小程序入门

吴小龙同學

Table of Contents

吴小龙同学

这款小程序诞生的前后。

2018 年将是小程序的红利年，我有计划今年做款小程序练下手，没想到雏形出来了，最近学习任务很重，《Java 开发接口》比我想象中更难，掉在坑里还没有爬出来，休息一下，玩下小程序，小程序虽小，但五脏俱全，到真的开始做了，就得想做什么样子产品，本来做个像我个人 APP 《微言》一样每天一篇好文，好处是数据源不用我多维护，注册小程序名字时《微言》、《每日晨读》、《大家》全被占了，又应征了那句话，新兴领域要想玩，得趁早啊，《大家》这个名字是从打算玩小程序就想到的名字，寓意文章皆出自大家之手，大家一起来鉴赏，可惜用不了，不过我个人还有个需求，我的公众号是坚持原创，但有时看到好的文章也想分享，本想再搞一个公众号专门来转载，担心精力不足，迟迟没做，现在小程序正好有这个契机，尽量每天分享一篇精选文章，最终我的小程序也是命名吴小龙同学。

玩小程序又燃起了我做产品的激情，产品规划如下：

V0.01

1、雏形完成；

2、我的模块，头像＋关于。

V0.02

1、首页文章列表，UI 抄公众号；

2、文章详情，Markdown 格式展示。

V0.03

1、首页文章列表分页；

2、文章评论，添加评论＋评论列表。

V0.04

1、收藏功能；

2、点赞；

3、阅读数。

V0.0.5

1、阅读足迹?

好了，暂时这么多，这款小程序肯定会长期维护下去，看看最终能做成什么样子，敬请一起期待。后续也将分享小程序相关开发教程。

后记

理想很美好，现实很骨感：

你的小程序"吴小龙同学"代码发布审核未通过，原因如下： 1:小程序内容不符合规则: (1):涉及个人小程序未允许内容：文娱-资讯，建议申请企业主体小程序。

能不能上线不重要了，我已经达到练手的目的了[捂脸]。

很早之前，小程序「吴小龙同学」就已经完成了七七八八，当我开始写这个教程，发现我要学习的东西好多，官方文档很详细，但是好多，不知道从何看起，我结合开发过程中遇到的，带着问题把官方文档仔细看了一遍，大概了然于胸，小程序相比 Android 真的简单很多。

适合对象

有一定编程基础或 Android 开发人员。

学习目标

入门，会小程序编程。

学习方法

小程序官方文档相当的全，也多，可能有点迷糊，最简单的方法，把所有文档全部过一遍，然后跟着敲代码练习。

成果展示

结语

2018 年将是小程序的红利年，很有必要学习一下，没有想象中那么难，我入门也不过花了一个月，可能算很慢了，小程序对个体很是不友好，很多功能受限，如果纯粹为了练手，那无所谓，来吧，小程序，玩起。

更新记录

- v0.01：写于 2018 年 5 月份。

准备

工欲善其事，必先利其器。

微信开发者工具

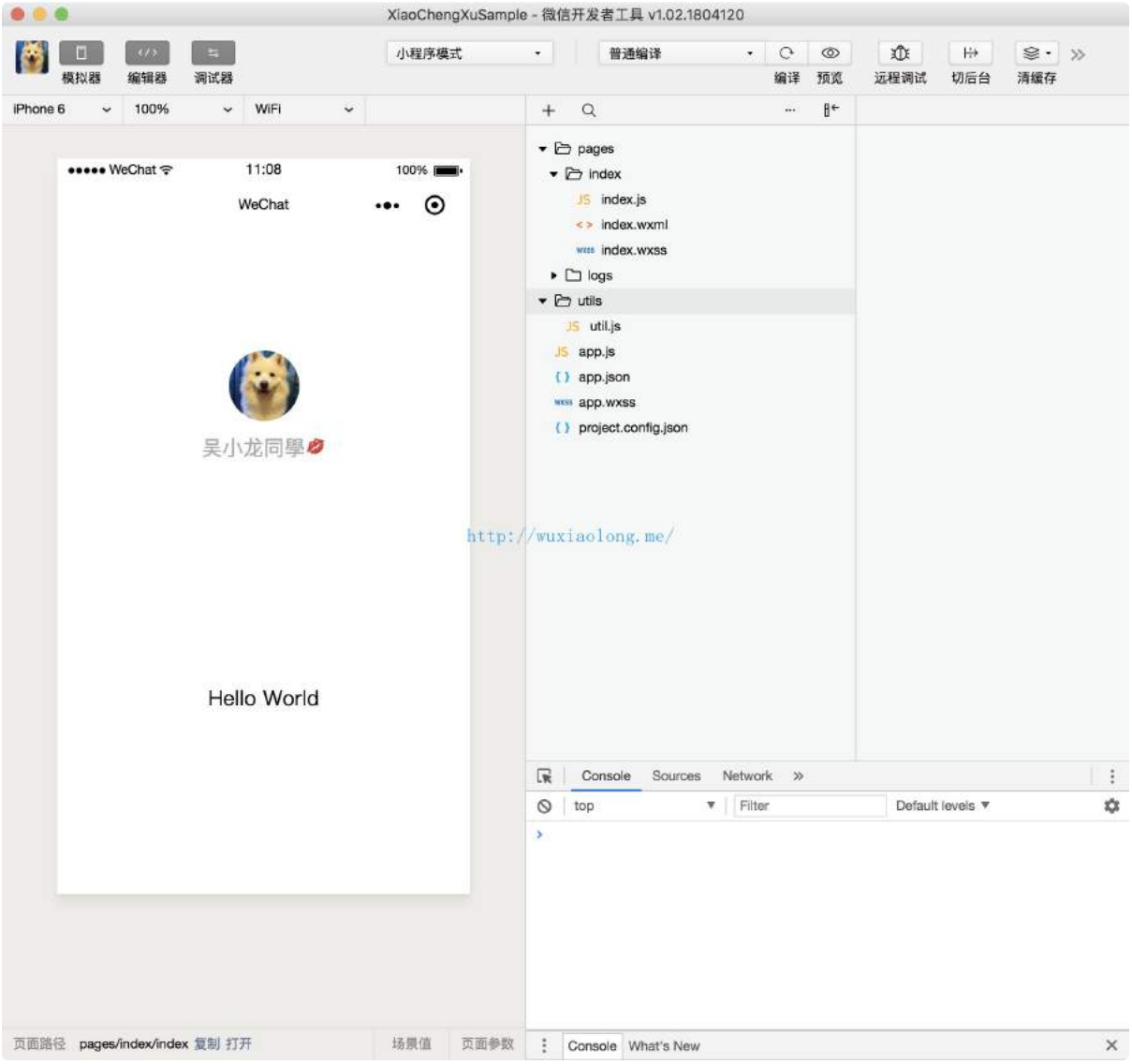
小程序官网：https://mp.weixin.qq.com/cgi-bin/wx ，这里提供了开发文档，开发工具，demo 等。

注册小程序

在我们下载了微信开发者工具，创建时，会让填 AppID，无小程序 AppID 部分功能受限，所以还是先注册下小程序，详细注册步骤见官网小程序注册。

微信开发者工具使用

创建项目，填写项目目录、AppID、项目名称，建立普通快速启动模板，进入是这样的：



详细使用教程见官网开发者工具的使用 。

如何导入新项目

微信开发者工具使用，没有找到直接导入项目的功能，只能通过新建项目，选要导入项目的目录，填入 AppID、项目名称，确定即可。

代码结构

官网：https://developers.weixin.qq.com/miniprogram/dev/framework/config.html

app.wxss

全局样式，对每一个页面都起作用。

app.json

全局配置，包括了小程序的所有页面路径、界面表现、网络超时时间、底部 tab 等。快速启动模板定义了两个页面，分别位于 pages/index/index 和 pages/logs/logs 目录。

```
{
  "pages":[
    "pages/index/index",
    "pages/logs/logs",
  ],
  "window":{
    "backgroundTextStyle":"light",
    "navigationBarBackgroundColor": "#fff",
    "navigationBarTitleText": "WeChat",
    "navigationBarTextStyle":"black"
  }
}
```

如何添加页面

直接在 app.json 的 pages 里添加：

```
{
  "pages":[
    "pages/index/index",
    "pages/logs/logs",
    //手动添加
    "pages/about/about"
  ],
}
```

会自动生成 about.js（交互逻辑）、about.json（独立定义每个页面的一些属性）、about.wxml（UI 布局 ）、about.wxss（样式） 四个文件。

注意：page 里面的样式，作为局部样式，会覆盖全局样式的样式，page.json 只能设置 window 相关的配置项，以决定本页面的窗口表现，所以无需写 window 这个键。如：

```
{
  "navigationBarTitleText": "详情"
}
```

安全域名校验

正式发布的小程序的网络请求是需要校验合法域名以及域名的 TLS 版本，详情 - 项目设置 - 「不校验合法域名、web-view（业务域名）、TLS 版本以及 HTTPS 证书」，在开发过程中勾上此选项，不进行校验。

快捷键

官网地址：https://developers.weixin.qq.com/miniprogram/dev/devtools/shortcut.html ，这里我摘抄了我常用的。

Mac OS	Windows	说明
⌘ + S	ctrl + S	保存文件
⇧ + ⌥ + F	shift + alt + F	格式化代码
⌥ + ⬆	alt + ⬆	代码上移一行
⌥ + ⬇	alt + ⬆	代码上移一行
⇧ + ⌥ + ⬆	shift + alt + ⬆	复制并向上粘贴
⇧ + ⌥ + ⬇	shift + alt + ⬆	复制并向下粘贴
⌘ + F	ctrl + F	文件内搜索
⇧ + ⌘ + F	shift + ctrl + F	项目内搜索

知识准备

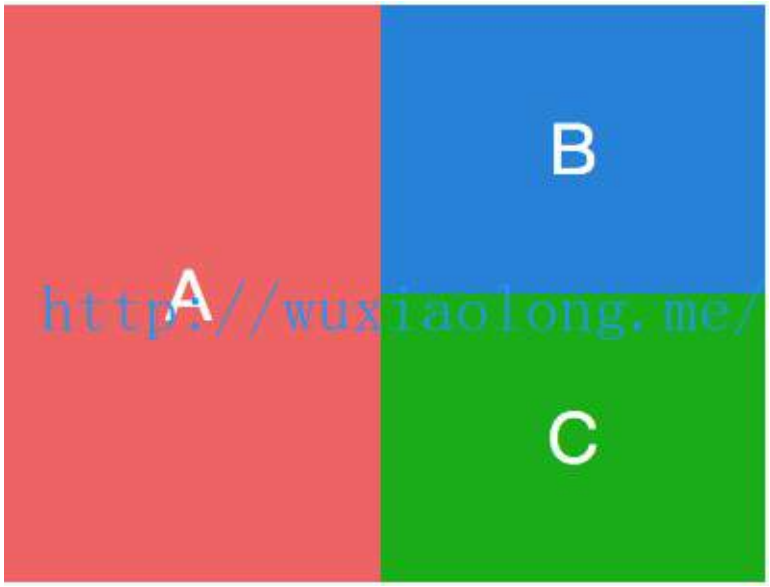
官方文档

框架：https://developers.weixin.qq.com/miniprogram/dev/framework/MINA.html

组件：https://developers.weixin.qq.com/miniprogram/dev/component/ 。

官方文档讲的很详细，需要自己静下心来，详细地过一遍，然后实践一下，CSS 语法不多讲，我也不会，说说后面我们实战会用到以及注意点。

Flex 布局



如图，这种布局，如何实现?

wxml

```
<view class='item'>

  <view class='item-text1'>A</view>
  <view  class='item-text'>
    <view class='item-text2'>B</view>
    <view class='item-text3'>C</view>
  </view>

</view>
```

wxss

```
.item {
```

```
  display: flex;
  flex-direction: row;
}

.item-text1 {
  width: 200rpx;
  height: 300rpx;
  background-color: #ee6363;
  color: #fff;
  font-size: 36rpx;
  display: flex;
  justify-content: center;
  align-items: center;
}

.item-text2 {
  width: 200rpx;
  height: 150rpx;
  background-color: #2782d7;
  color: #fff;
  font-size: 36rpx;
  display: flex;
  justify-content: center;
  align-items: center;
}

.item-text3 {
  width: 200rpx;
  height: 150rpx;
  background-color: #1aad19;
  color: #fff;
  font-size: 36rpx;
  display: flex;
  justify-content: center;
  align-items: center;
}
```

说明：

1、view 默认是 `display:block` 块内容器模式，总是使用新行开始显示； `display:flex` 可以指定为行内容器模式，在一行内显示子元素， `flex-wrap` 属性指定其是否换行；

2、 `flex-direction: row|row-reverse|column|column-reverse;` 可以指定水平的主轴（main axis）和垂直的交叉轴（cross axis）；

3、子元素对齐方式， `justify-conent` ：定义子元素在主轴上面的对齐方式， `align-items` ：定义子元素在交叉轴上对齐的方式。

数据绑定

```
<view> {{ message }} </view>
```

```
Page({
  data: {
    message: 'Hello World!'
  }
})
```

这样就进行了 message 数据绑定，详见官方文档：https://developers.weixin.qq.com/miniprogram/dev/framework/view/wxml/ 。

const VS var

1、const 声明的是常量，只可以在声明时赋值，在后面出现的代码中不能再修改；

2、var 声明的是变量，其作用域为该语句所在的函数内，let 是更好的 var，let 作用域为该语句所在的代码块内（大括号括起来的内容）。

点击事件

```
<view bindtap="onClick"> {{ message }} </view>
```

```
Page({

  /**
   * 页面的初始数据
   */
  data: {
    message: 'Hello World!'
  },

  onClick: function () {
    //不能直接修改 this.data 而不调用 this.setData 是无法改变页面的状态的，还会造成数据不一致
    this.setData({
      message: 'hhhhh'
    });
  },
})
```

tap：手指触摸后马上离开，事件绑定；

以 bind 或 catch 开头，然后跟上事件的类型，如这里的bindtap；

更多事件详见：https://developers.weixin.qq.com/miniprogram/dev/framework/view/wxml/event.html

页面跳转

调用 API 里 wx.navigateTo(OBJECT)，进行页面跳转，形式如 `'path?key=value&key2=value2'` ：

```
//跳转页面
wx.navigateTo({
    url: '/pages/detail/detail?articleId=' + articleId + '&title=' + title,
  })
```

1、 `/pages/detail/detail` ，下个页面的路径；

2、 `?articleId=' + articleId + '&title=' + title` ，路径后带参数。

官方文档：https://developers.weixin.qq.com/miniprogram/dev/framework/app-service/route.html

参数接受

```
 onLoad: function (options) {
    var articleId = options.articleId;
    var title = options.title;
    console.log(articleId);
    console.log(title);
  },
```

缓存

存入缓存

```
wx.setStorage({
       key: "openId", data: res.data.data.openid
     });
```

取缓存

```
wx.getStorage({
      key: "openId",
      success: function (res) {
        self.setData({
          userId: res.data
        })
      }
    })
```

官方文档：https://developers.weixin.qq.com/miniprogram/dev/api/data.html

网络请求

调用 API 里 wx.request(OBJECT)，发起网络请求。

```
wx.request({
      url: 'http://118.24.183.152/XiaoChengXu/article/detail',
      //设置 Http 方式
      method: "POST",
      //设置请求的 header
      header: {
        "Content-Type": "application/x-www-form-urlencoded"
      },
      //请求的参数
      data: {
        id: 4
      },
      success: function (res) {
        // 成功回调
      },
      fail: function () {
       // 失败回调
      }
    }),
```

官方文档：https://developers.weixin.qq.com/miniprogram/dev/api/network-request.html

数据准备

创建数据库

这部分结合我写得《Java 开发接口》教程是最好，当然如果对接口这块不感兴趣直接跳过，我会直接给出完整接口供练手，注意，我服务器只买了半年，截至 2018-10-25 22:16 到期，届时我提供的接口将无法访问。

用户表

微信登录，返回 openId，存到用户表，首先得查询这条数据在不在，没有的话，再插入，一开始是这样的，但感觉有些麻烦，两条 sql 语句，后来了解到有唯一索引，可以一条语句搞定。

表情问题

因为我的微信昵称：吴小龙同學　，有个表情，微信登录存储过程直接报错了，解决是用 utf8mb4 完全向后兼容 utf8，SSM 框架修改如下：

1、Navicat 将需要存储字符表情的字段字符集编码修改为 utf8mb4；

2、在需要插入字段的前面增加 set NAMES 'utf8mb4'。

WeiXinLoginDAO

```
public interface WeiXinLoginDAO {
    int setCharacters();
}
```

WeiXinLoginDAO.xml

```
<mapper namespace="com.wuxiaolong.xiaochengxu.dao.WeiXinLoginDAO">
    <update id="setCharacters">
        set NAMES 'utf8mb4'
    </update>
</mapper>
```

WeiXinLoginService

```
@Service
@Transactional
public class WeiXinLoginService {
    @Autowired
    private WeiXinLoginDAO weiXinLoginDAO;

    public boolean insertUser(String openId, String nickname, String avatar) {
```

```
            weiXinLoginDAO.setCharacters();//增加这行
            return weiXinLoginDAO.insertUser(openId, nickname, avatar);
        }
    }
```

文章列表

直接查询，做了分页，分页 sql 语句：`SELECT * FROM 表名 ORDER BY id DESC LIMIT 每页个数*(页码-1),每页个数` 。

文章阅读量

也是先判断这篇文章用户有没有看过，如果看过，阅读量将不增加，有个问题，假设用户没有看过，这个数据插入成功，但是文章表没有加 1，是不是要把刚刚文章阅读量插入的数据删掉呢，这里涉及事务，了解后，其实 SSM 已经帮我们做好了，Service 层做了，如：

```
public boolean updateReadNum(int articleId, String userId) {
    try {
        return readNumDAO.addReadNum(articleId, userId) && articleDAO.updateReadNum
(articleId);
    } catch (Exception e) {
        logger.error(e.getMessage(), e);
        return false;
    }
}
```

如果后者不成功，前者数据也不会插成功，完美。

文章点赞

文章点赞逻辑和阅读量一样，点赞也能取消，先文章点赞表记录删了，再将文章表点赞数减 1，同样涉及事务。还有个状态要判断，就是一进文章详情，得判断用户是否已经点赞。

评论列表

直接查询，没有做分页，实际开发，应该要做下分页。

添加评论

直接插入，允许多条评论。

添加收藏、我的收藏、用户反馈

没有做，收藏逻辑和点赞一样，用户反馈和添加评论逻辑一样，留给读者自己完成吧。

实战

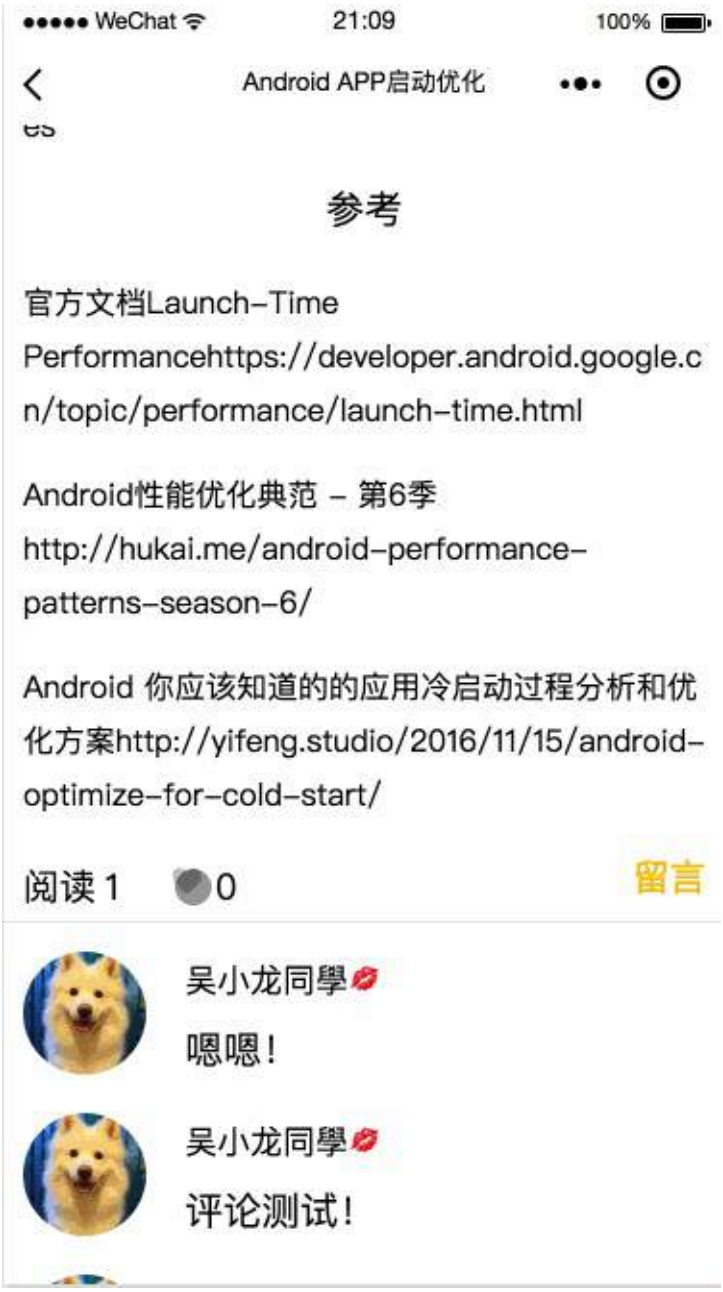
效果预览

1、文章列表

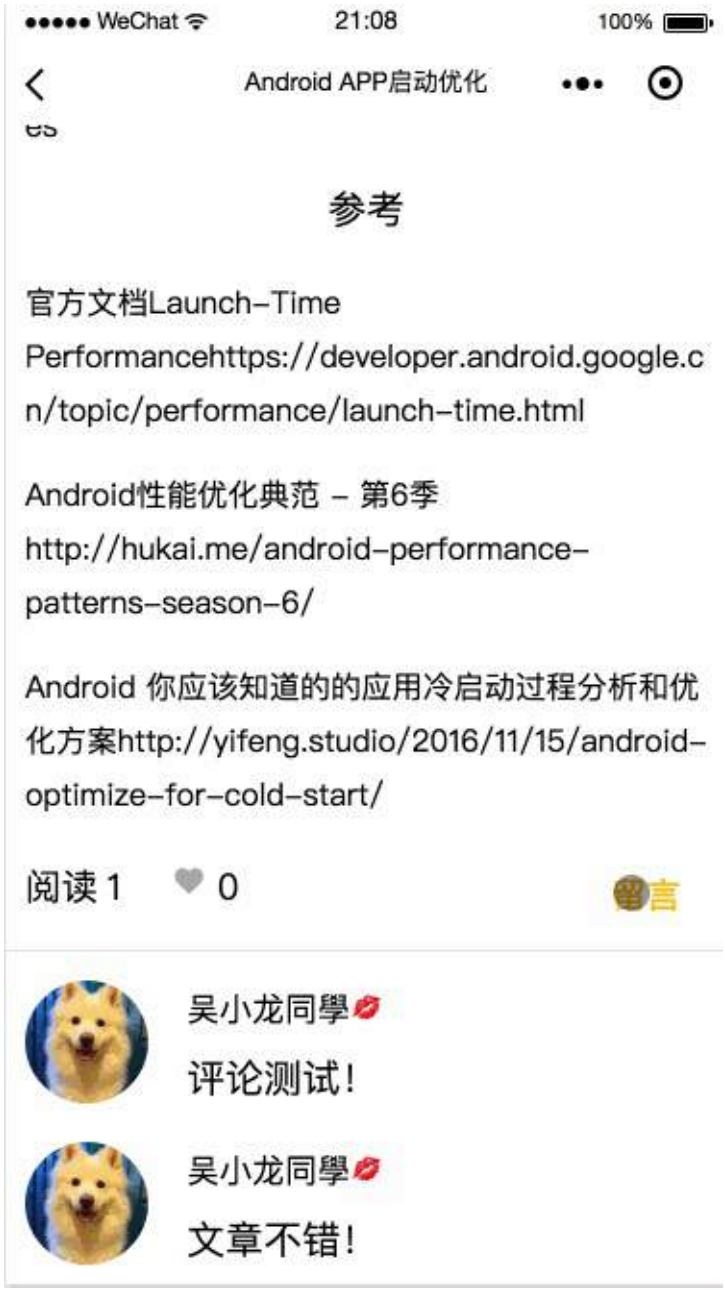
2、文章详情


WeChat
21:11
100%
Android APP启动优化
什么是APP启动屏幕
可能你也发现了，点击桌面图标，有时候出现一个白屏，然后才进入主界面，这是因为从桌面点击APP图标启动应用开始，程序会显示一个启动窗口等待Activity的创建加载完毕再进行显示。
来看一下Google官方文档《Launch-Time Performance》对应用启动方式的概述：
冷启动
冷启动指的是应用程序从头开始：系统的进程没有，直到此开始，创建了应用程序的进程。 在应用程序自设备启动以来第一次启动或系统杀死应用程序等情况下会发生冷启动。 这种类型的启动在最小化启动时间方面是最大的挑战，因为系统和应用程序比其他启动状态具有更多的工作。
热启动
与冷启动相比，热启动应用程序要简单得多，开销

3、点赞、取消点赞


WeChat
21:09
100%
Android APP启动优化
参考
官方文档Launch-Time Performancehttps://developer.android.google.cn/topic/performance/launch-time.html
Android性能优化典范 - 第6季 http://hukai.me/android-performance-patterns-season-6/
Android 你应该知道的的应用冷启动过程分析和优化方案http://yifeng.studio/2016/11/15/android-optimize-for-cold-start/
阅读 1
0
留言
吴小龙同學
嗯嗯！
吴小龙同學
评论测试！

4、添加评论



备注：PDF 并不支持动态图。

接下来，我们就要完成这个小程序。涉及功能有：文章列表、文章详情、文章评论列表、添加评论、阅读量增加、点赞、取消点赞、微信授权登录。

文章列表

列表渲染

在调接口之前，我们可以本地写死一个列表，展示页面，这里涉及知识点列表渲染，可以用 `wx:for` 控制属性绑定一个数组。首先新建 images 放入图片资源。

article-list.wxml

```
<view class='item'  wx:for="{{articleList}}" wx:key="articleList" >

  <view class='date'>
    <text>{{item.date}}</text>
  </view>
  <view class='content'>
    <image class='cover' src='{{item.cover}}' mode='aspectFill'></image>

    <view class='title'>
      <text>{{item.title}}</text>
    </view>

    <view class='summary'>
      <text>{{item.summary}}</text>
    </view>
  </view>
</view>
```

article-list.wxss

```
.date {
  display: flex;
  justify-content: center;
  margin-top: 20rpx;
  align-items: center;
}

.content {
  background: #fff;
  margin-left: 20rpx;
  margin-top: 10rpx;
  margin-right: 20rpx;
  border-radius: 10rpx;
  padding-bottom: 20rpx;
}

.cover {
  width: 100%;
```

```
  height: 400rpx;
  border-top-left-radius: 10rpx;
  border-top-right-radius: 10rpx;
}

.title {
  margin-left: 20rpx;
  margin-top: 20rpx;
  margin-right: 20rpx;
  font-size: 40rpx;
}

.summary {
  margin-left: 20rpx;
  margin-top: 20rpx;
  margin-right: 20rpx;
  font-size: 30rpx;
}
```

article-list.js

```
// pages/article-list/article-list.js
Page({

  /**
   * 页面的初始数据
   */
  data: {
    articleList: [
      {
        date: "2018年4月28日 18:00",
        cover: "/images/image1.jpg",
        title: "Android APP启动优化",
        summary: "打开一个APP，如果启动半天，你还有耐心等它吗？",
        content: "# 什么是AP"
      },
      {
        date: "2018年4月27日 18:00",
        cover: "/images/image2.jpg",
        title: "手把手教你做个人 app",
        summary: "没了接口，没了美工，就不能开发个人app了吗？",
        content: "我们都知道"
      },
    ],
  },

})
```

这样，当页面渲染时，绑定了 Page 的 data，呈现列表形式。

说明：

1、wx:for-item 可以指定数组当前元素的变量名；

2、wx:for-index 可以指定数组当前下标的变量名：

3、wx:key 来指定动态列表中项目的唯一的标识符。

接口请求

接口部分，我已经提供了，接口返回格式跟 articleList 一样，就能动态绑定。

分页

用到了 scroll-view 的 bindscrolltolower 用来监听滚动到底部/右边。

完整代码

index.wxml

```
<view class='items'>
  <scroll-view scroll-y="true" style="height: 1208rpx" bindscrolltolower="onLoadMor
e">
    <view class='item' style="flex-direction:column;" wx:for="{{articleList}}" wx:k
ey="articleList" bindtap='onItemClick' data-index="{{index}}">

      <view class='date'>
        <text>{{item.date}}</text>
      </view>
      <view class='content'>
        <image class='cover' src='{{item.cover}}' mode='aspectFill'></image>

        <view class='title'>
          <text>{{item.title}}</text>
        </view>

        <view class='summary'>
          <text>{{item.summary}}</text>
        </view>
      </view>
    </view>
    <view class="loadMore" hidden="{{!isLoadMore}}">正在载入...</view>
    <view class="loadFinish" hidden="{{!isLoadFinish}}">已加载全部</view>
  </scroll-view>
</view>
```

index.wxss

```
.items {
  background: #ebebeb;
  padding-top: 10rpx;
  padding-bottom: 10rpx;
  width: 100%;
  height: 100%;
}

.date {
  display: flex;
  justify-content: center;
  margin-top: 20rpx;
  align-items: center;
}

.content {
  background: #fff;
  margin-left: 20rpx;
  margin-top: 10rpx;
  margin-right: 20rpx;
  border-radius: 10rpx;
  padding-bottom: 20rpx;
}

.cover {
  width: 100%;
  height: 400rpx;
  border-top-left-radius: 10rpx;
  border-top-right-radius: 10rpx;
}

.title {
  margin-left: 20rpx;
  margin-top: 20rpx;
  margin-right: 20rpx;
  font-size: 40rpx;
}

.summary {
  margin-left: 20rpx;
  margin-top: 20rpx;
  margin-right: 20rpx;
  font-size: 30rpx;
}

.loadMore {
  display: flex;
  justify-content: center;
  margin-top: 20rpx;
  align-items: center;
```

```
}

.loadFinish {
  display: flex;
  justify-content: center;
  margin-top: 20rpx;
  align-items: center;
}
```

index.js

```
//index.js
//获取应用实例
const app = getApp()

Page({
  data: {
    articleList: [],
    page: 1,
    isLoadMore: true,
    isLoadFinish: false,
  },

  onLoad: function () {
    this.getArticleList();
  },
  //文章列表
  getArticleList: function () {
    var self = this;
    var page = self.data.page
    wx.request({
      url: 'http://118.24.183.152/XiaoChengXu/article/list',
      method: "POST",
      header: {
        "Content-Type": "application/x-www-form-urlencoded"
      },
      data: {
        page: page
      },
      success: function (res) {

        var code = res.data.code;
        if (code == 0) {
          //有数据返回
          var datas = res.data.data;
          var listTemp = self.data.articleList;
          listTemp = listTemp.concat(datas);
          self.setData({
            articleList: listTemp,
            isLoadMore: true,
```

```
          });
        } else {
          self.setData({
            isLoadMore: false,
            isLoadFinish: true,
          });
        }

      },
      fail: function () {

      },

    })
  },
  //跳转详情页
  onItemClick: function (e) {
    var index = e.currentTarget.dataset.index;
    var articleId = this.data.articleList[index].id;
    var title = this.data.articleList[index].title;

    //跳转页面
    wx.navigateTo({
      url: '/pages/detail/detail?articleId=' + articleId + '&title=' + title,
    })

  },
  //滚动到底部触发事件
  onLoadMore: function () {
    var self = this;
    if (self.data.isLoadMore && !self.data.isLoadFinish) {
      self.setData({
        page: self.data.page + 1,  //每次触发上拉事件，把 page + 1
      });
      self.getArticleList();
    }

  }
})
```

文章详情

Markdown 展示

因为我的博客都是 Markdown 写的，因此小程序也想做成 Markdown 格式的，这里用的第三方库 wemark，GitHub 地址：https://github.com/TooBug/wemark 。

集成步骤

1、下载 wemark 并拷贝到小程序根目录；

2、WXML 引入模板：`<import src="../../wemark/wemark.wxml" />`，并加入：

```
<view class="article">
  <!-- data中的参数和上方确定的数据名称保持一致 -->
  <template is="wemark" data="{{...wemark}}"></template>
</view>
```

3、WXSS 引用样式：`@import '/wemark/wemark.wxss';`

4、js 部分：

```
// pages/markdown/markdown.js
var wemark = require('../../wemark/wemark.js')
Page({

  /**
   * 页面的初始数据
   */
  data: {
    // 确定一个数据名称
    wemark: {},
  },

  /**
   * 生命周期函数--监听页面加载
   */
  onLoad: function (options) {
    var content = '# 吴小龙同學\n' +
      '1. 个人博客: [http://wuxiaolong.me/](http://wuxiaolong.me/)\n' +
      '2. 公众号: 吴小龙同学';
    wemark.parse(content, this, {
      name: 'wemark'
    })
  },
})
```

OK，示例完成，请求接口，将 content 换成真实数据即可。

阅读量

进入文章详情，调接口给文章的阅读量 + 1，接口有判断，一个用户只能 + 1。

点赞、取消点赞

进入文章详情，首先需要判断用户是否已经点赞，可以点赞，可以取消点赞。

完整代码

detail.wxml

```
<!--pages/detail/detail.wxml-->
<!-- 引入模板 -->
<import src="../../wemark/wemark.wxml" />

<view class="article">
  <!-- data中的参数和上方确定的数据名称保持一致 -->
  <template is="wemark" data="{{...wemark}}"></template>
</view>

<view class="likeRead">
  <view style="flex-basis: 40%;">
    <text class="readText">阅读</text>
    <text class="readNum">{{readNum}}</text>
    <image class="likeImage" src="{{likeImage}}" bindtap='addLikeClick' />
    <text class="likeNum">{{likeNum}}</text>
  </view>

  <view style="flex-basis: 60%;">
    <button class="userLogin" wx:if="{{!hasUserInfo}}" open-type="getUserInfo" bind
getuserinfo="getUserInfo">留言</button>
    <view wx:else class="addComment" bindtap='addComment'>留言</view>
  </view>
</view>

<view class='line' />

<view wx:for="{{commentList}}" wx:key="commentList" bindtap='onItemClick' data-index
="{{index}}">
  <view class='item'>
    <image class='avatar' src='{{item.avatar}}'></image>
    <view class='comment'>
      <view class='nickname'>
        <text>{{item.nickname}}</text>
      </view>
```

```
      <view class='content'>
        <text>{{item.content}}</text>
      </view>
    </view>
  </view>
</view>
```

detail.wxss

```
@import '/wemark/wemark.wxss';

.article {
  margin: 20rpx;
}

.likeRead {
  display: flex;
  flex-direction: row;
}

.likeImage {
  width: 40rpx;
  height: 40rpx;
  margin-top: 5rpx;
  margin-left: 50rpx;
}

.likeNum {
  margin-left: 10rpx;
}

.readText {
  margin-left: 20rpx;
}

.readNum {
  margin-left: 10rpx;
}

.userLogin {
  text-align: right;
  margin-right: 20rpx;
  border-radius: 50%;
  color: #ffba00;
  background-color: #fff;
  /* background-image: '/images/cover.jpg' */
}

/*使用 button::after{ border: none; } 来去除边框*/
```

```
.userLogin::after {
  border: none;
}

.addComment {
  display: flex;
  justify-content: flex-end;
  align-items: flex-end;
  margin-right: 20rpx;
  color: #ffba00;
}

.line {
  width: 100%;
  height: 1rpx;
  background: #e1e1e1;
  margin-top: 10rpx;
  margin-bottom: 10rpx;
}

.item {
  display: flex;
}

.avatar {
  width: 128rpx;
  height: 128rpx;
  margin: 20rpx;
  border-radius: 50%;
}

.comment {
  flex-direction: row;
}

.nickname {
  width: auto;
  height: auto;
  font-size: 35rpx;
  margin-left: 20rpx;
  margin-top: 20rpx;
  margin-right: 20rpx;
}

.content {
  margin-left: 20rpx;
  margin-top: 20rpx;
  margin-right: 20rpx;
  font-size: 40rpx;
  width: auto;
```

```
  height: auto;
}
```

detail.js

```
var wemark = require('../../wemark/wemark.js')
//获取应用实例
const app = getApp()
Page({

  /**
   * 页面的初始数据
   */
  data: {
    articleId: null,
    userId: null,
    readNum: 0,
    likeNum: 0,
    isLike: false,
    likeImage: '/images/collect1.png',
    hasUserInfo: false,
    // 确定一个数据名称
    wemark: {},
    commentList: [],

  },


  /**
   * 生命周期函数--监听页面加载
   */
  onLoad: function (options) {
    //options为页面路由过程中传递的参数
    console.log('onLoad');
    var self = this;
    self.setData({
      articleId: options.articleId,
      mername: options.title
    })
    wx.setNavigationBarTitle({
      title: self.data.mername//页面标题为路由参数
    })
    //获取用户ID
    wx.getStorage({
      key: "openId",
      success: function (res) {
        self.setData({
          userId: res.data
        })
      }
```

```
    })

    // 由于 getUserInfo 是网络请求，可能会在 Page.onLoad 之后才返回
    // 所以此处加入 callback 以防止这种情况
    app.userInfoReadyCallback = res => {
      this.setData({
        hasUserInfo: true
      })
    }
    //全局用户信息
    if (app.globalData.userInfo) {
      this.setData({
        hasUserInfo: true
      });
    }

    this.getArticleDetail()
    this.queryLike()
    this.addReadNum()

  },
  // 函数如何传参?
  getArticleDetail: function () {
    var self = this;
    wx.request({
      url: app.globalData.serveUrl + '/article/detail',
      method: "POST",
      header: {
        "Content-Type": "application/x-www-form-urlencoded"
      },
      //请求的参数
      data: {
        id: self.data.articleId
      },
      success: function (res) {
        self.setData({
          readNum: res.data.data.readNum,
          likeNum: res.data.data.likeNum,
        });
        var content = res.data.data.content;
        wemark.parse(content, self, {
          // 新版小程序可自适应宽高
          // imageWidth: wx.getSystemInfoSync().windowWidth - 40,
          name: 'wemark'
        })
      },
      fail: function () {

      }
    })
  },
```

```
  //评论列表
  getCommentList: function () {
    var self = this;
    wx.request({
      url: app.globalData.serveUrl + '/comment/list',
      method: "POST",
      header: {
        "Content-Type": "application/x-www-form-urlencoded"
      },
      data: {
        articleId: self.data.articleId
      },
      success: function (res) {
        var code = res.data.code;
        if (code == 0) {
          var datas = res.data.data;
          self.setData({
            commentList: datas,
          });
        } else {

        }

      },
      fail: function () {

      },

    })


  },

  //查询当前用户是否对这篇文章进行了点赞
  queryLike: function () {
    var self = this;
    wx.request({
      url: app.globalData.serveUrl + '/likeNum/query',
      method: "POST",
      header: {
        "Content-Type": "application/x-www-form-urlencoded"
      },
      data: {
        articleId: self.data.articleId,
        userId: self.data.userId,
      },
      success: function (res) {
        var code = res.data.code;
        if (code == 0) {
          //点赞过
          self.setData({
```

```
            isLike: true,
            likeImage: '/images/collect1.png',
          });
        } else {
          //没有点赞过
          self.setData({
            isLike: false,
            likeImage: '/images/collect0.png',
          });
        }
      },
      fail: function () {

      },

    })


  },

  //点赞点击事件
  addLikeClick: function () {
    if (this.data.isLike) {
      this.cancelLikeNum();
    } else {
      this.addLikeNum();
    }
  },

  //调用点赞接口
  addLikeNum: function () {
    var self = this;
    wx.request({
      url: app.globalData.serveUrl + '/likeNum/add',
      method: "POST",
      header: {
        "Content-Type": "application/x-www-form-urlencoded"
      },
      data: {
        articleId: self.data.articleId,
        userId: self.data.userId,
      },
      success: function (res) {
        var code = res.data.code;
        if (code == 0) {
          self.setData({
            isLike: true,
            likeImage: '/images/collect1.png',
            likeNum: self.data.likeNum + 1
          });
          wx.showToast({
```

```
            icon: 'success',
            duration: 2000
          })
        } else {

        }

      },
      fail: function () {

      },
    })
  },

  //调用取消点赞接口
  cancelLikeNum: function () {
    var self = this;
    wx.request({
      url: app.globalData.serveUrl + '/likeNum/cancel',
      method: "POST",
      header: {
        "Content-Type": "application/x-www-form-urlencoded"
      },
      data: {
        articleId: self.data.articleId,
        userId: self.data.userId,
      },
      success: function (res) {
        var code = res.data.code;
        if (code == 0) {
          self.setData({
            isLike: false,
            likeImage: '/images/collect0.png',
            likeNum: self.data.likeNum - 1
          });
          wx.showToast({
            icon: 'success',
            duration: 2000
          })
        } else {

        }

      },
      fail: function () {

      },

    })
```

```
  },

  //增加阅读量接口
  addReadNum: function () {
    var self = this;
    wx.request({
      url: app.globalData.serveUrl + '/readNum/add',
      method: "POST",
      header: {
        "Content-Type": "application/x-www-form-urlencoded"
      },
      data: {
        articleId: self.data.articleId,
        userId: self.data.userId,
      },
      success: function (res) {
        var code = res.data.code;
        if (code == 0) {
          self.setData({
            readNum: self.data.readNum + 1
          });
        } else {

        }

      },
      fail: function () {

      },

    })


  },

  //授权获取用户信息，如果没有登录，其实这里「留言」是一个按钮
  getUserInfo: function (e) {
    console.log('getUserInfo');
    var self = this
    // //调用应用实例的方法获取全局数据
    app.getUserInfo(function (userInfo) {
      //更新数据
      self.setData({
        hasUserInfo: true
      })
      self.addComment();
    })
  },

  //跳转到留言页面
  addComment: function () {
```

```
    console.log('addComment');
    var self = this;
    //跳转页面
    wx.navigateTo({
      url: '/pages/add-comment/add-comment?articleId=' +
      self.data.articleId + '&title=' + self.data.mername,
    })
  },
  /**
   * 生命周期函数--监听页面初次渲染完成
   */
  onReady: function () {
    console.log('onReady');
  },

  /**
   * 生命周期函数--监听页面显示
   */
  onShow: function () {
    console.log('onShow');
    this.getCommentList();
  },
})
```

微信登录

登录流程

1、wx.getSetting()：判断是否进行过用户信息授权；

2、wx.getUserInfo() ： 如果授权，直接获取用户信息；

3、wx.login() ： 如果没有授权，调用登录 API，获取临时 code；

4、wx.getUserInfo() ： 登录调成功，获取用户信息；

5、wx.request() ： 获取用户信息成功，请求自己服务器接口，获取 openId，并将用户信息保存到自己服务器上去。

具体实现

因为多次用到登录，因此改需要放到 app.js，登录只能用 button 的 bindgetuserinfo 来获取用户信息。

account.wxml

```
  <view class="userinfo">
    <button class="userinfo-avatar-default" wx:if="{{!hasUserInfo}}" open-type="get
UserInfo" bindgetuserinfo="getUserInfo"/>
    <block wx:else>
      <image class="userinfo-avatar" src="{{userInfo.avatarUrl}}" background-size="
cover"></image>
      <text class="userinfo-nickname">{{userInfo.nickName}}</text>
    </block>
  </view>
```

account.wxss

```
/* pages/account/account.wxss */

.userinfo {
  display: flex;
  flex-direction: column;
  align-items: center;
}

.userinfo-avatar-default {
  width: 128rpx;
  height: 128rpx;
  margin: 20rpx;
  border-radius: 50%;
}
```

```
/*使用 button::after{ border: none; } 来去除边框*/
.userinfo-avatar-default::after {
  border: none;
}
.userinfo-avatar {
  width: 128rpx;
  height: 128rpx;
  margin: 20rpx;
  border-radius: 50%;
}

.userinfo-nickname {
  margin-top: 20px;
}
```

account.js

```
//获取应用实例
const app = getApp()

Page({
  data: {
    userInfo: {},
    hasUserInfo: false,
    isLogin: false
  },

  onLoad: function () {
    var self = this;
    // 由于 getUserInfo 是网络请求，可能会在 Page.onLoad 之后才返回
    // 所以此处加入 callback 以防止这种情况
    app.userInfoReadyCallback = res => {
      this.setData({
        userInfo: res.userInfo,
        hasUserInfo: true
      })
    }
    if (app.globalData.userInfo) {
      this.setData({
        userInfo: app.globalData.userInfo,
        hasUserInfo: true
      });
    }

  },
  getUserInfo: function (e) {
    var self = this
    //调用应用实例的方法获取全局数据
    app.getUserInfo(function (userInfo) {
      //更新数据
```

```
        self.setData({
          userInfo: userInfo,
          hasUserInfo: true
        })

      })
    },

})
```

这里 getUserInfo 调用了 app.js 里的 getUserInfo，app.js 里 onLaunch 调用wx.getSetting()，判断用户是否进行过信息授权，已经授权，直接获取用户信息。

```
//app.js
App({
  globalData: {
    userInfo: null,
    serveUrl: 'http://118.24.183.152/XiaoChengXu',
  },
  onLaunch: function () {
    // 小程序启动之后 触发
    // 获取用户信息
    wx.getSetting({
      success: res => {
        if (res.authSetting['scope.userInfo']) {
          // 已经授权，可以直接调用 getUserInfo 获取头像昵称，不会弹框
          wx.getUserInfo({
            success: res => {
              // 可以将 res 发送给后台解码出 unionId
              this.globalData.userInfo = res.userInfo

              // 由于 getUserInfo 是网络请求，可能会在 Page.onLoad 之后才返回
              // 所以此处加入 callback 以防止这种情况
              if (this.userInfoReadyCallback) {
                this.userInfoReadyCallback(res)
              }
            }
          })
        }
      }
    })
  },
  //微信登录
  getUserInfo: function (callback) {
    var self = this
    if (this.globalData.userInfo) {
      //判断 callback 是不是函数类型同时将一个参数传入名为 callback 的函数下。
      typeof callback == "function" && callback(this.globalData.userInfo)
    }
    else {
```

```
      // 登录
      wx.login({
        success: function (res) {
          wx.showLoading({
            title: '加载中',
          })
          // 发送 res.code 到后台换取 openId, sessionKey, unionId
          if (res.code) {
            var loginModel = res
            // 获取用户信息
            wx.getUserInfo({
              success: function (res) {
                var userInfoModel = res
                //发起网络请求，保存用户信息到自己服务器上去
                wx.request({
                  url: self.globalData.serveUrl + '/weixin/login',
                  method: "POST",
                  header: {
                    "Content-Type": "application/x-www-form-urlencoded"
                  },
                  data: {
                    codeTemp: loginModel.code,
                    nickname: userInfoModel.userInfo.nickName,
                    avatar: userInfoModel.userInfo.avatarUrl
                  },
                  success: function (res) {
                    var code = res.data.code;
                    if (code == 0) {
                      self.globalData.userInfo = userInfoModel.userInfo
                      wx.setStorage({ key: "openId", data: res.data.data.openid });
                      typeof callback == "function" && callback(userInfoModel.userI
nfo)
                    }
                    wx.hideLoading()
                  },
                  fail: function (res) {
                    wx.hideLoading()
                  }
                })

              },
              fail: function (res) { //用户点了“拒绝”
                console.log('用户点了“拒绝”')
                console.log(res)
                // wx.authorize({
                //   scope: 'scope.userInfo',
                //   success() {

                //   }
                // })
              }
```

```
            })
          } else {
            console.log('登录失败! ' + res.errMsg)
          }
        },
        fail: function (res) {
          console.log('登录失败! ' + res.errMsg)
        }
      })
    }
  },

})
```

添加评论

点击文章详情留言，跳到添加评论页面，在跳转之前，我做了是否登录的判断，没有登录，先授权。

add-comment.wxml

```
<view class="section">
  <form bindsubmit="bindFormSubmit">
    <textarea class='content' placeholder="说点什么……" name="textarea"/>
    <button class='submit' form-type="submit"> 提交 </button>
  </form>
</view>
```

add-comment.wxss

```
.content {
  margin: 20rpx;
}

.submit {
  margin: 20rpx;
  background-color: #ffba00;
  color: #fff;
}
```

add-comment.js

```
// pages/add-comment/add-comment.js
//获取应用实例
const app = getApp()
Page({

  /**
   * 页面的初始数据
   */
  data: {
    articleId: null,
    userId: null
  },

  /**
   * 生命周期函数--监听页面加载
   */
  onLoad: function (options) {
    var self = this;
    self.setData({
      articleId: options.articleId,
```

```
      mername: options.title
    }),
      wx.getStorage({
        key: "openId",
        success: function (res) {
          console.log(res.data)
          self.setData({
            userId: res.data
          })
        }
      }),
      wx.setNavigationBarTitle({
        title: self.data.mername//页面标题为路由参数
      })


  },
  bindFormSubmit: function (e) {
    var self = this;
    var content = e.detail.value.textarea;
    if (content.length > 0) {
      console.log(self.data.articleId)
      console.log(self.data.userId)
      wx.request({
        url: app.globalData.serveUrl +'/comment/add',
        method: "POST",
        header: {
          "Content-Type": "application/x-www-form-urlencoded"
        },
        data: {
          articleId: self.data.articleId,
          userId: self.data.userId,
          content: content,
        },
        success: function (res) {
          console.log(res.data);

          var code = res.data.code;
          if (code == 0) {
            wx.showToast({
              title: '评论成功',
              icon: 'success',
              duration: 2000
            }),
              wx.navigateBack({

              })
          } else {

          }
```

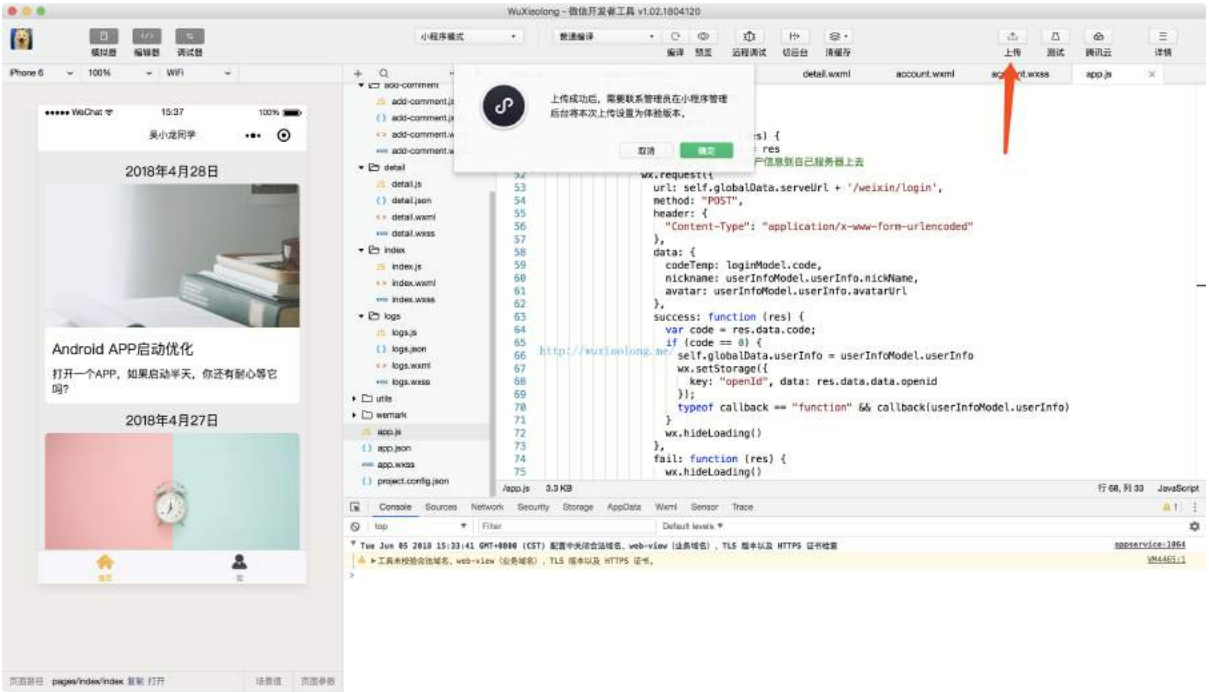
```
          },
          fail: function () {

          },

        })
      }
    },
})
```

源码

公众号「吴小龙同学」回复：吴小龙同学Plus，获取完整的小程序。

上线

提交审核



在开发者工具中上传了小程序代码之后，登录 小程序管理后台 - 开发管理 - 开发版本，找到提交上传的版本，填写相关的信息，即可以将小程序提交审核。

发布

审核通过之后，管理员的微信中会收到小程序通过审核的通知，此时登录 小程序管理后台 - 开发管理 - 审核版本中可以看到通过审核的版本。请点击发布，既可发布小程序（未验证）。