小程序系统API调用

王红元 coderwhy

目录
content

网络请求 – API参数

■ 微信提供了专属的API接口,用于网络请求: wx.request(Object object)

属性	类型	默认值	必填	说明	最低版本
url	string		是	开发者服务器接口地址	
data	string/object/ArrayBuffer		否	请求的参数	
header	Object		否	设置请求的 header，header 中不能设置 Referer。 `content-type` 默认为 `application/json`	
timeout	number		否	超时时间，单位为毫秒。默认值为 60000	2.10.0
method	string	GET	否	HTTP 请求方法	
dataType	string	json	否	返回的数据格式	
responseType	string	text	否	响应的数据类型	1.7.0

■ 比较关键的几个属性解析:

- url: 必传, 不然请求什么.
- data: 请求参数
- method: 请求的方式
- success: 成功时的回调
- fail: 失败时的回调

网络请求 – API使用

- 直接使用wx.request(Object object)发送请求:

```
wx.request({
  url: "http://www.codercba.com:1888/api/city/all",
  success: (res) => {
    console.log(res.data);
  },
  fail: (err) => {
    console.log(err);
  }
})
```

```
wx.request({
  url: 'http://www.codercba.com:1888/api/home/houselist',
  data: {
    page: 1
  },
  success: (res) => {
    console.log(res.data);
  },
  fail: (err) => {
    console.log(err);
  }
})
```

网络请求 – API封装

```
class HYRequest {
  request(url, method, params) {
    return new Promise((resolve, reject) => {
      wx.request({
        url: BASE_URL + url,
        method: method,
        data: params,
        success: function(res) {
          resolve(res.data)
        },
        fail: reject
      })
    })
  }

  get(url, params) {
    return this.request(url, "GET", params)
  }

  post(url, data) {
    return this.request(url, "POST", data)
  }
}

const hyRequest = new HYRequest()
```

网络请求域名配置

- 每个微信小程序需要事先设置通讯域名，小程序**只可以跟指定的域名进行网络通信。**
  - 小程序登录后台 – 开发管理 – 开发设置 – 服务器域名；
- **服务器域名请在 「小程序后台 - 开发 - 开发设置 - 服务器域名」 中进行配置，配置时需要注意：**
  - 域名只支持 https (wx.request、wx.uploadFile、wx.downloadFile) 和 wss (wx.connectSocket) 协议；
  - 域名不能使用 IP 地址（小程序的局域网 IP 除外）或 localhost；
  - 可以配置端口，如 https://myserver.com:8080，但是配置后只能向 https://myserver.com:8080 发起请求。如果向 https://myserver.com、https://myserver.com:9091 等 URL 请求则会失败。
  - 如果不配置端口。如 https://myserver.com，那么请求的 URL 中也不能包含端口，甚至是默认的 443 端口也不可以。如果向 https://myserver.com:443 请求则会失败。
  - 域名必须经过 ICP 备案；
  - 出于安全考虑，api.weixin.qq.com 不能被配置为服务器域名，相关 API 也不能在小程序内调用。 开发者应将 AppSecret 保存到后台服务器中，通过服务器使用 getAccessToken 接口获取 access_token，并调用相关 API；
  - 不支持配置父域名，使用子域名。

展示弹窗效果

- 小程序中展示弹窗有四种方式: showToast、showModal、showLoading、showActionSheet

```
wx.showToast({
  title: '哈哈哈',
  icon: "error",
  duration: 2000,
  mask: true,
  success: (res) => {
    console.log("展示成功:", res);
  },
  fail: (err) => {
    console.log("展示失败:", err);
  }
})
```


```
wx.showModal({
  title: "呵呵呵",
  cancelText: "要不起",
  cancelColor: '#f00',
  confirmText: "要",
  confirmColor: "#0f0",
  success: (res) => {
    console.log("展示成功:", res);
  },
  fail: (err) => {
    console.log("展示失败:", err);
  }
})
```


```
wx.showActionSheet({
  itemList: ["MacBook", "iPad", "iPhone"],
  itemColor: "#f00",
  success: (res) => {
    console.log("成功:", res);
  },
  fail: (err) => {
    console.log("失败:", err);
  }
})
```

分享功能

- **分享是小程序扩散的一种重要方式，小程序中有两种分享方式：**
  - 方式一：点击右上角的菜单按钮，之后点击转发
  - 方式二：点击某一个按钮，直接转发
- **当我们转发给好友一个小程序时，通常小程序中会显示一些信息：**
  - 如何决定这些信息的展示呢？通过 onShareAppMessage
  - 监听用户点击页面内转发按钮（button 组件 open-type="share"）或右上角菜单“转发”按钮的行为，并自定义转发内容。
  - 此事件处理函数需要 return 一个 Object，用于自定义转发内容；

字段	说明	默认值	最低版本
title	转发标题	当前小程序名称	
path	转发路径	当前页面 path，必须是以 / 开头的完整路径	
imageUrl	自定义图片路径，可以是本地文件路径、代码包文件路径或者网络图片路径。支持 PNG 及JPG。显示图片长宽比是 5:4。	使用默认截图	1.5.0

获取设备信息

- **在开发中，我们需要经常获取当前设备的信息，用于手机信息或者进行一些适配工作。**
  - 小程序提供了相关个API：**wx.getSystemInfo(Object object)**

Object object

属性	类型	默认值	必填	说明
success	function		否	接口调用成功的回调函数
fail	function		否	接口调用失败的回调函数
complete	function		否	接口调用结束的回调函数（调用成功、失败都会执行）

获取位置信息

- **开发中我们需要经常获取用户的位置信息，以方便给用户提供相关的服务：**
  - 我们可以通过API获取：**wx.getLocation(Object object)**

```
onGetLocationInfo() {
  wx.getLocation({
    success: (res) => {
      console.log(res);
    }
  })
}
```

- **对于用户的关键信息，需要获取用户的授权后才能获得：**
  - https://developers.weixin.qq.com/miniprogram/dev/reference/configuration/app.html#permission

```
"permission": {
  "scope.userLocation": {
    "desc": "获取你的位置信息"
  }
},
```

Storage存储

- 在开发中，某些常见我们需要将一部分数据存储在本地：比如token、用户信息等。
  - 小程序提供了专门的Storage用于进行本地存储。
- 同步存取数据的方法：
  - wx.setStorageSync(string key, any data)
  - any wx.getStorageSync(string key)
  - wx.removeStorageSync(string key)
  - wx.clearStorageSync()
- 异步存储数据的方法：
  - wx.setStorage(Object object)
  - wx.getStorage(Object object)
  - wx.removeStorage(Object object)
  - wx.clearStorage(Object object)

```
wx.setStorageSync('name', "why")
wx.setStorageSync('age', 18)
const name = wx.getStorageSync('name')
const age = wx.getStorageSync('age')
console.log(name, age);
wx.removeStorageSync('name')
wx.clearStorageSync()
```

```
wx.setStorage({
  key: "name",
  data: "coderwhy",
  encrypt: true,
  success: (res) => {
    console.log(res);
  }
})

wx.getStorage({
  key: "name",
  success: (res) => {
    console.log(res);
  }
})
```

界面跳转的方式

- 界面的跳转有两种方式：**通过navigator组件** 和 **通过wx的API跳转**
- **这里我们先以wx的API作为讲解：**

路由

名称	功能说明
wx.switchTab	跳转到 tabBar 页面，并关闭其他所有非 tabBar 页面
wx.reLaunch	关闭所有页面，打开到应用内的某个页面
wx.redirectTo	关闭当前页面，跳转到应用内的某个页面
wx.navigateTo	保留当前页面，跳转到应用内的某个页面
wx.navigateBack	关闭当前页面，返回上一页面或多级页面

页面跳转 - navigateTo

- **wx.navigateTo(Object object)**
  - 保留当前页面，跳转到应用内的某个页面；
  - 但是不能跳到 tabbar 页面；

Object object

属性	类型	默认值	必填	说明
url	string		是	需要跳转的应用内非 tabBar 的页面的路径 (代码包路径), 路径后可以带参数。参数与路径之间使用 `?` 分隔，参数键与参数值用 `=` 相连，不同参数用 `&` 分隔；如 'path?key=value&key2=value2'
events	Object		否	页面间通信接口，用于监听被打开页面发送到当前页面的数据。基础库 2.7.3 开始支持。
success	function		否	接口调用成功的回调函数
fail	function		否	接口调用失败的回调函数
complete	function		否	接口调用结束的回调函数（调用成功、失败都会执行）

```
// pages/main/main.js
Page({
  pushDetail() {
    const name = "why"
    const age = 18

    wx.navigateTo({
      url: `/pages/detail/detail?name=${name}&age=${age}`,
    })
  }
})
```

```
// pages/detail/detail.js
Page({
  onLoad(options) {
    const name = options.name
    const age = options.age
    console.log(name, age);
  }
})
```

页面返回 - navigateBack

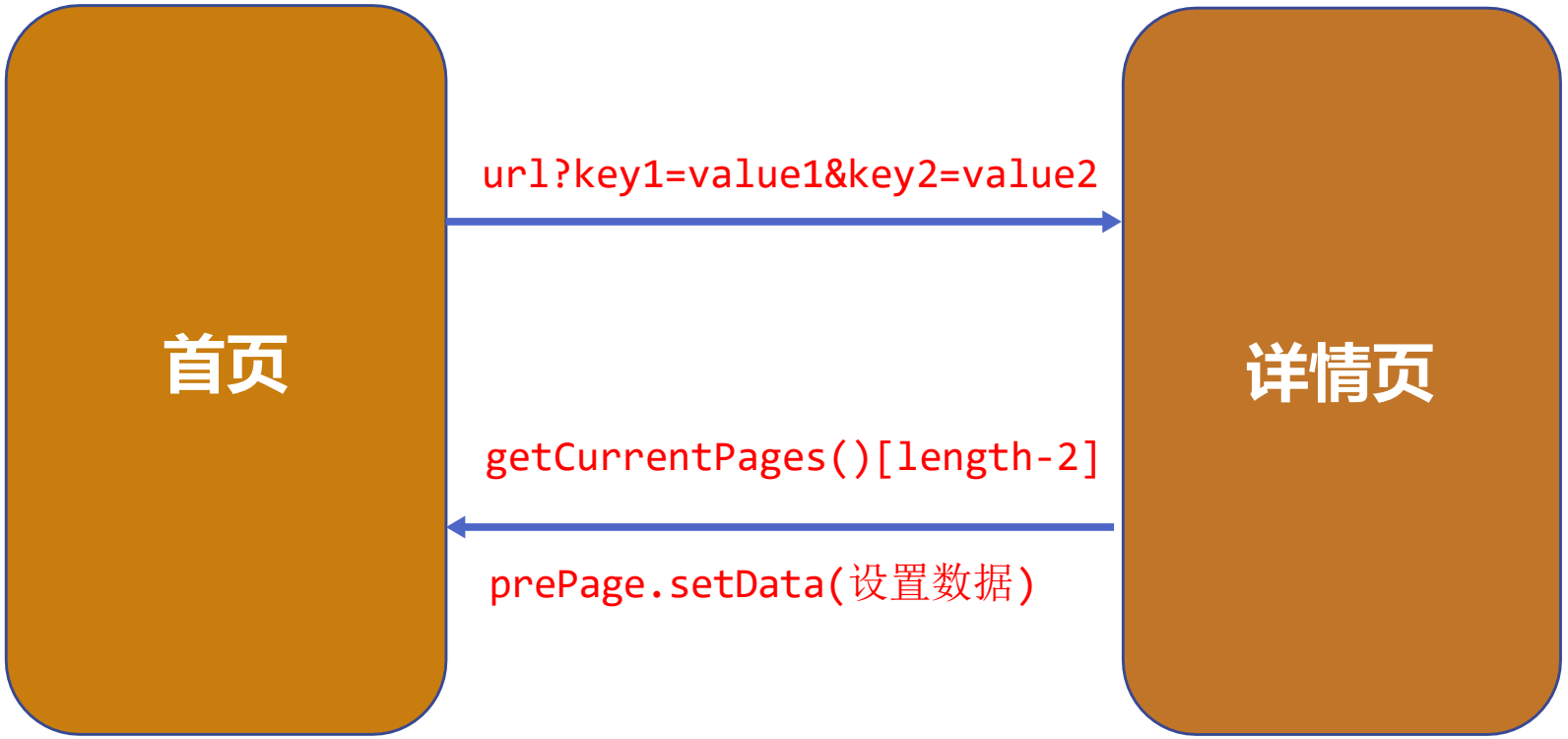
- **wx.navigateBack(Object object)**
  - 关闭当前页面，返回上一页面或多级页面。

Object object

属性	类型	默认值	必填	说明
delta	number	1	否	返回的页面数，如果 delta 大于现有页面数，则返回到首页。
success	function		否	接口调用成功的回调函数
fail	function		否	接口调用失败的回调函数
complete	function		否	接口调用结束的回调函数（调用成功、失败都会执行）

页面跳转 - 数据传递（一）

- 如何在界面跳转过程中我们需要相互传递一些数据，应该如何完成呢?
  - 首页 -> 详情页：使用URL中的query字段
  - 详情页 -> 首页：在详情页内部拿到首页的页面对象，直接修改数据




```
pushDetail() {
  const name = "why"
  const age = 18

  wx.navigateTo({
    url: `/pages/detail/detail?name=${name}&age=${age}`
  })
}
```

```
onPassBackTap() {
  const pages = getCurrentPages()
  const prevPage = pages[pages.length - 2]
  prevPage.setData({
    title: "呵呵呵"
  })
}
```

页面跳转 - 数据传递（二）

- 早期数据的传递方式只能通过上述的方式来进行，在小程序基础库 2.7.3 开始支持events参数，也可以用于数据的传递。

events	Object	否	页面间通信接口，用于监听被打开页面发送到当前页面的数据。基础库 2.7.3 开始支持。

```
pushDetail() {
  const name = "why"
  const age = 18

  wx.navigateTo({
    url: `/pages/detail/detail?name=${name}&age=${age}`,
    events: {
      acceptSomeData(data) {
        console.log("main中接受参数:", data);
      }
    }
  })
}
```

```
onPassBack2Tap() {
  // 1.拿到eventChannel
  const eventChannel = this.getOpenerEventChannel()

  // 2.传递数据给上一个页面
  eventChannel.emit("acceptSomeData", {
    name: "detail",
    count: 1000
  })
}
```

界面跳转的方式

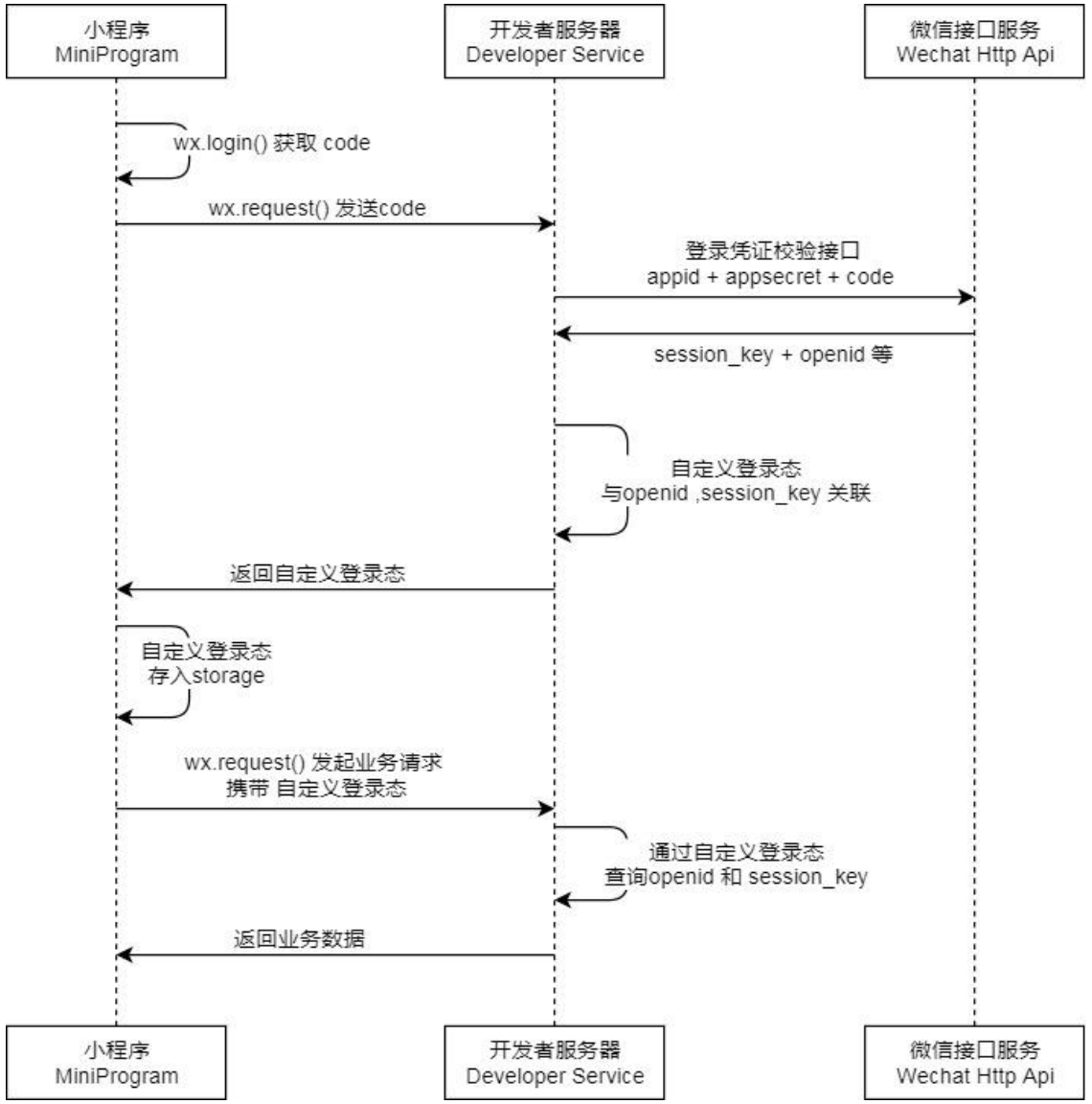
- navigator组件主要就是用于界面的跳转的，也可以跳转到其他小程序中：

属性	类型	默认值	必填	说明	最低版本
target	string	self	否	在哪个目标上发生跳转，默认当前小程序	2.0.7
url	string		否	当前小程序内的跳转链接	1.0.0
open-type	string	navigate	否	跳转方式	1.0.0
delta	number	1	否	当 open-type 为 'navigateBack' 时有效，表示回退的层数	1.0.0
app-id	string		否	当`target="miniProgram"`且`open-type="navigate"`时有效，要打开的小程序 appId	2.0.7
path	string		否	当`target="miniProgram"`且`open-type="navigate"`时有效，打开的页面路径，如果为空则打开首页	2.0.7

小程序登录解析

- **为什么需要用户登录?**
  - 增加用户的粘性和产品的停留时间;
- **如何识别同一个小程序用户身份?**
  - 认识小程序登录流程
  - openid和unionid
  - 获取code
  - 换取authToken
- **用户身份多平台共享**
  - 账号绑定
  - 手机号绑定

小程序用户登录的流程




```
// 1.获取code
const res = await wx.login({})
const code = res.code

// 2.换取token
wx.request({
  url: 'http://123.207.32.32:3000/login',
  data: {
    code
  },
  method: "post",
  success: (res) => {
    const token = res.data.token
    wx.setStorageSync('token', token)
  }
})
```