云开发公开课

预习小册

目录

Part1：快速入门小程序

1.1 前置知识

1.1.1 技术栈

在小程序开发前，你需要对 Html、CSS、JavaScript 三门前端语言有一定的了解，它们能够帮助你快速上手小程序开发。如果你曾经从未接触过这三门语言，我们推荐使用 PC 访问下列第三方网站进行学习。

学习 Html：http://www.w3school.com.cn/html/index.asp

学习 CSS：http://www.w3school.com.cn/css/index.asp

学习 JS：http://www.w3school.com.cn/js/index.asp

1.1.2 微信开发文档

除了提前学习技术栈，在课程开始之前还需要了解微信提供的小程序开发教程及开发文档。在正式上课期间，对本部分不会有过多的讲解。你可通过微信扫描下方二维码快速查阅相关文档：

开发教程

框架、组件及 API 文档

1.2 准备工作

1.2.1 注册小程序

在开始开发小程序前，你需要拥有一个小程序账号，通过这个账号可以管理你的小程序。需要使用 PC 进入以下链接自行注册：

https://mp.weixin.qq.com/wxopen/waregister?action=step1

1.2.2 下载 IDE

使用 PC 进入以下链接下载微信开发者工具：

https://developers.weixin.qq.com/miniprogram/dev/devtools/download.html

如需进一步了解微信开发者工具，可以通过微信扫码查阅：

微信 IDE 使用文档

Part 2：云开发入门实战

2.1 初探云开发

2.1.1 云开发简介

「云开发」是腾讯云为移动开发者提供的一站式后端云服务，它帮助开发者统一构建和管理资源，免去了移动应用开发过程中繁琐的服务器搭建及运维、域名注册及备案、数据接

口实现等繁琐流程，让开发者可以专注于业务逻辑的实现，而无需理解后端逻辑及服务器运维知识，开发门槛更低，效率更高。

2.1.2 云开发能力

云数据库

- 文档型：数据库包含多个近似于 JSON 数组的集合，数组中的对象是记录，格式 JSON 文档。
- 简单易用：数据库 API 包含增删改查，操作简单；支持触发器，满足特殊场景。
- 权限控制：通过 API 在客户端内和云函数内进行数据操作，安全可靠。

云存储

- 快速上传：提供文件存储空间，可在客户端和云函数端通过 API 使用存储。
- 权限管理：基于用户身份的安全控制，带权限管理的云端下载。
- CDN 加速：存储内的文件，天然 CDN 加速，提升用户体验。

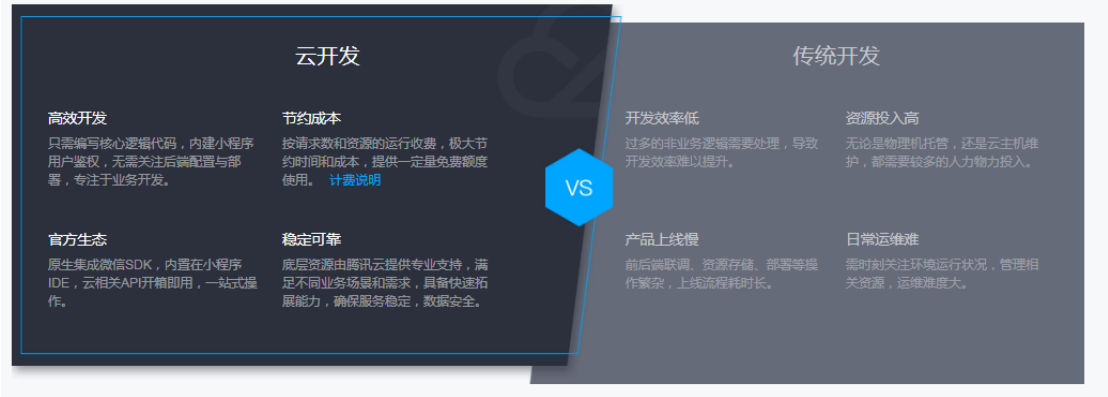
云函数

- 云端运行：无需采购、部署、运维传统硬件，节约人力及成本。
- 高效开发：每个函数单独运行、部署，上传代码后即可自动部署，提升了独立开发和迭代的速度。
- 弹性伸缩：根据请求量实现毫秒级实时弹性伸缩，函数未执行不产生任何费用
- 云调用：在云函数中使用云调用调用微信开放能力，无需换取 access_token
- 本地调试：云开发提供了云函数本地调试能力，方便开发者在本地进行云函数调试。

2.1.3 云开发优势

在传统开发模式中，开发者需要从小程序端通过额外引用的 SDK 请求后端，需要关心弹性伸缩、异地容灾、网络防护、安全加固等众多条件。

相比而言，云开发模式中，开发者从小程序端通过小程序原生接口请求云开发即可，只需要关心云数据库、存储以及云函数。

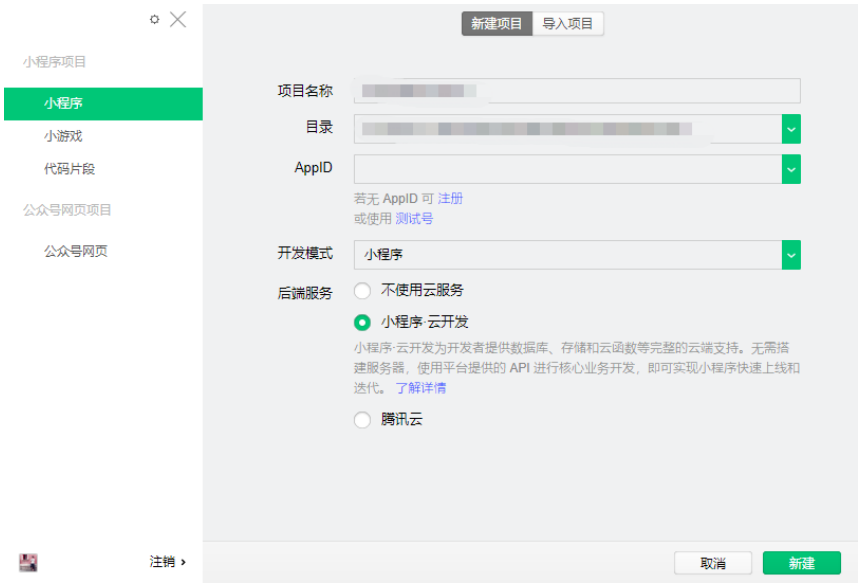


2.2 Hello World

2.2.1 初始化云开发小程序

新建项目选择一个空目录，填入小程序的 AppID，选择 “小程序·云开发”后端服务，点击创建即可得到一个展示云开发基础能力的示例小程序。该小程序与普通 QuickStart 小程序有以下不同需注意：

- 无游客模式、也不可以使用 测试号
- `project.config.json` 中增加了字段 `cloudfunctionRoot` 用于指定存放云函数的目录
- `cloudfunctionRoot` 指定的目录有特殊的图标
- 云开发能力从基础库 2.2.3 开始支持



2.2.2 开通小程序•云开发

创建了第一个云开发小程序后，在使用云开发能力之前需要先开通云开发。在开发者工具工具栏左侧，点击 “云开发” 按钮即可开通云开发。云开发开通后自动获得一套云开发环境，各个环境相互隔离，每个环境都包含独立的数据库实例、存储空间、云函数配置等资源。每个环境都有唯一的环境 ID 标识，初始创建的环境自动成为默认环境。

*注：AppID 首次开通云环境后，需等待大约 10 分钟方可正常使用云 API，在此期间官方后台服务正在做准备服务，如尝试在小程序中调用云 API 则会报 `cloud init error: { errMsg: "invalid scope" }` 的错误

2.2.3 熟悉云开发控制台

云开发提供了一个控制台用于可视化管理云资源。控制台包含以下几大模块。

- 概览：查看云资源的总体使用情况
- 用户管理：查看小程序的用户访问记录
- 数据库：管理数据库集合、记录、权限设置、索引设置

- 存储管理：管理云文件、权限设置
- 云函数：管理云函数、查看调用日志、监控记录
- 统计分析：查看云资源详细使用统计

在用户管理中会显示使用云能力的小程序的访问用户列表，默认以访问时间倒叙排列，访问时间的触发点是在小程序端调用 `wx.cloud.init` 方法，且其中的 `traceUser` 参数传值为 `true`。例：

```
wx.cloud.init({
  traceUser: true
})
```

2.3 云数据库

2.3.1 插入数据

可以通过在集合对象上调用 `add` 方法往集合中插入一条记录。还是用待办事项清单的例子，比如我们想新增一个待办事项：

```
db.collection('todos').add({
  // data 字段表示需新增的 JSON 数据
  data: {
    // _id: 'todo-identifiant-aleatoire', // 可选自定义 _id，在此处场景下用数据库自动分配的就可以了
    description: 'learn cloud database',
    due: new Date('2018-09-01'),
    tags: [
      'cloud',
      'database'
    ],
    // 为待办事项添加一个地理位置（113°E，23°N）
    location: new db.Geo.Point(113, 23),
    done: false
  },
  success(res) {
    // res 是一个对象，其中有 _id 字段标记刚创建的记录的 id
    console.log(res)
  }
})
```

2.3.2 删除数据

删除一条记录

对记录使用 `remove` 方法可以删除该条记录，比如：

```
db.collection('todos').doc('todo-identifiant-aleatoire').remove({
  success(res) {
    console.log(res.data)
  }
})
```

删除多条记录

如果需要更新多个数据，需在云函数进行操作。可通过 `where` 语句选取多条记录执行删除，只有有权限删除的记录会被删除。比如删除所有已完成的待办事项：

```
// 使用了 async await 语法
const cloud = require('wx-server-sdk')
const db = cloud.database()
const _ = db.command

exports.main = async (event, context) => {
  try {
    return await db.collection('todos').where({
      done: true
    }).remove()
  } catch (e) {
    console.error(e)
  }
}
```

2.3.3 查询数据

获取一个记录的数据

我们先来看看如何获取一个记录的数据，假设我们已有一个 ID 为 `todo-identifiant-aleatoire` 的在集合 todos 上的记录，那么我们可以通过在该记录的引用调用 `get` 方法获取这个待办事项的数据：

```
db.collection('todos').doc('todo-identifiant-aleatoire').get({
  success(res) {
    // res.data 包含该记录的数据
    console.log(res.data)
  }
})
```

获取多个记录的数据

我们也可以一次性获取多条记录。通过调用集合上的 `where` 方法可以指定查询条件，再调用 `get` 方法即可只返回满足指定查询条件的记录，比如获取用户的所有未完成的待办事项：

```
db.collection('todos').where({
  _openid: 'user-open-id',
  done: false
})
  .get({
    success(res) {
    // res.data 是包含以上定义的两条记录的数组
      console.log(res.data)
    }
  })
```

获取一个集合的数据

如果要获取一个集合的数据，比如获取 todos 集合上的所有记录，可以在集合上调用 `get` 方法获取，但通常不建议这么使用，在小程序中我们需要尽量避免一次性获取过量的数据，只应获取必要的数据。为了防止误操作以及保护小程序体验，小程序端在获取集合数据时服务器一次默认并且最多返回 20 条记录，云函数端这个数字则是 100。开发者可以通过 `limit` 方法指定需要获取的记录数量，但小程序端不能超过 20 条，云函数端不能超过 100 条。

```
db.collection('todos').get({
  success(res) {
    // res.data 是一个包含集合中有权限访问的所有记录的数据，不超过 20 条
    console.log(res.data)
  }
})
```

2.3.4 更新数据

局部更新

使用 `update` 方法可以局部更新一个记录或一个集合中的记录，局部更新意味着只有指定的字段会得到更新，其他字段不受影响。

比如我们可以用以下代码将一个待办事项置为已完成：

```
db.collection('todos').doc('todo-identifiant-aleatoire').update({
  // data 传入需要局部更新的数据
  data: {
    // 表示将 done 字段置为 true
    done: true
  },
  success(res) {
    console.log(res.data)
  }
})
```

除了用指定值更新字段外，数据库 API 还提供了一系列的更新指令用于执行更复杂的更新操作，更新指令可以通过 `db.command` 取得：

更新指令	说明
set	设置字段为指定值
remove	删除字段
inc	原子自增字段值
mul	原子自乘字段值
push	如字段值为数组，往数组尾部增加指定值
pop	如字段值为数组，从数组尾部删除一个元素
shift	如字段值为数组，从数组头部删除一个元素
unshift	如字段值为数组，往数组头部增加指定值

比如我们可以将一个待办事项的进度 +10%：

```
const _ = db.command
db.collection('todos').doc('todo-identifiant-aleatoire').update({
  data: {
    // 表示指示数据库将字段自增 10
    progress: _.inc(10)
  },
  success(res) {
    console.log(res.data)
  }
})
```

替换更新

如果需要替换更新一条记录，可以在记录上使用 `set` 方法，替换更新意味着用传入的对象替换指定的记录：

```
const _ = db.command
db.collection('todos').doc('todo-identifiant-aleatoire').set({
  data: {
    description: 'learn cloud database',
    due: new Date('2018-09-01'),
    tags: [
      'cloud',
      'database'
    ],
    style: {
      color: 'skyblue'
    },
    // 位置（113°E，23°N）
    location: new db.Geo.Point(113, 23),
    done: false
  },
  success(res) {
    console.log(res.data)
  }
})
```

2.3.5 索引管理

建立索引是保证数据库性能、保证小程序体验的重要手段。我们应为所有需要成为查询条件的字段建立索引。建立索引的入口在控制台中，可分别对各个集合的字段添加索引。

单字段索引

对需要作为查询条件筛选的字段，我们可以创建单字段索引。如果需要对嵌套字段进行索引，那么可以通过 "点表示法" 用点连接起嵌套字段的名称。比如我们需要对如下格式的记录中的 color 字段进行索引时，可以用 `style.color` 表示。

```
{
  "_id": "",
  "style": {
    "color": ""
  }
}
```

在设置单字段索引时，指定排序为升序或降序并没有关系。在需要对索引字段按排序查询时，数据库能够正确的对字段排序，无论索引设置为升序还是降序。

组合索引

组合索引即一个索引包含多个字段。当查询条件使用的字段包含在索引定义的所有字段或前缀字段里时，会命中索引，优化查询性能。索引前缀即组合索引的字段中定义的前 1 到多个字段，如有在 A, B, C 三个字段定义的组合索引 `A，B，C`，那么 `A` 和 `A，B` 都属于该索引的前缀。

组合索引具有以下特点：

1. 字段顺序决定索引效果

定义组合索引时，多个字段间的顺序不同是会有不同的索引效果的。比如对两个字段 A 和 B 进行索引，定义组合索引为 `A，B` 与定义组合索引为 `B，A` 是不同的。当定义组合索引为 `A，B` 时，索引会先按 A 字段排序再按 B 字段排序。因此当组合索引设为 `A，B` 时，即使我们没有单独对字段 A 设立索引，但对字段 A 的查询可以命中 `A，B` 索引。需要注意的是，此时对字段 B 的查询是无法命中 `A，B` 索引的，因为 B 不属于索引 `A，B` 的前缀之一。

2. 字段排序决定排序查询是否可以命中索引

加入我们对字段 A 和 B 设置以下索引：

A：升序
B：降序

那么当我们查询需要对 A, B 进行排序时，可以指定排序结果为 A 升序 B 降序或 A 降序 B 升序，但不能指定为 A 升序 B 升序或 A 降序 B 降序。

唯一性限制

创建索引时可以指定增加唯一性限制，具有唯一性限制的索引会要求被索引集合不能存在被索引字段值都相同的两个记录。即对任意具有唯一性限制的索引 `I`，假设其索引字段为 `<F1, F2, ..., Fn>`，则对集合 `S` 中任意的两个记录 `R1` 和 `R2`，必须满足条件 `R1.F1 != R2.F1 && R1.F2 != R2.F2 && ... && R1.Fn != R2.Fn`。需**特别注意**的是，假如记录中不存在某个字段，则对索引字段来说其值默认为 `null`，如果索引有唯一性限制，则不允许存在两个或以上的该字段为空 / 不存在该字段的记录。

在创建索引的时候索引属性选择 `唯一` 即可添加唯一性限制。

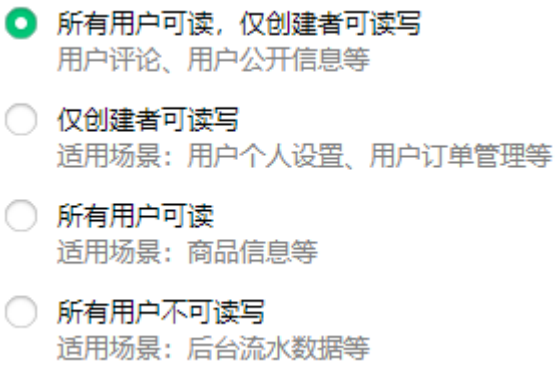
2.3.6 权限管理

数据库的权限分为小程序端和管理端，管理端包括云函数端和控制台。小程序端运行在小程序中，读写数据库受权限控制限制，管理端运行在云函数上，拥有所有读写数据库的权限。云控制台的权限同管理端，拥有所有权限。小程序端操作数据库应有严格的安全规则限制。

初期我们对操作数据库开放以下几种权限配置，每个集合可以拥有一种权限配置，权限配置的规则是作用在集合的每个记录上的。出于易用性和安全性的考虑，云开发为云数据库做了小程序深度整合，在小程序中创建的每个数据库记录都会带有该记录创建者（即小程序

用户）的信息，以 `_openid` 字段保存用户的 `openid` 在每个相应用户创建的记录中。因此，权限控制也相应围绕着一个用户是否应该拥有权限操作其他用户创建的数据展开。

以下按照权限级别从宽到紧排列如下：



在设置集合权限时应谨慎设置，防止出现越权操作。

2.4 云存储

2.4.1 上传文件

在小程序端可调用 `wx.cloud.uploadFile` 方法进行上传：

```
wx.cloud.uploadFile({
  cloudPath: 'example.png', // 上传至云端的路径
  filePath: '', // 小程序临时文件路径
  success: res => {
    // 返回文件 ID
    console.log(res.fileID)
  },
  fail: console.error
})
```

上传成功后会获得文件唯一标识符，即文件 ID，后续操作都基于文件 ID 而不是 URL。

2.4.2 获取临时链接

可以根据文件 ID 换取临时文件网络链接，文件链接有有效期为两个小时：

```
wx.cloud.getTempFileURL({
  fileList: ['cloud://xxx.png'],
  success: res => {
    // fileList 是一个有如下结构的对象数组
    // [{
    //    fileID: 'cloud://xxx.png', // 文件 ID
    //    tempFileURL: '', // 临时文件网络链接
    //    maxAge: 120 * 60 * 1000, // 有效期
    // }]
    console.log(res.fileList)
  },
  fail: console.error
})
```

2.4.3 权限管理

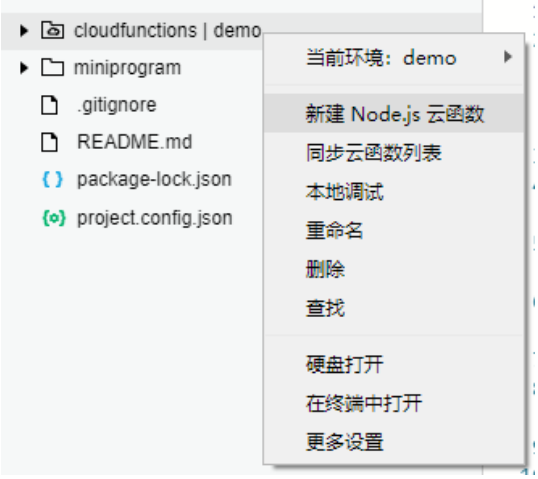
和云数据库一样，云存储再云开发控制台中也拥有类似的权限管理功能。以下按照权限级别从宽到紧排列如下：

2.5 云函数

2.5.1 创建、安装依赖与部署

创建云函数

我们在云函数根目录上右键，在右键菜单中，可以选择创建一个新的 Node.js 云函数，我们将该云函数命名。



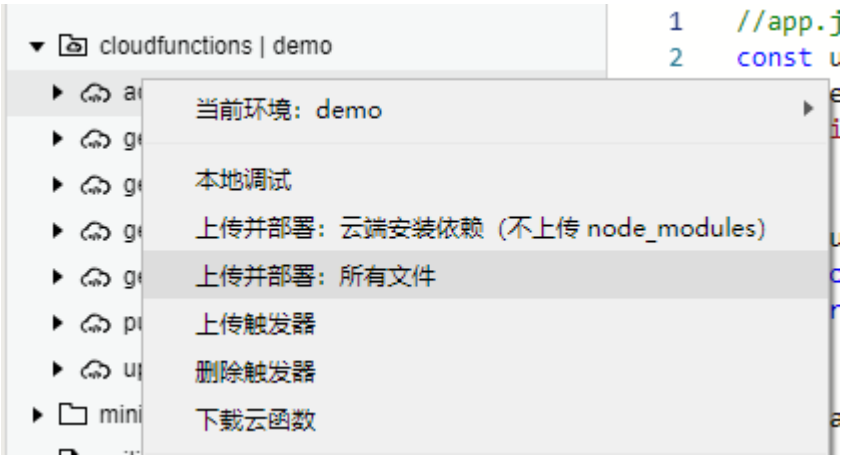
安装依赖

在云函数新建好后，开发者工具会在本地创建出云函数目录和入口 index.js 文件，同时在线上环境中创建出对应的云函数。创建成功后，工具会提示是否立即本地安装依赖，确定后工具会自动安装 wx-server-sdk。

若未安装成功，需要前往终端使用 npm install 命令进行安装。

上传与部署云函数

右键需要上传的云函数，点击“上传并部署”即可。



2.5.2 调用云函数

云函数部署完成后，我们可以在小程序中这样调用它：

```
wx.cloud.callFunction({
  // 云函数名称
  name: 'add',
  // 传给云函数的参数
  data: {
    a: 1,
    b: 2,
  },
  success(res) {
    console.log(res.result.sum) // 3
  },
  fail: console.error
})
```

2.5.3 获取用户登录态

云开发的云函数的独特优势在于与微信登录鉴权的无缝整合。当小程序端调用云函数时，云函数的传入参数中会被注入小程序端用户的 openid，开发者无需校验 openid 的正确性，因为微信已经完成了这部分鉴权，开发者可以直接使用该 openid。与 openid 一起同时注入云函数的还有小程序的 appid。

从小程序端调用云函数时，开发者可以在云函数内使用 wx-server-sdk 提供的 getWXContext 方法获取到每次调用的上下文（appid、openid 等），无需维护复杂的鉴权机制，即可获取天然可信任的用户登录态（openid）。可以写这么一个云函数进行：

```
// index.js
const cloud = require('wx-server-sdk')
exports.main = (event, context) => {
  // 这里获取到的 openId、 appId 和 unionId 是可信的，注意 unionId 仅在满足 unionId 获取条件时返回
  const {OPENID, APPID, UNIONID} = cloud.getWXContext()

  return {
    OPENID,
    APPID,
    UNIONID,
  }
}
```

2.5.4 云调用

功能简介

云调用是云开发提供的基于云函数使用小程序开放接口的能力。云调用需要在云函数中通过 `wx-server-sdk` 使用。在云函数中使用云调用调用服务端接口无需换取 `access_token`，只要是在从小程序端触发的云函数中发起的云调用都经过微信自动鉴权，可以在登记权限后直接调用如发送模板消息等开放接口。

支持 API 列表

目前大部分小程序服务端 API 均支持云调用，可以使用微信扫码查看支持的接口列表。如果接口支持云调用，接口名称旁边会展示“云调用”的标签。

服务端 API 接口

使用方法

在开发之前，需要**配置云调用权限**，每个云函数需要声明其会使用到的接口，否则无法调用，声明的方法是在云函数目录下的 `config.json`（如无需新建）配置文件的 `permissions.openapi` 字段中增加要调用的接口名，值必须为所需调用的服务端接口名称。以下是一个示例的声明了使用发送模板消息接口的配置文件：

```
{
  "permissions": {
    "openapi": ["templateMessage.send"]
  }
}
```

各接口从属的类别名称和方法名称可以通过接口名称查看，接口名称均以 `<类别>.<方法>` 命名，如发送模板消息的接口名称是 templateMessage.send。下面是一个给自己发送模板消息的示例：

```
const cloud = require('wx-server-sdk')
cloud.init()
exports.main = async (event, context) => {
  try {
    const result = await cloud.openapi.templateMessage.send({
      touser: cloud.getWXContext().OPENID, // 通过 getWXContext 获取 OPENID
      page: 'index',
      data: {
        keyword1: {
          value: '339208499'
        },
        keyword2: {
          value: '2015年01月05日 12:30'
        },
        keyword3: {
          value: '腾讯微信总部'
        },
        keyword4: {
          value: '广州市海珠区新港中路397号'
        }
      },
      templateId: 'TEMPLATE_ID',
      formId: 'FORMID',
      emphasisKeyword: 'keyword1.DATA'
    })
    // result 结构
    // { errCode: 0, errMsg: 'openapi.templateMessage.send:ok' }
    return result
  } catch (err) {
    // 错误处理
    // err.errCode !== 0
    throw err
  }
}
```

2.6 更多云开发能力

如想了解更多云开发知识以及能力，例如云函数定时触发器、TCBRouter 等功能的使用，可查阅云开发文档及云开发团队的 Github 主页。

云开发文档：

https://developers.weixin.qq.com/miniprogram/dev/wxcloud/basis/getting-started.html

云开发 Github：https://github.com/TencentCloudBase

云开发文档

关注腾讯云云开发公众号

上小程序，用云开发