4.4 发起 HTTPS 网络通信

小程序经常需要往服务器传递数据或者从服务器拉取信息，这个时候可以使用 wx.request 这个 API，在这一节我们会重点讨论 wx.request 的使用和注意事项。为了叙述方便，假设我们的服务器域名是 test.com。

4.4.1 wx.request 接口

如果我们需要从 https://test.com/getinfo 接口拉取用户信息，其代码示例如下所示，详细参数如表 4-1 所示。

代码清单 4-7 wx.request 调用示例

```
wx.request({
  url: 'https://test.com/getinfo',
  success: function(res) {
    console.log(res)// 服务器回包信息
  }
})
```

表 4-1 wx.request 详细参数

参数名	类型	必填	默认值	描述
url	String	是		开发者服务器接口地址

参数名	类型	必填	默认值	描述
data	Object/String	否		请求的参数
header	Object	否		设置请求的 header，header 中不能设置 Referer，默认 header['content-type'] = 'application/json'
method	String	否	GET	（需大写）有效值：OPTIONS, GET, HEAD, POST, PUT, DELETE, TRACE, CONNECT
dataType	String	否	json	回包的内容格式，如果设为 json，会尝试对返回的数据做一次 JSON 解析
success	Function	否		收到开发者服务成功返回的回调函数，其参数是一个 Object，见表 4-2。
fail	Function	否		接口调用失败的回调函数

参数名	类型	必填	默认值	描述
complete	Function	否		接口调用结束的回调函数（调用成功、失败都会执行）

4.4.2 服务器接口

url 参数是当前发起请求的服务器接口地址，小程序宿主环境要求 request 发起的网络请求必须是 https 协议请求，因此开发者服务器必须提供 HTTPS 服务的接口，同时为了保证小程序不乱用任意域名的服务，wx.request 请求的域名需要在小程序管理平台进行配置[2]，如果小程序正式版使用 wx.request 请求未配置的域名，在控制台会有相应的报错。

一般我们在开发阶段时，处于开发阶段的服务器接口还没部署到现网的域名下，经常会通过另一个域名来进行开发调试，考虑到这一点，为了方便开发者进行开发调试，开发者工具、小程序的开发版和小程序的体验版在某些情况下[3]允许 wx.request 请求任意域名。

由于我们一直在迭代更新小程序，那么就会有一个问题：在新版小程序发布时的某段时间内，会有部分用户使用旧版本的小程序[4]。如果接口需要支持新的特性需要修改返回的数据格式，那接口的参数和返回字段至少向前兼容一个版本。举个例子，假设前边的 https://test.com/getinfo 接口返回的 JSON 数据为：{ "username": "zhangsan","sex": "man" }，在新版本中，我们需要把 sex 字

段的值改成用 0，1 来表示性别男女。为了保持接口向前兼容，我们不应该直接改 sex 字段值的类型，而是返回的 JSON 数据中再定义多一个字段 sexNumber，这样旧版本通过这个接口拿到的数据格式依旧是能够正常工作的。

4.4.3 请求参数

通过 wx.request 这个 API，有两种方法把数据传递到服务器：通过 url 上的参数以及通过 data 参数。举个例子：我们需要向服务器拿 id 为 1 的用户的信息，同时我们把当前小程序的版本带给服务器，让服务器可以做新旧版逻辑兼容，两种方法的代码示例如代码 4-8 所示。

代码清单 4-8 wx.request 调用示例

```
// 通过 url 参数传递数据
wx.request({
  url:'https://test.com/getinfo?id=1&version=1.0.0',
  success: function(res) {
    console.log(res)// 服务器回包信息
  }
})
   // 通过 data 参数传递数据
wx.request({
  url: 'https://test.com/getinfo',
     data: { id:1, version:'1.0.0' },
  success: function(res) {
```

```
    console.log(res)// 服务器回包信息
  }
})
```

两种实现方式在 HTTP GET 请求的情况下表现几乎是一样的，需要留意的是url是有长度限制的，其最大长度是 1024 字节，同时 url 上的参数需要拼接到字符串里，参数的值还需要做一次 urlEncode。向服务端发送的数据超过 1024 字节时，就要采用 HTTPPOST 的形式，此时传递的数据就必须要使用 data 参数，基于这个情况，一般建议需要传递数据时，使用 data 参数来传递。

我们再来单独看看 POST 请求的情况，并不是所有请求都是按照键值对 key=value 的形式传递到后台服务器，有时候需要传一些比较复杂的数据结构到后台的时候，用 JSON 格式会更加合适。此时我们可以在 wx.request 的 header 参数设置 content-type 头部为 application/json，小程序发起的请求的包体内容就是 data 参数对应的 JSON 字符串，代码示例如下。

代码清单 4-9 wx.request 发起 POST 请求包体使用 json 格式

```
// 请求的包体为 {"a":{"b":[1,2,3],"c":{"d":"test"}}}
wx.request({
  url: 'https://test.com/postdata',
  method: 'POST',
  header: { 'content-type': 'application/json'},
  data: {
    a: {
```

```
    b: [1, 2, 3],
    c: { d: "test" }
  }
 },
 success: function(res) {
  console.log(res)// 服务器回包信息
 }
})
```

4.4.4 收到回包

通过 wx.request 发送请求后，服务器处理请求并返回 HTTP 包，小程序端收到回包后会触发 success 回调，同时回调会带上一个 Object 信息，详细参数表 4-2 所示。

表 4-2 wx.request 的 success 返回参数

参数名	类型	描述
data	Object/String	开发者服务器返回的数据
statusCode	Number	开发者服务器返回的 HTTP 状态码

参数名	类型	描述
header	Object	开发者服务器返回的 HTTP Response Header

尤其注意，只要成功收到服务器返回，无论 HTTP 状态码是多少都会进入 success 回调。因此开发者自己通过对回包的返回码进行判断后再执行后续的业务逻辑。

success 回调的参数 data 字段类型是根据 header['content-type']决定的，默认 header['content-type']是'application/json'，在触发 success 回调前，小程序宿主环境会对 data 字段的值做 JSON 解析，如果解析成功，那么 data 字段的值会被设置成解析后的 Object 对象，其他情况 data 字段都是 String 类型，其值为 HTTP 回包包体。

4.4.5 一般使用技巧

1. 设置超时时间

小程序发出一个 HTTPS 网络请求，有时网络存在一些异常或者服务器存在问题，在经过一段时间后仍然没有收到网络回包，我们把这一段等待的最长时间称为请求超时时间。小程序 request 默认超时时间是 60 秒，一般来说，我们不需要这么长的一个等待时间才收到回包，可能在等待 3 秒后还没收到回包就需要给用户一个明确的服务不可用的提示。在小程序项目根目录里边的 app.json 可以指定 request 的超时时间。

代码清单 4-10 app.json 指定 wx.requset 超时时间为 3000 毫秒

```
{
  "networkTimeout": {
    "request": 3000
  }
}
```

2. 请求前后的状态处理

大部分场景可能是这样的，用户点击一个按钮，界面出现“加载中...”的 Loading 界面，然后发送一个请求到后台，后台返回成功直接进入下一个业务逻辑处理，后台返回失败或者网络异常等情况则显示一个“系统错误”的 Toast，同时一开始的 Loading 界面会消失。我们给出一个常见的 wx.request 的示例代码，如下所示。

代码清单 4-11 wx.request 常见的示例代码

```
var hasClick = false;

Page({

  tap: function() {

    if (hasClick) {

      return

    }

    hasClick = true

    wx.showLoading()
```

```
    wx.request({
      url: 'https://test.com/getinfo',
      method: 'POST',
      header: { 'content-type':'application/json' },
      data: { },
      success: function (res) {
        if (res.statusCode === 200) {
          console.log(res.data)// 服务器回包内容
        }
      },
      fail: function (res) {
        wx.showToast({ title: '系统错误' })
      },
      complete: function (res) {

        wx.hideLoading()

        hasClick = false

      }
    })
  }
})
```

为了防止用户极快速度触发两次 tap 回调，我们还加了一个 hasClick 的“锁”，在开始请求前检查是否已经发起过请求，如果没有才发起这次请求，等到请求返回之后再把锁的状态恢复回去。

4.4.6 排查异常的方法

在使用 wx.request 接口我们会经常遇到无法发起请求或者服务器无法收到请求的情况，我们罗列排查这个问题的一般方法：

1. 检查手机网络状态以及 wifi 连接点是否工作正常。
2. 检查小程序是否为开发版或者体验版，因为开发版和体验版的小程序不会校验域名。
3. 检查对应请求的 HTTPS 证书是否有效，同时 TLS 的版本必须支持 1.2 及以上版本，可以在开发者工具的 console 面板输入 showRequestInfo() 查看相关信息。
4. 域名不要使用 IP 地址或者 localhost，并且不能带端口号，同时域名需要经过 ICP 备案。
5. 检查 app.json 配置的超时时间配置是否太短，超时时间太短会导致还没收到回报就触发 fail 回调。
6. 检查发出去的请求是否 302 到其他域名的接口，这种 302 的情况会被视为请求别的域名接口导致无法发起请求。