**1：创建项目**

首先我们先在app.json中修改窗口头部的信息：



然后大家观察到window窗口的底部也多出了几个标签页导航：

我们小程序中的标签页导航叫做：tabBar:

list中有三个对象，分别对应我们的底部标签：

pagePath:文件路径

text：文字（标签名字）

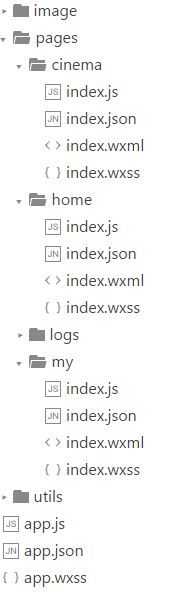
iconPath:图标的图片（未选中状态的图片）

selectedIconPath:选中状态的图片



 相应的我们要在pages目录下创建相应的文件夹：

目录如下：



 最后，我们的入口文件也需要改一下：



**2：上映页面**

上映页面对应的使我们的home文件夹；

index.wxml

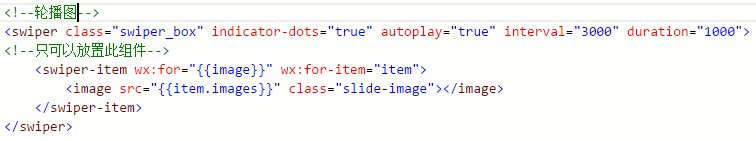
2.1 首先创建页面的容器：

<view class="container">

</view>

2.2 轮播图

    2.2.1轮播图在微信小程序中使用swiper组件，里面只能放swiper-item组件



    2.2.2 swiper相关知识

**Swiper在我们的小程序中被称为滑块视图容器。**

****

              注意：其中只可放置<swiper-item/>组件，否则会导致未定义的行为。

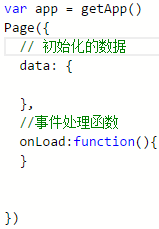
    2.2.3 数据的获取

    大家应该发现我们的页面什么都没有显示，这是为什么呢？原因是，我们的数据是从后台的真实接口中获取的。

    如何获取接口呢？

    使用我们的小程序提供的数据请求方式：wx.request

    打开我们的index.js文件

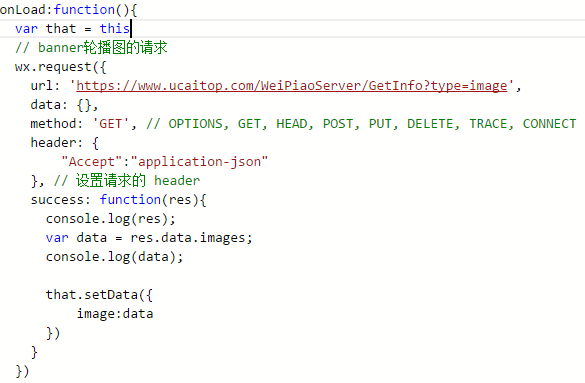


     注意：这里需提到一下我们的页面生命周期



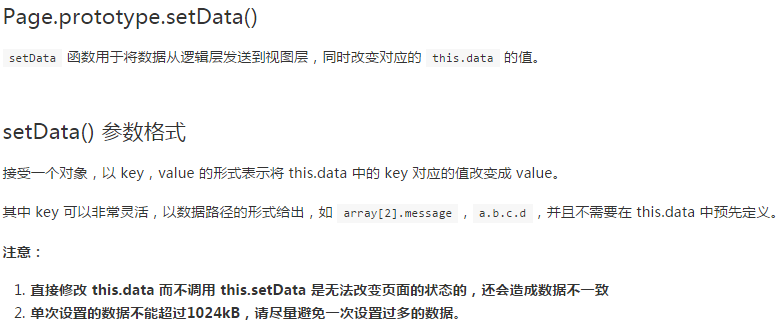
        因为我们的页面需要在加载的时候就显示出数据：

        所以我们的数据请求应该写在我们的onLoad函数中，



        在这个数据获取的过程中，我们如何检测是否获取到数据，可以使用console.log(data).

提示：this.setData()



    2.2.4 数据与前台的对应

        因为我们注意到接口数据为json,而我们想要的数据包含在一个数组中，所以我们在index.js中，将res.data.images（数组的名字）赋值给我们的image变量。

        那么在前台我们如何用到它，并且找到我们想要的每个数据呢？这里我们就需要用到我们的循环方法了

        在微信小程序中，提供了wx:for

        <swiper-item wx:for="{{image}}" wx:for-item="item">

            <image src="{{item.images}}" class="slide-image"></image>

        </swiper-item>

            swiper代表我们的每一个轮播块，我们的循环要放在他的标签中，{{}}是我们的数据绑定方式，写在其中的变量对应我们的js文件中的变量。这里我们要循环的是image，所以{{}}中的变量名就是我们的image, wx:for-item="item"中的item就代表我们数组中的每一个对象。

    2.2.5 样式的编写

                这里我们注意到，数据已经获取成功了，我们的轮播图还是没有出现，怎么办呢？

                打开我们的调试按钮，点击wxml

               找到我们的swiper部分，发现位置不对，我们就需要修改样式了。



这个时候我们看到轮播图不再最上面，是因为app.wxss里面关于container的边距我们没有注释，把它注释掉。

.container {

  height: 100%;

  display: flex;

  flex-direction: column;

  align-items: center;

  justify-content: space-between;

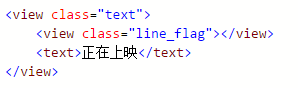
  /\*padding: 200rpx 0;\*/

  box-sizing: border-box;

}

2.3 正在上映的文字字体部分

 2.3.1 index.wxml



 2.3.2 index.wxss

    我们发现只需要修改样式就可以



2.4 电影信息部分

  2.4.1 index.wxml



 2.4.2 数据的获取

    同样我们需要使用wx.request，而且同样是在onLoad函数中



    这里我们同样可以使用console.log()方法检测我们的数据是否获取成功

    然后在json数据中找到对应的数组名称，赋值给我们的变量items

    将数据与前台对应，并在前台显示出来。

 2.4.3 样式的修改

    这里我们将使用到flex布局，然后将我们的样式分为三部分，左边，中间，和右边。

    因为要做到屏幕的自适应，所以我们使用百分比来规定每个部分所占的比例。

    左侧和右侧各为20%，中间部分为60%。

    这里注意：因为display：flex 所以我们的padding值只对上下边距有效。

    小程序的视图层布局主要是弹性框模型

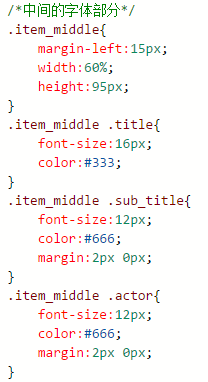
    弹性框模型分为弹性容器以及弹性项目

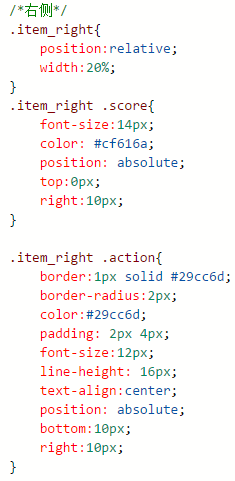
    我们设置最外层的容器为display：flex或者inline-flex时，该容器为弹性容器

    弹性容器的子容器一般称为弹性项目。

    横向布局：（类似于float的效果）----flex-direction:row







**3：影院页面**

   因为我们的影院是在cinema文件夹中，所以我们需要在此文件夹编写代码。

  3.1 index.wxml



 3.2 获取数据

   index.js

   同样使用wx.request



   这里查看数据是否获取成功同样使用console.log().

 3.3 样式的编写

  数据获取成功之后，发现我们的样式不对。

