点击一下，click; 绑定：bindtap 或band:tap

Input:文本框输入； bindinput bind:input

Change:状态改变时触发bindchange bind:change

事件对象的属性列表

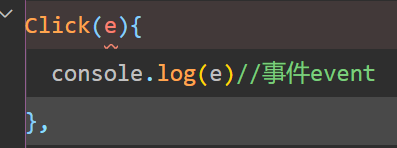
|  |  |  |
| --- | --- | --- |
| 属性 | 类型 | 说明 |
| target | Object | 触发事件的属性组合 |
| detail | Object | 额外信息 |
|  |  |  |

例：target 组件

bindtap的语法格式：

用t事件来响应用户的触摸行为

<button type ="primary" bindtap="bindTapHandler">按钮</button>



我后来把名称改成Click，注意Click不在data里面。

在data里面定义了Count =0,然后调用Count的值，每按一下按钮让count加一，但是，，Appdata里面没有变化。

## （5）为data中的数据赋值

//定义按钮的事件处理函数

Click(e){

  console.log(e)//事件event

},

//+1按钮的点击事件处理函数

Count(){

  this.setData({ Count:this.data.Count+1}

  )

},

## （6）事件传参数

不能在绑定事件时同时传参数，可以为组件提供data-\*自定义属性传参，\*代表参数名字。

<button type ="primary"bindtap ="btntap2" data-info="{{12345678}}">+2</button>

t页面

 btntap2(e)

  {

    console.log(e)

  },

更改一下内容，此后每按一下+2button,count都加8.

 btntap2(e)

  {

    this.setData({

      Count:this.data.Count+e.target.dataset.info

    })

  },

## （7）bindinput的语法格式

Wxml

<input bindinput="inputone"></input>

Js:

//input输入框的事件处理函数

inputone(e){

  console.log(e.detail.value)

},

拿到最新的文本框的值

## 文本框和data之间的数据同步

wxml中：

<input value ="{{msg}}" bindinput="inputone"></input>

Js.data中添加变量

  msg:'你好，今天是2023.10.31',

并且添加处理函数

  //input输入框的事件处理函数

inputone(e){

 // console.log(e.detail.value)

 this.setData({

   msg:e.detail.value

 })

},

在wxss中添加渲染：

input{

  border:1px solid ;

  margin: 5px ;

  padding : 5px ;

  border-radius: 3px;

}

2、条件渲染

1、if 和elif

改变tap的值，可以改变输出的值。

<view wx:if ="{{type==1}}">nan</view>

<view wx:elif ="{{type==1}}">nv</view>

<view wx:else>保密</view>

综合block控制多个组件的展示和隐藏

<block wx:if="{{true}}">

<view>view1</view>

<view>view2</view>

</block>

外层不会被渲染，但实际效果<block>和<view>相似。

<block wx:if="{{false}}">

<view>view1</view>

<view>view2</view>

</block>

隐藏两个view12.

1. Hidden

<view hidden={{flag或别的东西}}>条件为true时隐藏，否则显示</view>

1. hidden和if 对比

Wx:if 动态创建和移除元素；

Hidden:给元素加样式。

频繁切换时：使用hidden;控制条件复杂时，建议用if elif 和else

1. 列表渲染
2. wx:for 可以循环渲染重复的组件结构

先在index.js.data中添加数组arr1,(list.ts中添加数组就会报错）

 arr1:['春','夏','秋']

再在index.wxml中添加

<!--列表渲染-->

<view wx:for="{{arr1}}">索引{{index}}内容:{{item}}</view>

索引{{index}}，内容{{item}}

1. wx:key的使用

Index.js.data中

 userList:[

      {id:1 ,name:"l"},

      {id:2 ,name:"z"},

      {id:3 ,name:"y"}

    ]

Wxml:

<view wx:for="{{userList}}" wx :key="id">{{item.name}}</view>

# WXSS模板样式

1. Rpx：屏幕总宽度750rpx，用于屏幕适配。

rpx和px的单位换算：iPhone 6上，750rpx=375px=750物理像素。

1. 样式导入：@import导入外联样式表；

@import"../../common/common.wxss";

路径值报错

3、全局样式和局部样式：

App.scss作用于全部页面；

页面中的scss只作用于当前界面，

就近原则：当局部样式和全局样式冲突时，当前页面会用局部样式。

view:nth-child(4){

  background-color: blue;

}

1. 全局配置

（1）Pages:页面：

1. window:窗口外观：

导航栏背景色只支持以#开头的十六进制颜色。

文本颜色支持黑色和白色。

全局开启下拉刷新功能。

设置上拉触底的距离：

1. tabBar:用于实现多页面的快速切换、
2. 底部tabBar

简单配置如下：

小技巧：  
a、输入tabBar按下回车可以自动弹出组件；

b、图片./images/首页.png

  "tabBar": {

  "list":[

    {

      "pagePath": "pages/index/index",

      "text":"首页",

      "iconPath": "./images/首页.png",

      "selectedIconPath": "./images/首页-active.png"

    },

    {

      "pagePath": "pages/logs/logs",

      "text":"发布",

      "iconPath": "./images/发布.png",

      "selectedIconPath": "./images/发布-active.png"

    },

    {

      "pagePath": "pages/list/list",

      "text":"我",

      "iconPath": "./images/我.png",

      "selectedIconPath": "./images/我-active.png"

    }

  ]

  },

1. 顶部tabBar没有图标icon,只有文本

style:样式