

阿里云 小程序云

小程序Serverless开发指南

文档版本：20200327

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。
2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。
3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。
5. 阿里云文档中所有内容，包括但不限于图片、架构设计、页面布局、文字描述，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。
6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	 禁止： 重置操作将丢失用户配置数据。
	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	 警告： 重启操作将导致业务中断，恢复业务时间约十分钟。
	用于警示信息、补充说明等，是用户必须了解的内容。	 注意： 权重设置为0，该服务器不会再接受新请求。
	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	 说明： 您也可以通过按Ctrl + A选中全部文件。
>	多级菜单递进。	单击设置 > 网络 > 设置网络类型。
粗体	表示按键、菜单、页面名称等UI元素。	在结果确认页面，单击确定。
Courier字体	命令。	执行 <code>cd /d C:/window</code> 命令，进入Windows系统文件夹。
##	表示参数、变量。	<code>bae log list --instanceid Instance_ID</code>
[]或者[a b]	表示可选项，至多选择一个。	<code>ipconfig [-all -t]</code>
{ }或者{a b}	表示必选项，至多选择一个。	<code>switch {active stand}</code>

目录

法律声明.....	I
通用约定.....	I
1 什么是小程序Serverless?	1
2 基础入门.....	3
2.1 入门概览.....	3
2.2 开发支付宝小程序.....	4
2.3 开发钉钉小程序.....	14
2.4 开发微信小程序.....	24
3 服务空间管理.....	33
3.1 创建服务空间.....	33
3.2 删除服务空间.....	34
4 云函数.....	35
4.1 概述.....	35
4.2 使用云函数.....	35
4.3 传递参数.....	38
4.4 打印日志.....	38
4.5 调用数据和文件存储服务.....	39
4.6 调用HTTP服务.....	39
4.7 开启云函数HTTP触发功能.....	39
4.8 控制台操作指南.....	42
4.8.1 新建云函数.....	42
4.8.2 部署代码.....	43
4.8.3 查看日志.....	44
5 数据存储.....	45
5.1 数据结构设计.....	45
5.2 使用数据存储服务.....	47
5.3 数据权限管理.....	47
5.4 控制台操作指南.....	50
5.4.1 创建数据表.....	50
5.4.2 添加记录.....	50
5.4.3 设置数据表权限.....	51
5.4.4 使用高级模式.....	52
5.4.5 设置数据库索引.....	52
5.5 数据管理最佳实践.....	53
5.5.1 管理数据集合.....	53
5.5.2 原子操作.....	57
6 文件存储.....	59
6.1 使用文件存储服务.....	59
6.2 文件管理最佳实践.....	60

6.3 文件权限管理.....	62
6.4 控制台操作指南.....	64
6.4.1 上传文件.....	64
6.4.2 删除图片.....	66
6.4.3 设置文件权限.....	67
7 扩展能力.....	68
7.1 云调用概述.....	68
7.2 使用云调用.....	69
7.3 公共数据结构.....	71
8 统计分析.....	73
8.1 集成统计分析服务.....	73
8.1.1 支付宝小程序集成.....	73
8.1.2 微信小程序集成.....	77
8.2 查看统计数据.....	81
8.3 基本统计指标说明.....	81
8.4 概况分析.....	82
8.4.1 整体趋势.....	82
8.4.2 实时统计.....	85
8.5 用户分析.....	86
8.5.1 新增用户.....	86
8.5.2 活跃用户.....	87
8.5.3 活跃用户.....	89
8.6 活跃用户.....	91
8.6.1 活跃用户.....	93
8.7 留存用户.....	95
8.7.1 留存用户.....	96
8.8 渠道.....	97
8.8.1 时段详情.....	97
8.8.2 渠道列表.....	98
9 跨端配置指南.....	100
9.1 uni-app跨平台开发扩展使用教程.....	100
9.2 开发多端小程序教程.....	104
9.3 开发者工具Vue扩展使用教程.....	110
10 设置小程序.....	113
10.1 添加支付宝小程序.....	113
10.1.1 添加支付宝小程序.....	113
10.1.2 生成支付宝RSA密钥.....	114
10.2 添加微信小程序.....	116
11 SDK 文档.....	118
11.1 变更历史.....	118
11.2 安装客户端SDK2.2版本.....	118
11.3 安装客户端SDK2.2版本.....	122
11.4 安装客户端SDK2.0版本.....	125

11.5 安装客户端SDK1.0版本.....	127
12 客户端API参考.....	128
12.1 云函数.....	128
12.1.1 invoke.....	128
12.2 数据存储.....	128
12.2.1 aggregate.....	129
12.2.2 count.....	130
12.2.3 deleteOne.....	131
12.2.4 deleteMany.....	132
12.2.5 distinct.....	132
12.2.6 find.....	133
12.2.7 findOne.....	134
12.2.8 findOneAndDelete.....	135
12.2.9 findOneAndReplace.....	135
12.2.10 findOneAndUpdate.....	136
12.2.11 replaceOne.....	137
12.2.12 insertMany.....	138
12.2.13 insertOne.....	139
12.2.14 updateMany.....	139
12.2.15 updateOne.....	140
12.3 文件存储.....	141
12.3.1 uploadFile.....	141
12.3.2 deleteFile.....	142
12.4 用户管理.....	143
12.4.1 getInfo.....	143
12.4.2 authorize.....	145
13 钉钉隐藏暂时.....	147

1 什么是小程序Serverless?

阿里云小程序Serverless提供包括云函数、数据存储、文件存储等一整套后端服务。开发者通过API方式即可获取云函数、数据存储、文件存储、音视频、图像处理等服务，不需要关心服务器或底层运维设施，可以更专注于代码和业务本身。

Serverless是一种构建和管理基于微服务架构的完整流程。计算资源作为服务而不是服务器的概念出现，开发者不再需要考虑服务器的问题，可以更专注于代码和业务，更快速地开发软件。Serverless通常包含BaaS和FaaS服务：

- **BaaS：后端即服务（Backend as a Service）**，向开发者提供后端基础功能，通过API方式进行调用。例如文件存储、数据库、消息等。
- **FaaS：函数即服务（Function as a Service）**，允许开发者直接将程序部署到平台上，以函数作为最小单元，通过API方式进行调用。

小程序Serverless功能

小程序Serverless提供以下三种功能：

- **数据存储**

数据存储服务是基于MongoDB托管在云端的数据库，数据以JSON格式存储。数据库中的每条记录都是一个JSON格式的对象。一个数据库可以有多个集合（相当于关系型数据中的表）。

- **文件存储**

文件存储服务支持文本、图片和其他由用户生成的内容存储到云端。开发者可以在小程序端和控制台使用云存储功能。图片上传成功后，系统会自动生成一个资源链接。开发者可以在小程序中使用该图片地址。

- **云函数**

云函数服务支持使用Node.js进行开发。作为开发者，您可以可将代码提交到云端运行，在客户端使用小程序提供的API进行调用。您还可以在云函数中直接通过API调用数据存储和文件存储的服务资源。

小程序Serverless优势

小程序Serverless具有以下优势：

- **无运维**

开发者只需专注业务开发，无需理解后端服务的运维配置。

- **低成本**

按实际使用的资源和调用情况计费，有效降低了运维成本和研发成本。

- **高可用**

底层能力由阿里云存储、数据库团队提供支持，支持弹性扩容，同时提供可用性保障。

- **一云多端**

适配多种平台的小程序端框架，一套代码多端使用。

相关文档

- [#unique_4](#)
- [#unique_5](#)
- [使用数据存储服务](#)
- [使用文件存储服务](#)
- [安装客户端SDK1.0版本](#)

2 基础入门

2.1 入门概览

开发者通过API方式即可获取小程序Serverless提供的云函数、数据存储、文件存储等服务，不需要关心服务器或底层运维设施，可以专注于小程序的业务开发。

小程序Serverless提供了以下基础能力：

- **云函数**：支持在云端运行代码，开发者只需编写自身业务逻辑代码。更多信息，请参见[使用云函数](#)。
- **云数据库**：提供数据存储服务。云数据库是基于MongoDB托管在云端的数据库，数据以JSON格式存储。作为开发者，您可以在客户端内直接操作数据，也可以在云函数中读写数据。更多信息，请参见[使用数据存储服务](#)。
- **文件存储**：文件存储服务支持文本、图片和其他由用户生成的内容存储到云端。更多信息，请参见[使用文件存储服务](#)。
- **云调用**：基于云函数使用小程序开放接口的能力，包括服务端调用、获取开放数据等。更多信息，请参见[使用云调用](#)。

开发流程

使用小程序Serverless开发小程序的基本流程如下：



1. 登录要开发的小程序平台创建小程序应用并获取App ID。
2. 开通小程序Serverless服务，按需创建云资源。
3. 使用小程序Serverless SDK进行开发。

2.2 开发支付宝小程序

本教程介绍如何使用阿里云小程序Serverless服务开发一个简单的支付宝小程序。

教程介绍

本教程以一个入门小程序为例，了解学习如何在支付宝IDE中使用小程序Serverless服务，开发一个支付宝小程序。您只需要根据本教程的引导开通Serverless服务，配置好小程序后，就可以按照小程序中提供的示例调试这些功能。



说明：

本教程中使用的是小程序Serverless SDK 2.x版本，在安装SDK后，需要调用user.authorize进行授权。详细信息请参见[在支付宝小程序中使用SDK](#)。

准备工作

在开始前，确保您已经完成以下准备工作：

- 注册支付宝开发者账号。

如果您尚未注册支付宝小程序账号，使用支付宝账号登录[蚂蚁金服开放平台](#)，并完成开发者身份注册。详细信息，请参见[开发者入驻说明](#)。

- 下载并安装支付宝小程序开发者工具。

详细信息，请参见[下载说明](#)。

- 开通小程序云服务。

在首次使用小程序云服务前，您需要开通小程序云服务。详细信息，请参见[#unique_15](#)。

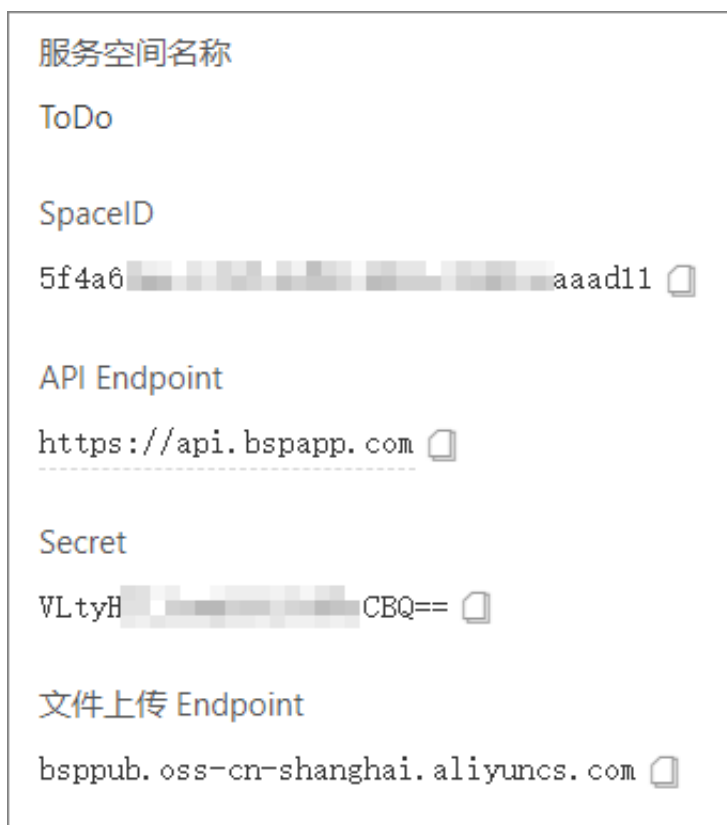
步骤一：创建小程序云服务空间

小程序开发中创建的云资源会和服务空间进行关联。每个服务空间都有一个全局唯一的space ID。在使用云资源时，都通过这个space ID进行关联。

参考以下步骤，创建服务空间并设置数据库权限：

1. 使用阿里云账号登录[小程序云控制台](#)。
2. 在左侧导航栏，单击小程序Serverless > 服务空间管理。
3. 单击创建服务空间，输入服务空间名称，然后单击确定。

4. 创建成功后，单击详情查看space ID、secret和endpoint信息。

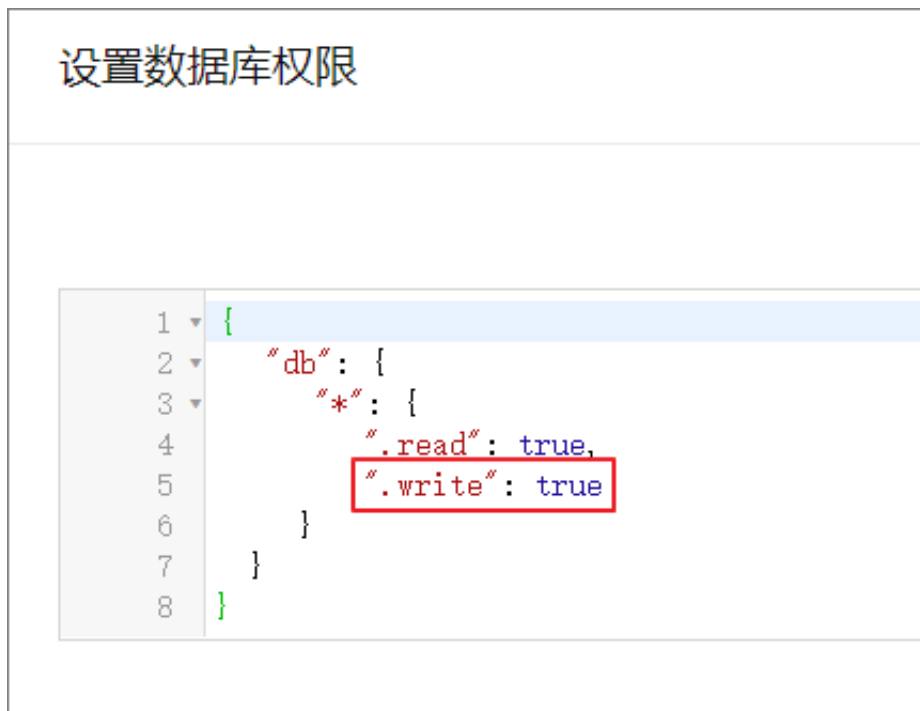


5. 单击小程序Serverless > 云数据库，然后在页面右上角选择已创建的服务空间。
6. 单击  修改数据库权限，将write权限修改为true。



说明:

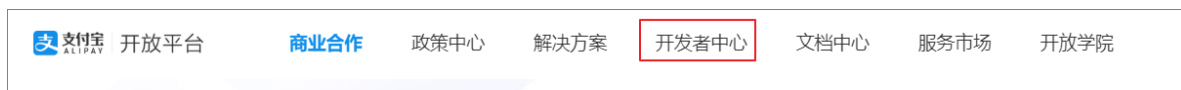
为了保证数据安全，默认数据库访问受限。您可以根据需要设置数据库权限，更多信息，请参见[数据库权限管理](#)。



步骤二：创建支付宝小程序应用

参考以下步骤，在蚂蚁金服开放平台创建支付宝小程序：

1. 使用支付宝账号登录[蚂蚁金服开放平台](#)进入开发者中心。



2. 选择小程序应用，然后单击创建应用并根据引导创建一个小程序应用。



3. 在左侧导航栏单击设置。

4. 单击开发设置，在开发信息下的接口加签方式区域配置支付宝公钥。

您可以通过RSA密钥生成工具一键生成小程序密钥。接口加签方式主要用于签名和验证。其中私钥由开发者保存，用于签名；公钥可对外公开，用于验证签名。开发者需要保存自己的私钥，通过支付宝提供的公钥进行通信来保证信息传输的完整性以及发送者身份的真实性。详细信息，请参见[生成支付宝RSA密钥](#)。



注意：

密钥格式必须为PKCS8（Java适用）。



5. 单击服务器域名白名单区域的添加，然后输入步骤一创建的服务空间的文件上传endpoint。



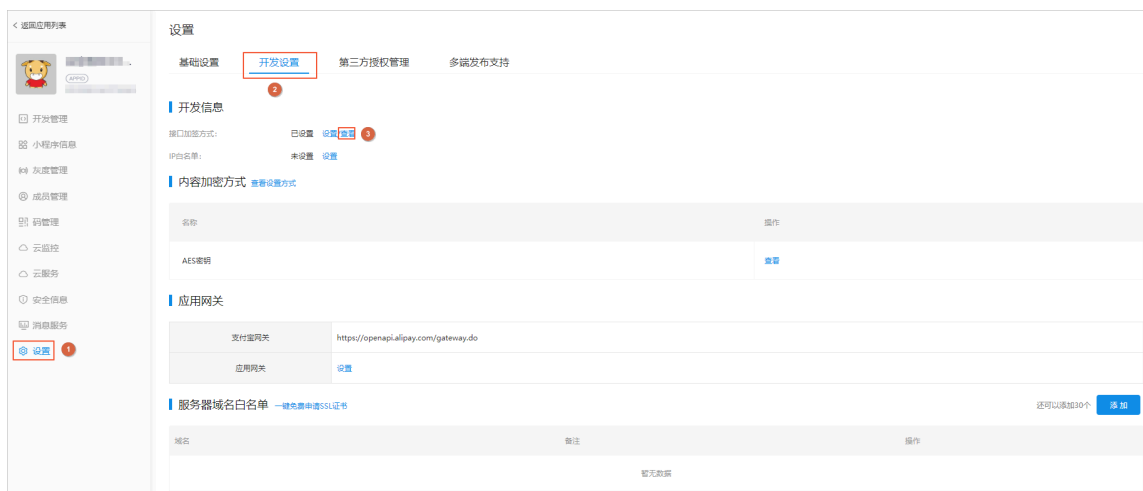
步骤三：添加支付宝小程序

在开发前，您需要将支付宝小程序信息添加到阿里云小程序Serverless控制台。

完成以下操作，添加支付宝小程序信息：

1. 登录[小程序云控制台](#)。

2. 在左侧导航栏，单击小程序Serverless > 设置。
3. 单击支付宝页签，然后单击添加密钥。
4. 输入AppID，即在步骤二中您创建的支付宝小程序的AppID。
5. 输入小程序私钥，即您在步骤二中生成的小程序密钥对中的私钥。
6. 输入支付宝公钥。参考以下步骤查看公钥：
 - a) 登录蚂蚁金服开放平台。
 - b) 在支付宝小程序菜单栏单击设置，然后单击开发设置。
 - c) 在接口加签方式区域，单击查看。



- d) 复制支付宝公钥信息，然后粘贴到阿里云支付宝小程序的配置中。



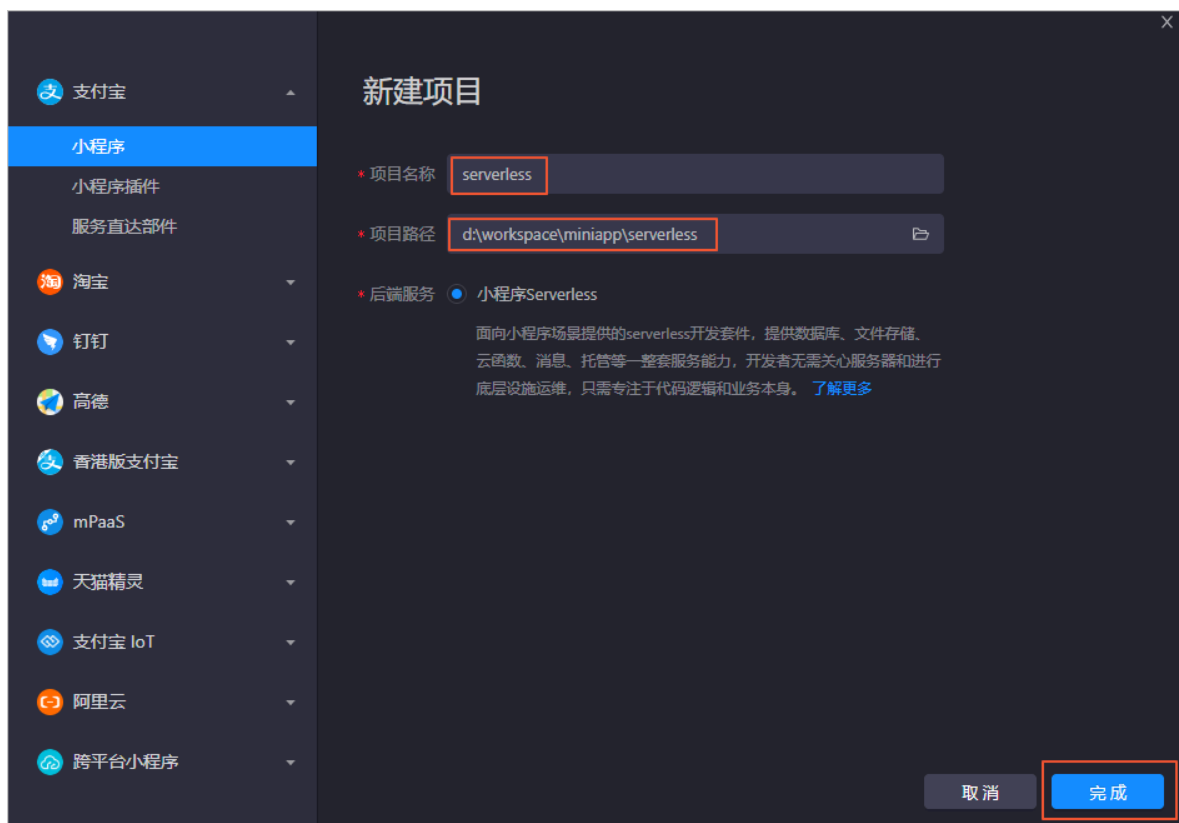
步骤四：开发支付宝小程序

本教程以一个小程序Serverless入门Demo为例。请参考以下步骤，开发支付宝小程序项目：

1. 打开**支付宝开发者工具**。
2. 在左侧导航栏选择**支付宝>小程序**，然后单击**模板选取**，最后在入门分类中选择**Serverless入门Demo模板**，单击**下一步**。



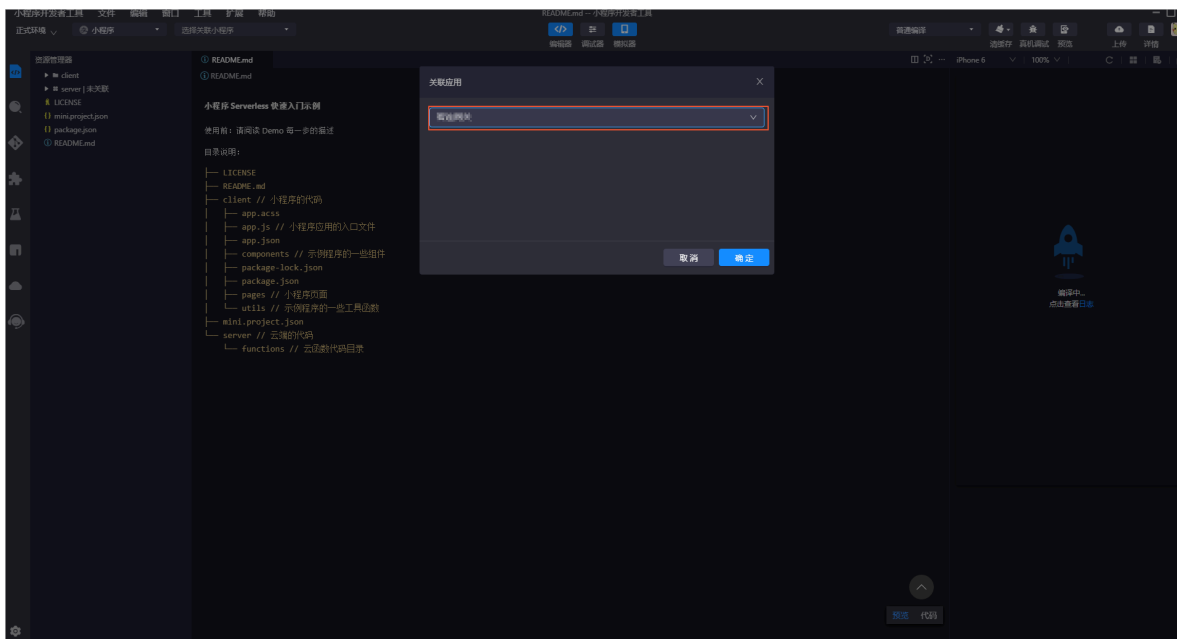
3. 设置项目名称和项目路径，然后单击完成。



4. 在IDE主窗口打开后会弹出二维码登录对话框，请使用手机支付宝扫描二维码登录IDE。



5. 接下来在弹出的关联应用对话框中，选择已创建的支付宝小程序，然后单击确定。

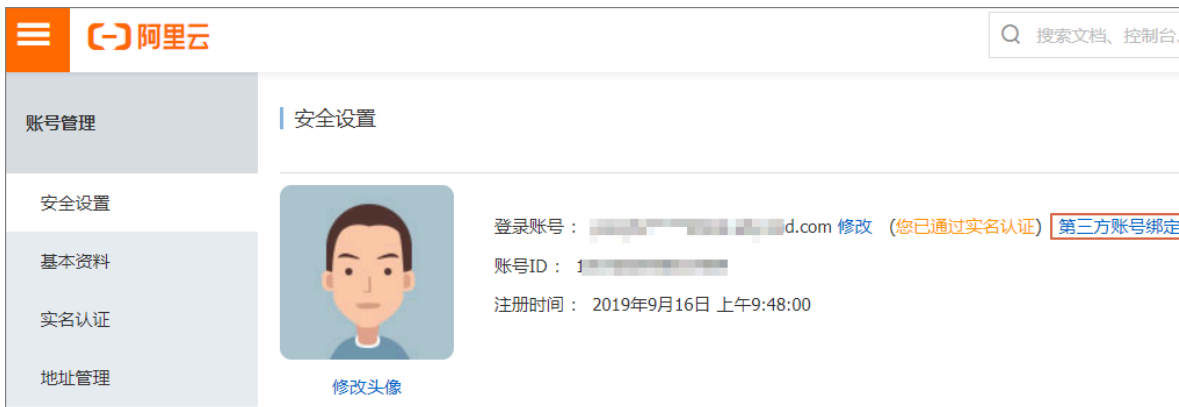


6. 当提示绑定阿里云账号时，单击前往绑定阿里云账号，然后在打开的验证页面选择一种账号验证方式并完成验证。或右键单击左侧的server|未关联目录，然后选择关联Serverless。

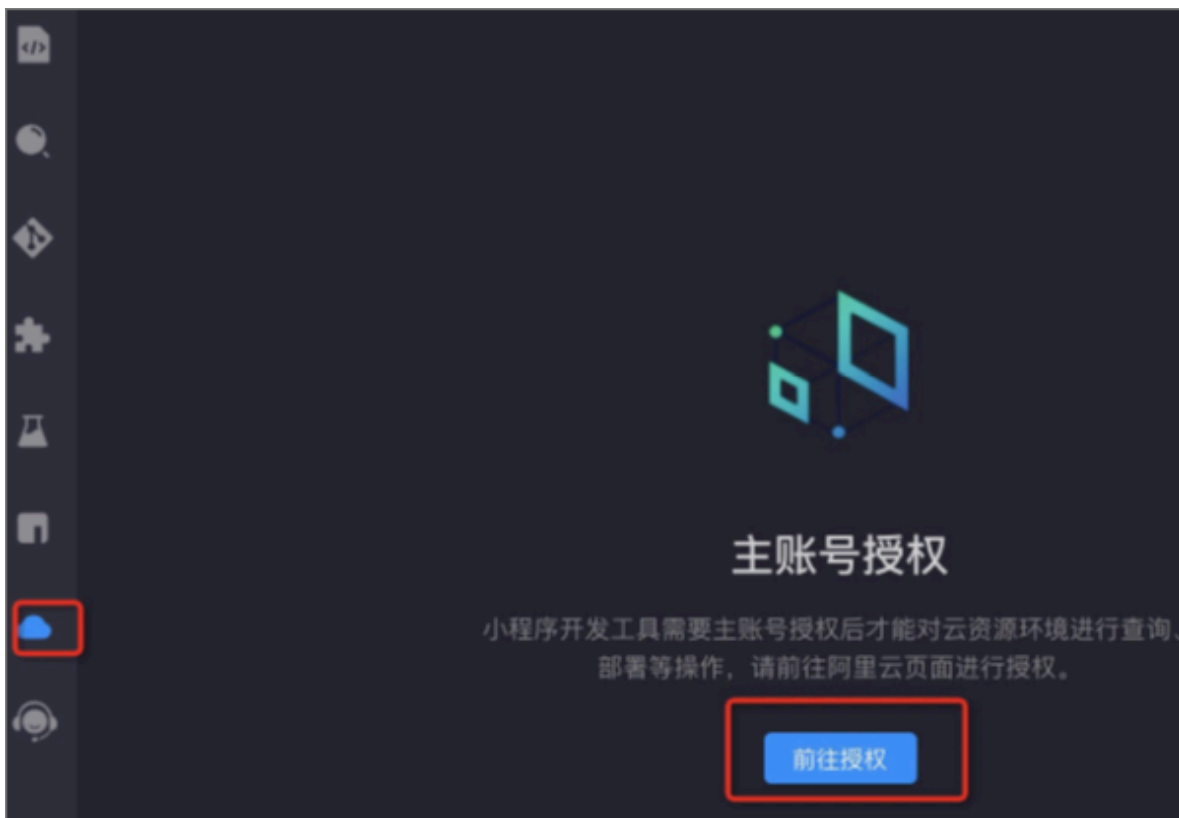


说明：

确保您要绑定的阿里云账号已关联支付宝账号。

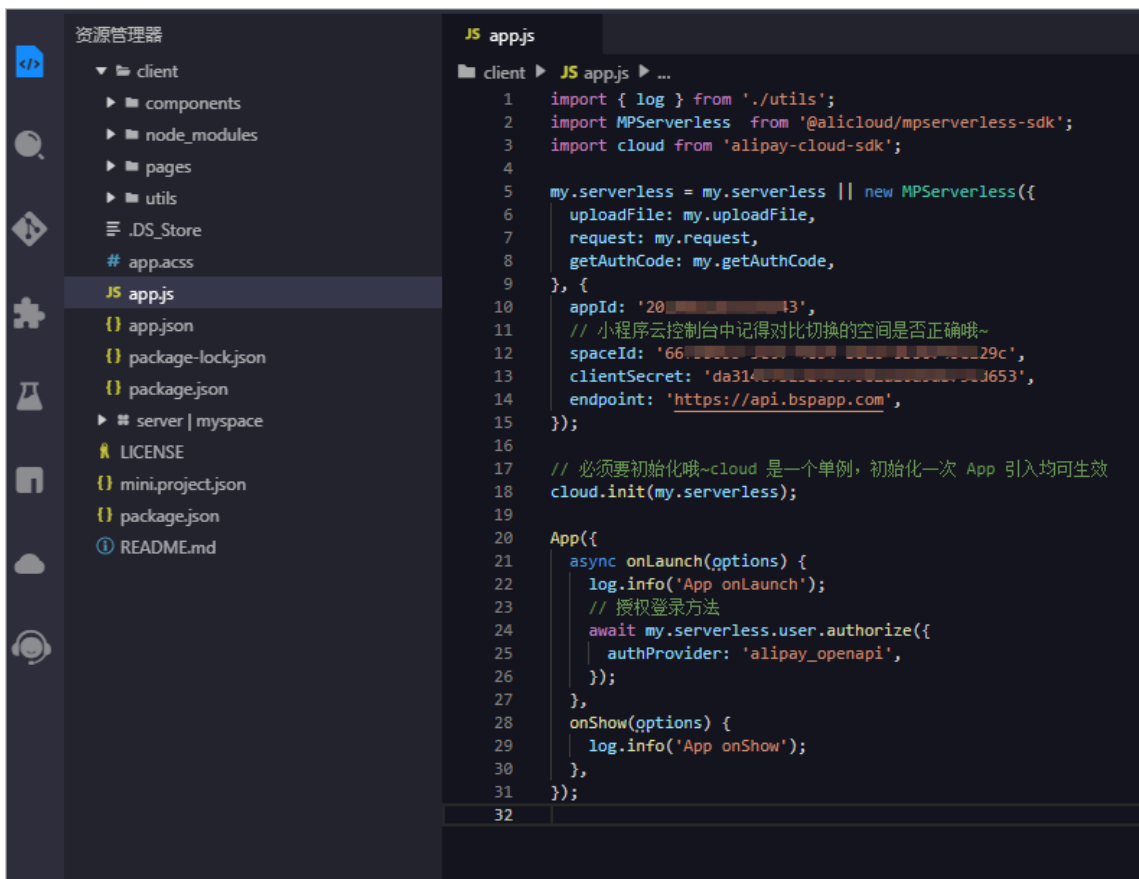


7. 单击左侧云图标，然后单击前往授权授权IDE访问您的阿里云云资源。



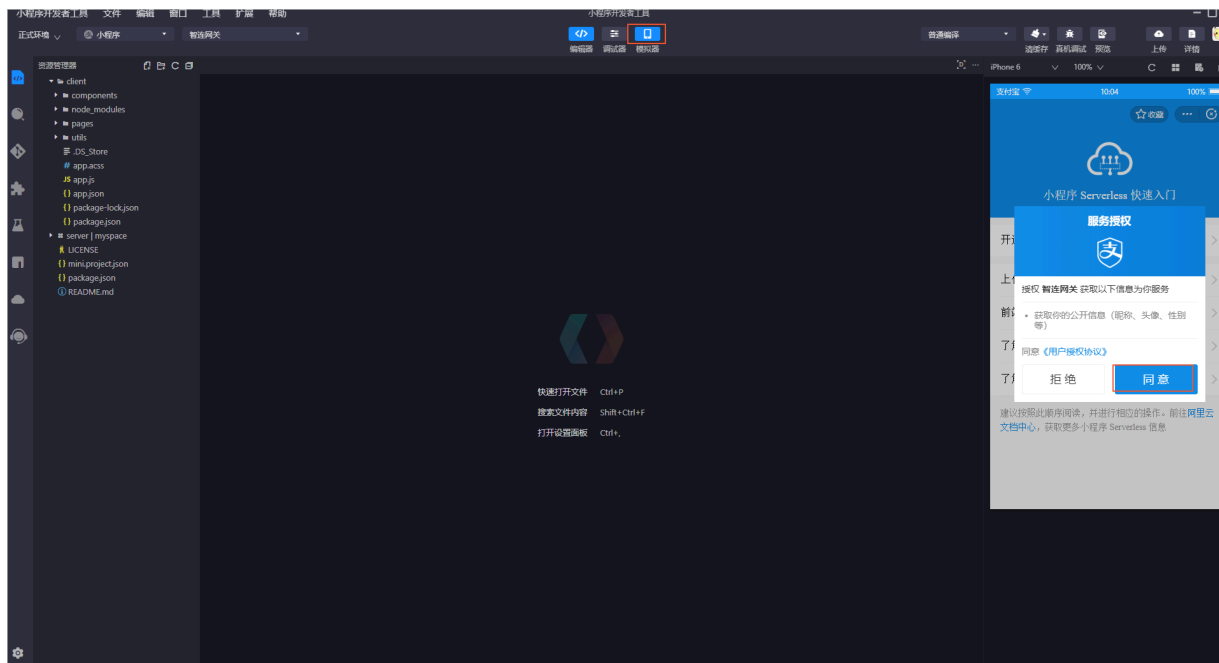
8. 在项目文件列表中找到`app.js`文件，配置以下信息，然后保存。

- appId是小程序的ID。您可以在蚂蚁金服开放平台的支付宝小程序页面查看App ID。
- spaceId、clientSecret和endpoint在小程序Serverless控制台创建服务空间后可以获得。详情参见[创建服务空间](#)。



步骤五：调试小程序

打开小程序模拟器，在弹出的服务授权框中仔细阅读《用户授权协议》后选择同意。



2.3 开发钉钉小程序

使用小程序Serverless服务开发一个简单的钉钉小程序。

教程介绍

本教程以一个可添加和删除待办事项的小程序为例，学习如何使用小程序Serverless SDK调用云数据库API，开发一个钉钉小程序。本教程使用的小程序示例代码的业务逻辑已经编写好，您只需要根据本教程的引导开通Serverless服务，配置好小程序，就可以运行该小程序了。



说明：

目前小程序Serverless服务仅支持类型为**第三方个人应用**的钉钉小程序。

准备工作

在开始前，确保您已经完成以下准备工作：

- 注册钉钉开发者账号。

如果您尚未注册钉钉小程序开发者账号，使用钉钉账号登录**钉钉开放平台**。

- 下载并安装小程序开发者工具。

详细信息，请参见**下载说明**。

- 开通小程序云服务。

在首次使用小程序云服务前，您需要开通小程序云服务。详细信息，请参见**#unique_15**。

步骤一：创建小程序云服务空间

小程序开发中创建的云资源会和服务空间进行关联。每个服务空间都有一个全局唯一的space ID。在使用云资源时，都通过这个space ID进行关联。

参考以下步骤，创建服务空间并设置数据库权限：

1. 使用阿里云账号登录[小程序云控制台](#)。
2. 在左侧导航栏，单击小程序Serverless > 服务空间管理。
3. 单击创建服务空间，输入服务空间名称，然后单击确定。
4. 创建成功后，单击详情查看space ID、secret和endpoint信息。

服务空间名称

ToDo

SpaceID

5f4a6[REDACTED]aaad11 

API Endpoint

https://api.bspapp.com 

Secret

VLtyH[REDACTED]CBQ== 

文件上传 Endpoint

bspapp.oss-cn-shanghai.aliyuncs.com 

5. 单击小程序Serverless > 云数据库，然后在页面右上角选择已创建的服务空间。
6. 单击  修改数据库权限，将write权限修改为true。



说明：

为了保证数据安全，默认数据库访问受限。您可以根据需要设置数据库权限，更多信息，请参见[数据权限管理](#)。

设置数据库权限

```
1 {  
2   "db": {  
3     "*": {  
4       ".read": true,  
5       ".write": true  
6     }  
7   }  
8 }
```

步骤二：创建钉钉小程序应用

参考以下步骤，在钉钉开放平台创建钉钉小程序：

1. 使用钉钉账号登录[钉钉开放平台](#)

2. 在钉钉开放平台，单击应用开发页签，然后在左侧导航栏选择第三方个人应用 > 小程序。



3. 在小程序页面，单击创建应用，输入小程序基本信息后，再单击创建应用。

4. 单击已创建的小程序，在应用首页页签，单击应用信息列的查看详情获取小程序的App ID及App Secret。

图 2-1: 应用详情



图 2-2: 小程序AppID和AppSecret



5. 单击设置安全域名页签，然后单击HTTP安全域名区域下的添加，加入阿里云的安全域名 aliyuncs.com 和 [步骤一：创建小程序云服务空间](#) 生成的空间 endpoint。



步骤三：添加钉钉小程序

在开发前，您需要在小程序Serverless服务中添加钉钉小程序。

完成以下操作，添加钉钉小程序：

1. 使用阿里云账号登录[小程序云控制台](#)。
2. 在左侧导航栏，选择小程序Serverless > 设置，然后单击钉钉。
3. 单击添加密钥，然后设置钉钉小程序的App ID和App Secret。



说明：

您可以在钉钉小程序详情页面获取App ID和App Secret。详情请参见[图 2-2：小程序AppID和AppSecret](#)。



步骤四：开发钉钉小程序

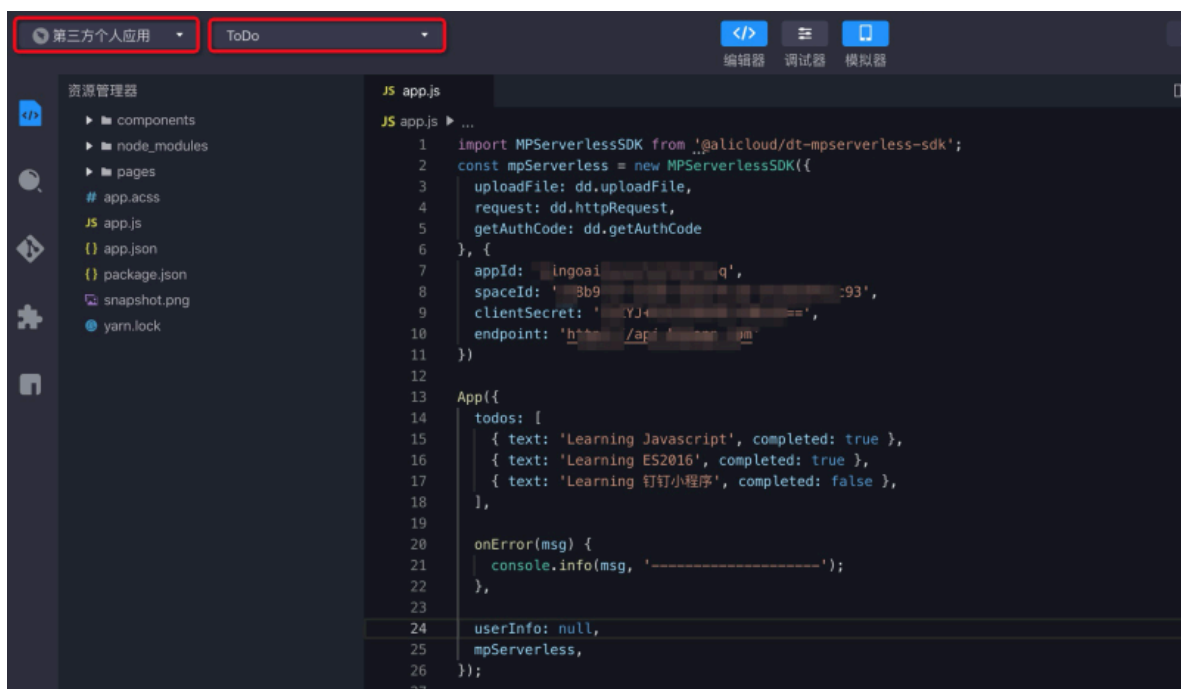
本教程以一个开发好的待办事项To Do小程序为例。在开始前，确保您已经了解了小程序的文件结构。详细信息，请参见[详细了解小程序文件结构](#)。

参考以下步骤，开发钉钉小程序：

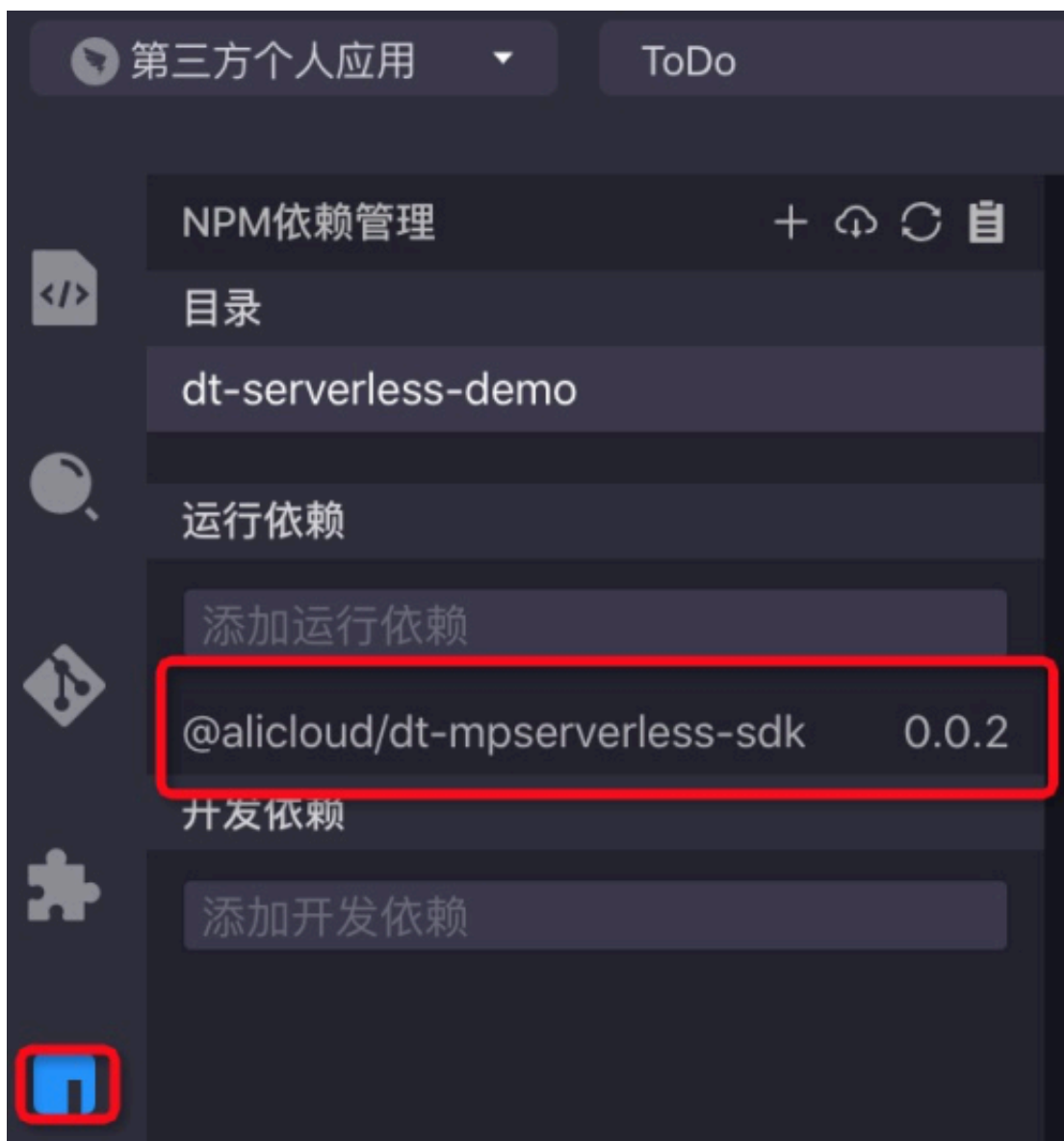
1. 单击[这里](#)下载To Do小程序示例项目文件，然后解压下载的文件。
2. 打开阿里云小程序开发工具，选择钉钉 > 第三方个人应用，然后单击打开项目，然后选择已解压的项目文件。



3. 关联**步骤三**中添加的钉钉小程序，然后使用钉钉账号登录IDE，并完成授权。

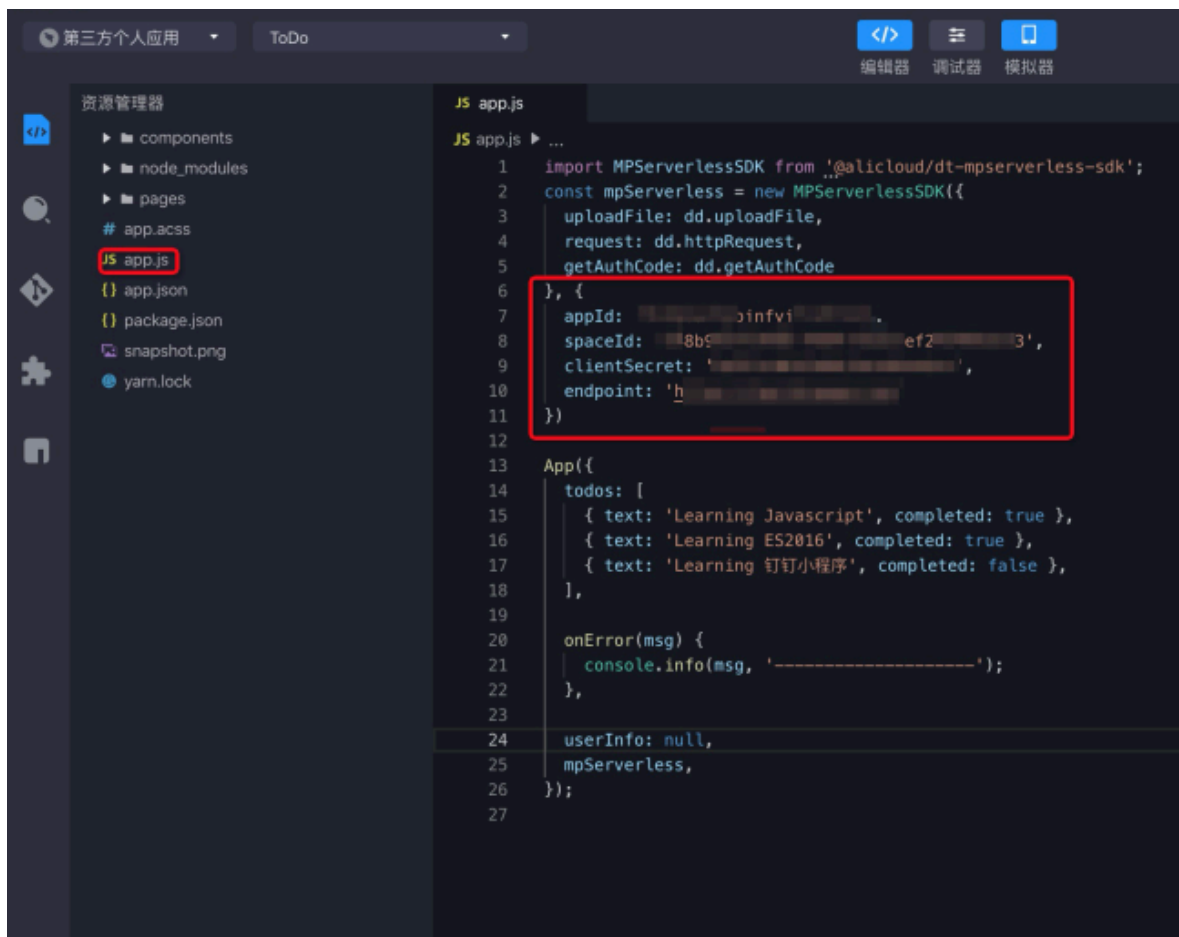


4. 在左侧菜单栏，单击NPM模块，然后选择安装小程序云SDK。



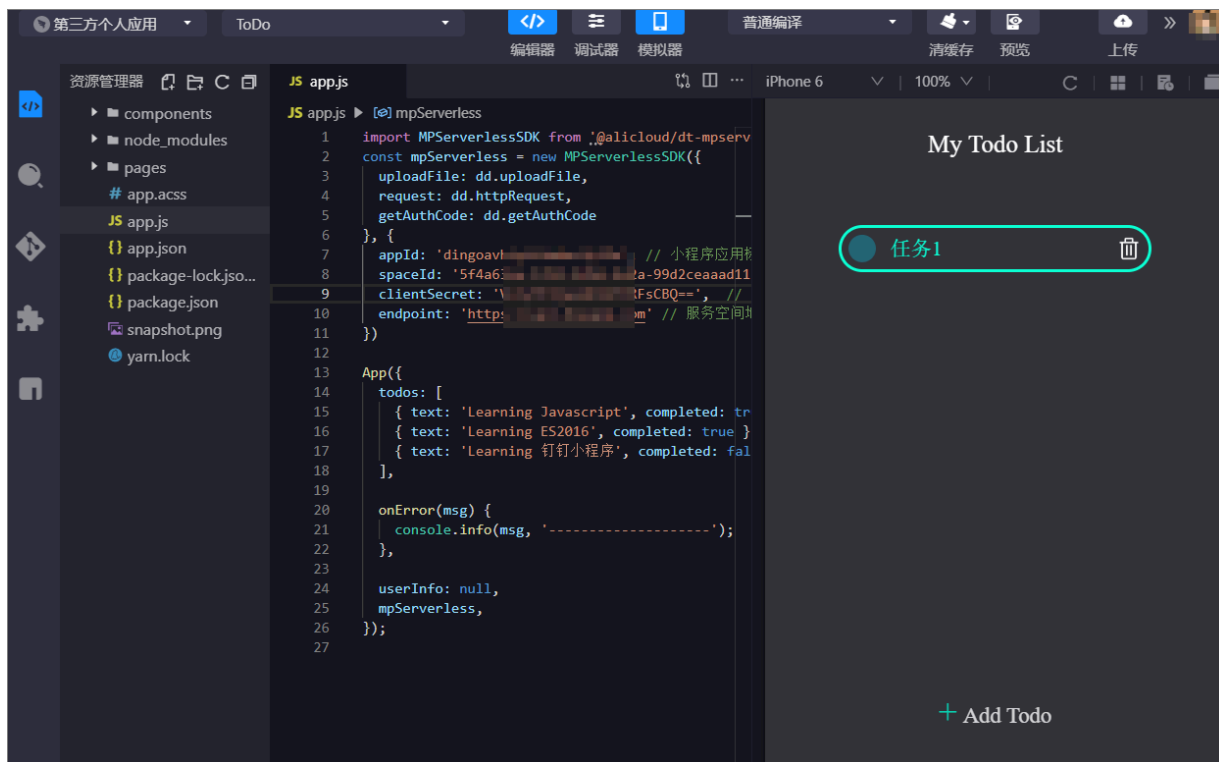
5. 在项目文件列表中找到`app.js`文件，配置以下信息，然后保存。

- appId是小程序的ID。您可以在钉钉开放平台的小程序详情页面查看App ID。
- spaceId、clientSecret和endpoint在小程序Serverless控制台创建服务空间后可以获得。详情参见[步骤一，创建小程云服务空间](#)。



步骤五：调试小程序

打开小程序模拟器，单击+AddTodo添加待办事项。



2.4 开发微信小程序

本教程介绍如何使用阿里云小程序云Serverless服务开发一个简单的待办事项（To Do）微信小程序。

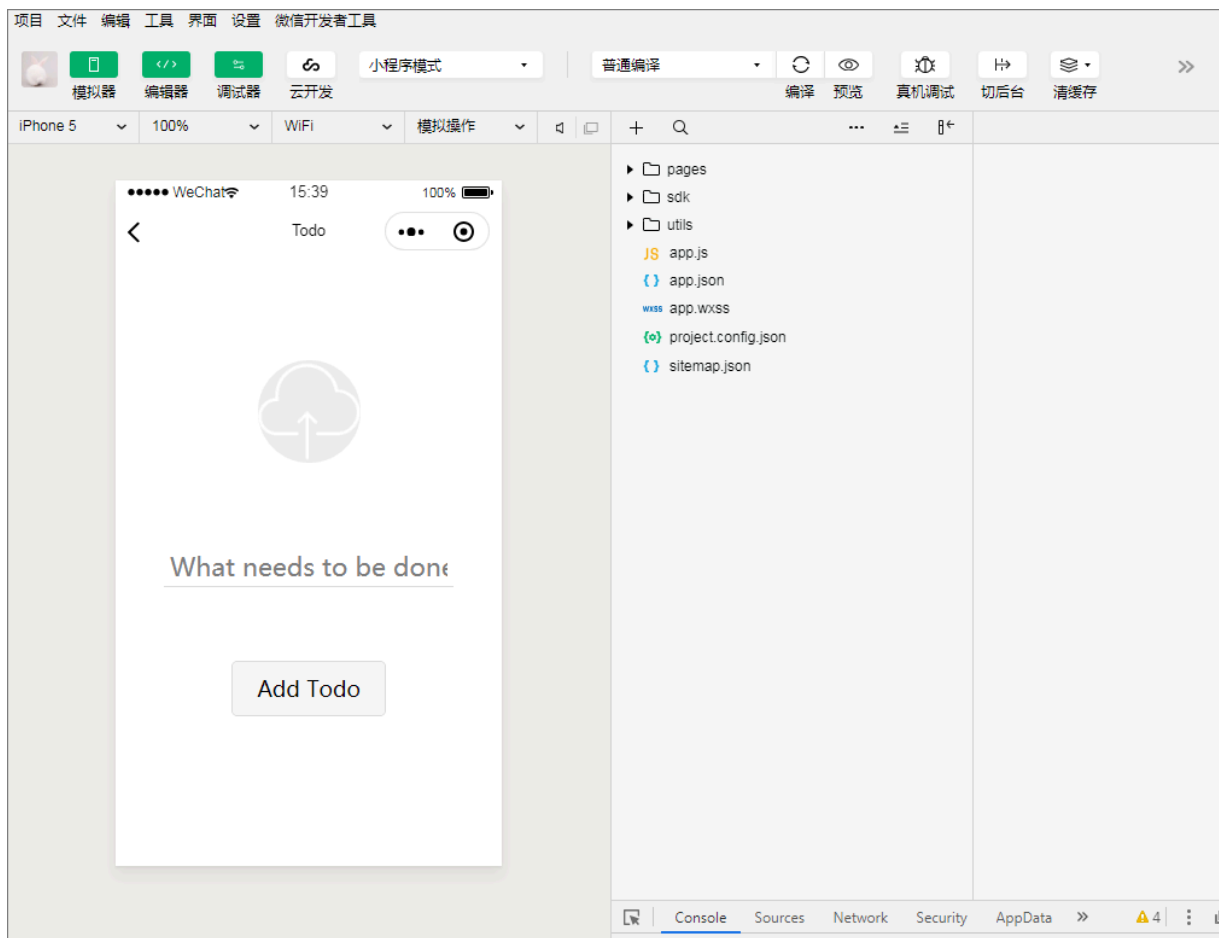
教程介绍

本教程以一个可添加、删除待办事项的小程序为例，了解学习如何在微信IDE中使用小程序云Serverless SDK调用云数据库API，开发一个微信小程序。本教程使用的小程序示例代码的业务逻辑已经编写好，您只需要根据本教程的引导开通Serverless服务，配置好小程序后，就可以运行该小程序了。



说明:

小程序Serverless SDK 2.0版本支持微信端，在引入SDK依赖后，需要调用`user.authorize`进行授权。详细信息请参见[在微信小程序中使用SDK](#)。



准备工作

在开始前，确保您已经完成以下准备工作：

- [注册微信小程序](#)
- 下载并安装微信开发者工具。

详细信息，请参见[下载说明](#)。



注意：

确保您下载的是最新版微信IDE。

- 开通阿里云小程序云服务

在首次使用小程序云服务前，您需要开通小程序云服务。详细信息，请参见[#unique_15](#)。

- [下载小程序示例项目](#)

步骤一：开通小程序云服务空间

小程序开发中创建的云资源会和服务空间进行关联。每个服务空间都有一个全局唯一的space ID。在使用云资源时，都通过这个space ID进行关联。

完成以下操作，创建小程序云服务空间并配置数据库权限：

1. 使用阿里云账号登录[小程序云控制台](#)。
2. 在左侧导航栏，单击小程序Serverless > 服务空间管理。
3. 单击创建服务空间，输入服务空间名称，然后单击确定。
4. 创建成功后，单击详情查看space ID、secret和endpoint信息。

服务空间名称

ToDo

SpaceID

5f4a6[REDACTED]aaad11 

API Endpoint

https://api.bspapp.com 

Secret

VLtyH[REDACTED]CBQ== 

文件上传 Endpoint

bsppub.oss-cn-shanghai.aliyuncs.com 

5. 单击小程序Serverless > 云数据库，然后在页面右上角选择已创建的服务空间。
6. 单击  修改数据库权限，将write权限修改为true。



说明：

为了保证数据安全，默认数据库访问受限。您可以根据需要设置数据库权限，更多信息，请参见[数据库权限管理](#)。

设置数据库权限

```
1 {  
2   "db": {  
3     "*": {  
4       ".read": true,  
5       ".write": true  
6     }  
7   }  
8 }
```

步骤二：添加微信小程序

在开发前，您需要将微信小程序信息添加到阿里云小程序Serverless控制台。

完成以下操作，添加微信小程序信息：

1. 登录[小程序云控制台](#)。
2. 在左侧导航栏，单击小程序Serverless > 设置。
3. 单击微信页签，然后单击添加密钥。

4. 在添加微信密钥页面，输入以下信息，然后单击确定。

- **AppID**：输入微信小程序的App ID。

登录[微信公众平台](#)，在开发设置页面查看App ID。



- **App Secret**：输入微信小程序的密钥。

添加微信密钥

* AppID

请输入小程序的 AppID

0/128

* App Secret

请输入小程序的 App Secret

步骤三：开发微信小程序

完成以下操作，开发微信小程序：



说明：

在微信小程序中通过小程序Serverless进行开发前，您需要引入小程序Serverless SDK依赖，然后便可以调用云函数、数据库和存储服务了。本示例中小程序Serverless SDK的JS已经添加到了示例文件中。在微信IDE中使用小程序Serverless SDK时，需要先引入SDK依赖，然后进行授权。更多详细信息，请参见[在微信小程序中使用SDK](#)。

1. 完成以下操作，配置小程序服务器域名：

- 登录[微信小程序公众号平台](#)。
- 在左侧导航栏，单击开发，然后单击开发设置。
- 在服务器域名区域，单击修改。
- 单击request合法域名文本框后的加号，输入微信小程序的域名、小程序API Endpoint和文件上传Endpoint。



您可以在[小程序Serverless控制台](#)的服务空间管理页面，单击服务空间详情查看endpoint信息。



- 单击[这里](#)下载To Do小程序示例项目文件，然后解压下载的文件。
- 在微信小程序IDE菜单栏中单击项目 > 导入项目导入步骤一中解压的小程序示例文件。



注意：

您需要替换成自己的微信小程序App ID。

4. 在微信IDE右上角，单击详情，勾选增强编译。

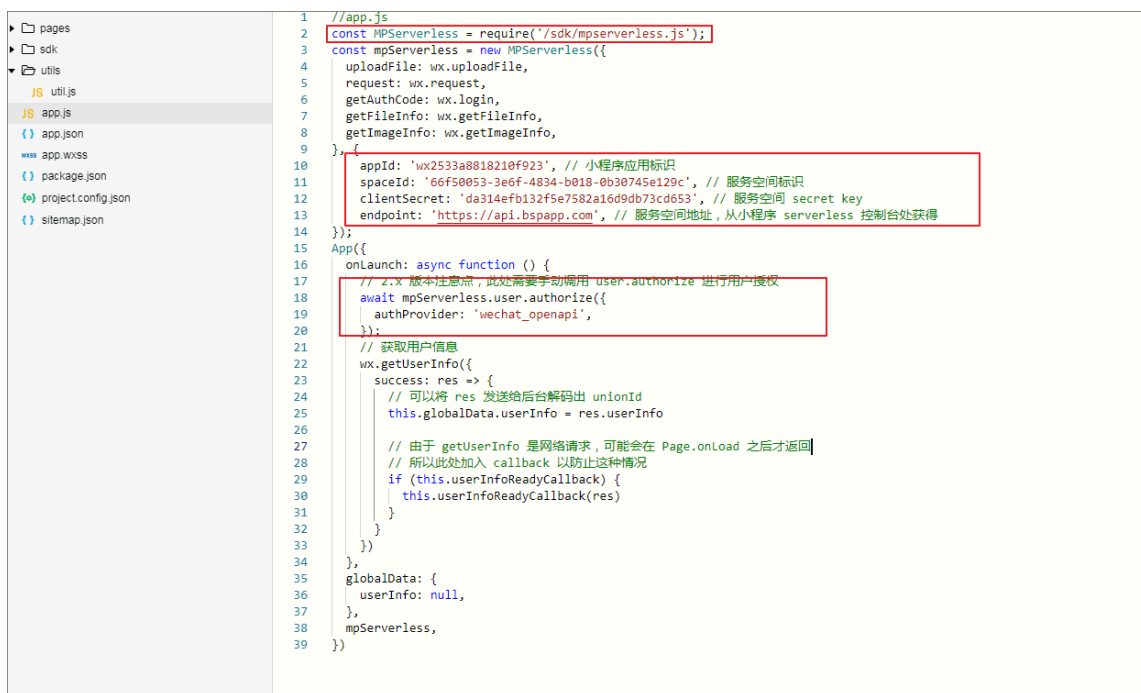


5. 单击app.js文件，配置以下信息，然后保存。

- appId是微信小程序的ID。您可以在[微信公众平台](#)的开发设置页面查看App ID。

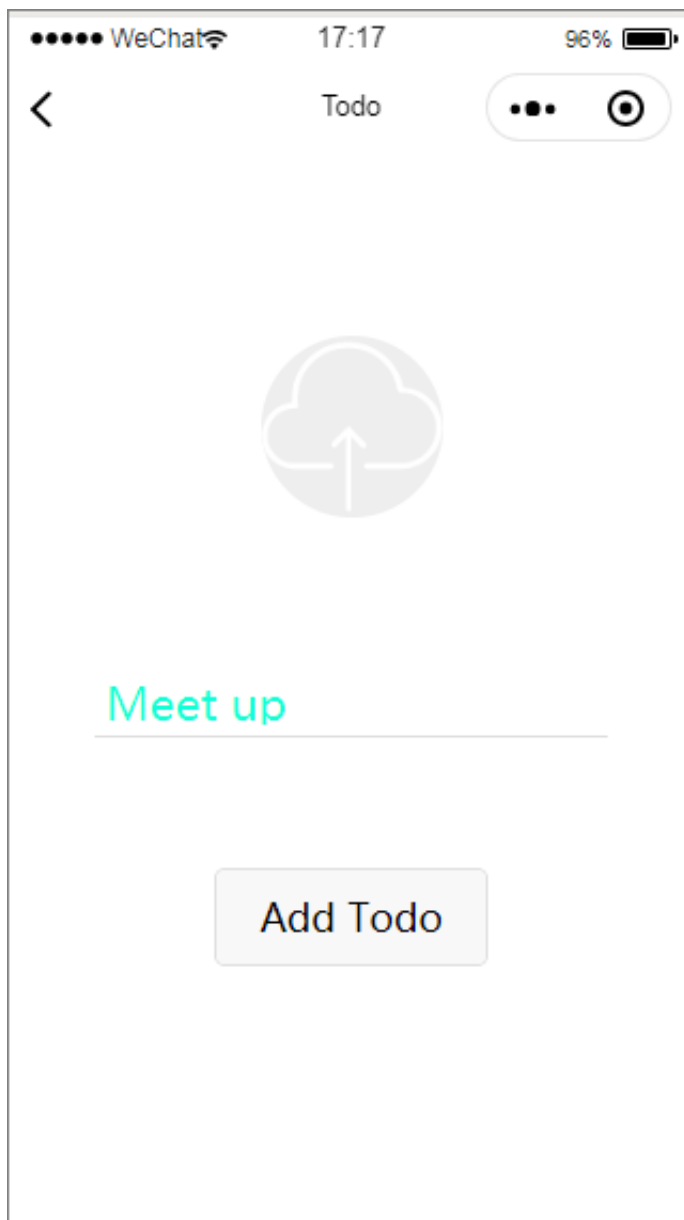


- spaceId、clientSecret和endpoint在小程序Serverless控制台创建服务空间后可以获得。详情参见[创建服务空间](#)。



步骤四：调试小程序

打开小程序模拟器，单击Add添加待办事项。



3 服务空间管理

3.1 创建服务空间

在使用小程序Serverless进行开发前，您需要先创建一个服务空间。小程序开发中创建的云资源会和服务空间进行关联。每个服务空间之间相互独立。

背景信息

所有的资源都是通过服务空间来进行分配和管理的。每个服务空间都有一个全局唯一的space ID。在使用云资源时，都通过这个space ID进行关联。

操作步骤

1. 登录[小程序云控制台](#)。
2. 在左侧导航栏，选择小程序Serverless > 服务空间管理。
3. 单击创建服务空间。
4. 输入服务空间名称和描述，然后单击确定。

服务空间名称可包含3~15个字符，不能以数字和特殊字符开头。

预期结果

服务空间创建成功后，在服务空间管理页面，单击详情获取space ID、endpoint和secret等信息。

服务空间名称	
ToDo	
SpaceID	
5f4a6	aaad11
API Endpoint	
https://api.bspapp.com	
Secret	
VLtyH	CBQ==
文件上传 Endpoint	
bsppub.oss-cn-shanghai.aliyuncs.com	

3.2 删除服务空间

当您不需要一个服务空间时，可以将其删除。服务空间关联的资源（云数据库、云存储等）也会随之删除。

操作步骤

1. 登录[小程序云控制台](#)。
2. 在左侧导航栏，选择小程序Serverless > 服务空间管理。
3. 找到要删除的服务空间，然后单击删除。
4. 在弹出的对话框，单击确定。

4 云函数

4.1 概述

云函数（FaaS）是一段运行在云端的、轻量的、无关联的、并且可重用的代码。无需管理服务器，只需编写和上传代码，即可获得对应的数据结果。使用云函数可以使企业和开发者不需要担心服务器或底层运维设施，可以更专注代码和业务本身，也可以使代码进一步解耦，增加其重用性。

在客户端只需依赖mpserverless-sdk，在应用的上下文中进行简单的鉴权配置，即可调用云函数。

云函数的入参只有一个ctx对象，该对象可获取调用的云函数时传入的参数，也可获取小程序serverless服务空间的云数据库，云存储等服务。

例如，我们定义一个云函数，名为hello。代码如下所示：

```
//云函数定义在index.js中，此文件为云函数的入口文件
module.exports = ctx => {
  const args = ctx.args;
  return {
    "text": args.hello,
  };
};
```

4.2 使用云函数

云函数支持使用Node.js进行开发。您可将代码提交到云端运行，在客户端使用小程序Serverless提供的API进行调用。您还可以在云函数中直接通过API调用数据存储和文件存储的服务资源。



说明：

您可以通过ctx.mpserverless.function.invoke调用其他云函数，详细说明请参见[invoke](#)。

步骤一：定义云函数

关联上对应的后端云服务后，您可以在服务端目录中编写云函数的代码。

1. 登录[小程序云](#)控制台，创建云函数。

详细操作，请参见[新建云函数](#)。



注意：

云函数的名称必须和要上传的Node.js代码包名称一致。



2. 在小程序的`server/functions`目录下创建云函数目录。

其中`index.js`是云函数`getImageList`的入口文件。

```
├── server/
│   └── functions
│       └── getImageList
│           └── index.js
```

3. 编写云函数代码。

以下代码示例展示了如何从数据库`images`里面查出特定用户上传的图片记录。

```
module.exports = async (ctx) => {
  const images = await ctx.mpserverless.db.collection('images').find(
    { owner: ctx.args.username });
  return { images };
};
```



说明:

云函数的返回值类型仅支持JSON格式。

4. 打包云函数文件。

确保云函数文件名称和控制台上创建的云函数名称一致且为ZIP格式。

步骤二：部署云函数

编写云函数代码后，您需要将云函数代码上传至控制台发布代码。

1. 在左侧导航栏，选择小程序Serverless > 云函数。
2. 单击已创建的函数名称链接。
3. 在发布管理页签，单击上传js包或更新js包，然后选择要上传的代码包。



说明:

上传的代码包必须满足以下要求：

- 代码包的名称必须和在控制台上创建的函数名称一致。
- 代码包必须是.zip文件。
- 上传的代码包必须包含index.js文件。

4. 上传成功后，单击代码部署。



步骤三：调用云函数

云函数代码发布后，在客户端您可以使用`mpserverless.function.invoke`方法调用上述定义的云函数。

以下代码示例展示了如何获取用户上传的图片记录，然后更新页面数据。

```
// 引入MPServerless模块
const MPServerless = require('@alicloud/mpserverless-sdk');
// 初始化MPServerless对象
const mpServerless = new MPServerless({
  uploadFile: my.uploadFile,
  request: my.request,
  getAuthCode: my.getAuthCode,
}, {
  appId: '1234456789', // 小程序应用标识
  spaceId: 'db4dd657-7041-470a-90xxxxx', // 服务空间标识
  clientSecret: '6c3c86xxxx6', // 服务空间 secret key
  endpoint: 'https://api.bspapp.com', // 服务空间地址，从小程序 serverless 控制台处获得
});
// 调用云函数getImageList
mpServerless.function.invoke('getImageList', {
  username: 'Vincent',
}).then((res) => {
  if (res.success && res.result) {
    this.setData({ imageUrl: res.result.imageUrl });
  }
});
```

```
}).catch(console.error);
```

4.3 传递参数

云函数是一个后端服务，有时需要获得从客户端传来的参数。通过云函数内置的`ctx.args`方法，您可以获得从客户端传来的参数。

客户端代码

以下是客户端代码示例：

```
await mpserverless.function.invoke('hello', { hello: 'world' });
```

云函数代码

以下是云函数代码示例：

```
module.exports = async ctx => {  
  ctx.logger.info('hello %s', ctx.args.hello);  
};
```

4.4 打印日志

为了便于开发调试、问题排查和日常监控，云函数提供日志服务。您可以在云函数中使用`ctx.logger`方法打印不同类型的日志信息，然后在云函数控制台中查看执行日志。

目前支持以下类型日志：

- **info**
- **warn**
- **error**
- **debug**

以`info`日志为例，打印客户端传来的参数内容如下：

```
module.exports = async (ctx) => {  
  ctx.logger.info('invoke args: %j', ctx.args);  
};
```

```
};
```

4.5 调用数据和文件存储服务

在云函数中，您可以直接调用同一环境（空间）的数据存储和文件存储服务。`ctx.mpserverless`在云函数封装了小程序的基础服务，API使用方式和客户端一致。

例如，要从数据存储获取数据，您可以直接调用`ctx.mpserverless.db`方法：

```
module.exports = async ctx => {
  const args = ctx.args;
  return await ctx.mpserverless.db.collection('user').find({ uid: args
    .uid });
};
```

4.6 调用HTTP服务

云函数提供了 `ctx.httpclient` HTTP 请求方法。无需额外依赖，您就可以请求任何 HTTP 和 HTTPS 协议的 Web 服务。

例如，请求一个网页：

```
module.exports = async ctx => {
  const res = await ctx.httpclient.request('https://www.alipay.com/x/
    notFound.htm');
  return {
    html: res.status === 200 ? res.data : '',
  };
};
```

其中，`ctx.httpclient`返回的是一个`urllib`实例。

4.7 开启云函数HTTP触发功能

小程序Serverless云函数支持由HTTP请求触发云函数调用的功能，本文介绍如何开启云函数HTTP触发功能。



注意：

当开启某一函数的 HTTP触发功能后，只要使用 HTTP请求就可以访问到函数，开发者需关注业务安全和资源安全。

- **业务安全：**开发者需在云函数代码中做好权限控制和安全防护，避免未授权访问触发敏感操作。
- **资源安全：**云函数开启了 HTTP 访问后，如果遇到大量恶意访问，消耗云函数资源，开发者可以将云函数的触发路径设置为空或关掉该Space的HTTP触发功能，停止HTTP触发支持。

开启方法

开启云函数HTTP触发功能，您需要开启Space的触发功能且配置云函数的HTTP触发路径。

- 开启Space的HTTP触发功能

调用接口 [UpdateHttpTriggerConfig](#) 接口来开启Space的HTTP触发功能，注意需要设置 `EnableService` 为 `true`。

- 开启云函数的HTTP触发功能

调用接口 [UpdateFunction](#) 接口来为云函数设置HTTP触发路径，只有设置了触发路径的云函数才能使用该功能。

限制说明

1. 请求路径必须以 `/http` 开头，不能以 `/` 结尾，同一个Space下面的云函数的请求路径不能重复，请求路径只支持 `(/)`、`(-)`、`(_)`、`(.)`、字母和数字组合，请求路径最长不超过128个字符。

2. 请求方法仅支持以下四种，其余方法会被忽略。

- GET
- POST
- PUT
- DELETE

3. 请求时Body的Content-Type仅支持以下三种，允许没有Body的请求，即没有Content-Type。其余Content-Type类型会被拒绝。

- `application/json`
- `application/x-www-form-urlencoded`
- `text/plain`

4. 通过HTTP触发的响应，都会添加固定的Header，即 `Content-Disposition: attachment`，此字段会使得返回结果在浏览器中以附件的方式下载。当前此字段无法覆盖，后续支持使用自定义域名之后，可以避免该问题。

5. 请求Body大小限制，不能超过4K。

6. 响应Body大小限制，不能超过4K。

云函数入参说明

在云函数中，通过 `ctx.args` 来获取请求时的内容。各参数的详细解释如下：

```
{
  path: 'HTTP请求路径，如 /hello',
```

```
httpMethod: 'HTTP请求方法, 如 GET',
headers: {HTTP请求头, 键值对形式},
queryStringParameters: {HTTP请求的Query, 键值对形式},
body: 'HTTP请求体',
isBase64Encoded: 'true or false, 表示body是否为Base64编码, 当前只会有false'
}
```

云函数返回值说明

云函数返回时, 可以返回`string`、`object`、`number`、`boolean`等类型的数据, 或者返回自定义响应, 随后小程序Serverless会将返回值转化为正常的HTTP响应。

- 返回`string`

当服务端返回的Content-Type是`text/plain`。

例如云函数返回:

```
'hello world!'
```

则最终的响应为:

```
HTTP/2 200
date: Mon, 23 Mar 2020 10:12:41 GMT
content-type: text/plain
content-length: 12
content-disposition: attachment
request-id: ac14000d1584958361033101870

hello world!
```

- 返回`object`、`number`或`boolean`

当服务端返回的Content-Type是`application/json`。

例如云函数返回:

```
{
  "keyString": "value1",
  "keyNumber": 1234
}
```

则最终的响应为:

```
HTTP/1.1 200 OK
Date: Mon, 23 Mar 2020 10:18:48 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 39
request-id: ac14000d1584958728277106170
content-disposition: attachment
```

```
{"keyString": "value1", "keyNumber": 1234}
```

- 返回自定义响应

服务端返回的Content-Type, HttpStatusCode, Headers都由云函数来指定。

要求云函数返回一个object，同时要设定mpserverlessComposedResponse为true。

云函数的返回格式如下：

```
{
  "mpserverlessComposedResponse": true,
  "isBase64Encoded": true|false,
  "statusCode": httpStatusCode,
  "headers": { "headerName": "headerValue", ... },
  "body": "..."
```

例如云函数返回：

```
{
  mpserverlessComposedResponse: true,
  isBase64Encoded : false,
  statusCode: 200,
  headers: {
    'content-type': 'text/html'
  },
  body: '<h1>Hello World</h1>'
}
```

则最终的响应为：

```
HTTP/2 200
date: Mon, 23 Mar 2020 10:40:09 GMT
content-type: text/html; charset=utf-8
content-length: 20
request-id: ac14000d1584960009970137070
content-disposition: attachment

<h1>Hello World</h1>
```

4.8 控制台操作指南

4.8.1 新建云函数

您可以在控制台上创建云函数。发布代码后，您可以在控制台上查看执行记录、部署历史以及资源使用情况。

前提条件

[创建服务空间](#)

操作步骤

1. 登录[小程序云控制台](#)。
2. 在左侧导航栏，选择小程序Serverless > 云函数。
3. 如果您尚未开通函数计算，单击前往开通，然后选择服务协议并单击确认开通。
4. 返回云函数页面，在页面右上角选择一个服务空间。
5. 单击新建云函数。
6. 输入函数名称。

函数名称长度在1-30个字符间，只能包含字母、数字、下划线和中划线，不能以数字、中划线开头。



注意：

云函数的名称必须和要上传的Node.js代码包名称一致。

7. 选择运行环境。目前支持Node.js8。
8. 输入描述。
9. 单击确定。

后续步骤

部署代码

4.8.2 部署代码

您可以在控制台上将代码提交到云端运行。

前提条件

新建云函数

操作步骤

1. 登录[小程序云控制台](#)。
2. 在左侧导航栏，选择小程序Serverless > 云函数。
3. 单击已创建的函数名称链接。
4. 在发布管理页签，单击上传js包或更新js包，然后选择要上传的代码包。



说明：

上传的代码包必须满足以下要求：

- 代码包的名称必须和在控制台上创建的函数名称一致。
- 代码包必须是.zip文件。
- 上传的代码包必须包含index.js文件。

5. 上传成功后，单击代码部署。



6. 部署成功后，单击代码执行，然后输入要执行的命令，单击确定。

执行完成后，您可以直接在控制台上查看执行结果。

4.8.3 查看日志

为了便于开发调试、问题排查和日常监控，云函数提供日志服务。您可以直接在控制台上查看日志。

背景信息

目前支持以下类型日志：

- info
- warn
- error
- debug

操作步骤

1. 登录[小程序云控制台](#)。
2. 在左侧导航栏，选择小程序Serverless > 云函数。
3. 单击已创建的函数名称链接。
4. 单击日志。
5. 根据执行时间、状态和Request ID等信息过滤您要查看的日志。

5 数据存储

5.1 数据结构设计

数据存储服务是基于MongoDB托管在云端的数据库，数据以JSON格式存储。作为开发者，您可以在客户端内直接操作数据，也可以在云函数中读写数据。

数据库结构

小程序Serverless服务使用的是MongoDB，以JSON格式存储数据。数据库中的每条记录都是一个JSON格式的对象，一个数据库可以包含多个集合（相当于关系型数据库中的表），每个集合可看做一个JSON数组。

小程序Serverless MongoDB数据库和关系型数据库MySQL的对比，如下表所示。

分布式文件存储数据库 (MongoDB)	关系型数据库 (MySQL)
数据库 (database)	数据库 (database)
表 (collection)	表 (table)
记录 (document)	行 (row)
域 (field)	列 (column)
索引 (index)	索引 (index)
自动将_id字段作为主键	主键 (primary key)

数据结构设计策略

MongoDB是一个基于分布式文件存储的NoSQL数据库，旨在为WEB应用提供可扩展的高性能数据存储解决方案。由于数据的灵活性以及通用性，在设计数据结构时也需要根据数据之间的关系来决定如何存储。

以电商平台为例。通常电商平台的核心数据包含产品、用户、购物车和订单。订单是由用户购买产品产生的，订单就可以认为是产品和用户之间的关联关系产生的。在订单产生之前，产品和用户之间的关联关系是通过购物车来维持的。

针对上述案例，可以设计一个名称为products的产品表存储以下信息：

- 产品基础信息：包含展示信息、商品规格等。
- 属性信息：属性都归属于一个产品，属于N对1的关系。因此作为子文档存储在products集合中较合适。

- 库存及价格信息：库存不仅仅跟产品关联，更直接对应产品属性。因此也应该放在products集合中。

最终的数据库设计如下：

```
{
  "id": 5573,
  "name": "Egg T恤",
  "desc": {
    "short": "限量版 Egg T恤，穿上它你离极客也不远了",
    "long": "这是一段非常长的描述",
    "category": {
      "_id": "48bf43a..29e90bc",
      "name": "上衣",
    },
  },
  "attributes": [
    {
      "id": 1151,
      "name": "尺寸",
      "values": [
        {
          "id": 3871,
          "value": "S",
        },
        {
          "id": 3874,
          "value": "M",
        },
        {
          "id": 3875,
          "value": "L",
        },
      ],
    },
    {
      "id": 1151,
      "name": "性别",
      "values": [
        {
          "id": 3872,
          "value": "男",
        },
        {
          "id": 3873,
          "value": "女",
        },
      ],
    },
  ],
  "sku": [
    {
      "id": 1153,
      "stock": 30,
      "attributeIds": [ 3871, 3872 ],
      "attributes": [
        {
          "key": "尺寸",
          "value": "S",
        },
        {
          "key": "性别",

```

```
      "value": "女",
    },
  ],
},
]
```

5.2 使用数据存储服务

您可以通过`mpserverless.db`对象调用数据存储相关的API，也可以通过云函数直接调用同一个云服务的数据存储服务。

SDK调用示例

以下代码展示了如何通过小程序SDK获得当前云服务中`todos`集合内所有任务的列表：

```
mpserverless.db.collection('todos').find().then((res) => {
  this.setData({ todos: res.result });
}).catch(console.error);
```

云函数调用示例

以下代码展示了如何通过云函数获得当前云服务中`todos`集合内所有任务的列表：

```
module.exports = async (ctx) => {
  const res = await ctx.mpserverless.db.collection('todos').find();
  return res.result;
};
```

更多信息

[安装客户端SDK](#)

[数据存储API参考](#)

5.3 数据权限管理

小程序Serverless提供了一套简明易懂的JSON语法用来控制用户对资源的访问，类似于身份验证体系里的IAM或者网络权限中的ACL。您可以在控制台上通过修改默认权限规则来管理数据表的权限。

规则语法

一组权限规则通常由数据库集合和权限操作组成：

```
{
  "db": {
    "${collection}": {
      ".read": true,
      ".write": false,
      "*": false
    }
  }
}
```

```
    },
    "*": {
      ".read": true,
      ".write": false,
      "*": false
    }
  }
}
```

- JSON的第一层级key是数据库集合名称，代表进行权限管理的范围。



说明:

"*" 代表没有匹配到集合名的默认集合。

- JSON 第二层级key代表操作类型:

- .read : 读操作
- .write : 写操作
- * : 所有操作

- 操作取值支持: true, false。

匹配规则

当第一层级匹配到集合后:

- 匹配当前集合中对应的操作类型。若当前集合中没有匹配到操作类型，会查找当前集合中 "*" 操作类型。
- 没有匹配后继续查找第一层级 "*" 集合中对应的操作类型。若还是没有匹配的操作，继续查找第一层级中 "*" 集合中 "*" 操作类型。
- 若上述所有规则都未匹配，拒绝请求。

当第一层级没有匹配到指定的集合:

- 查找第一层级中 "*" 集合中对应的操作类型，没有匹配到操作类型，继续查找第一层级中 "*" 集合中的 "*" 操作类型。
- 若上述所有规则都未匹配，拒绝请求。

修改权限规则

下表列举了常用的数据权限管理规则配置及其权限说明。关于如何修改数据库权限，请参见[设置数据库权限](#)。

规则设置	说明
<code>".read": true</code> <code>".write": true</code>	所有用户都对该数据表拥有全部读写权限。  说明: 该权限规则会放开数据库操作权限，请谨慎设置。

规则示例

权限规则如下：

```
{
  "db": {
    "posts": {
      ".write": true,
      ".read": true
    },
    "*": {
      "*": false
    }
  }
}
```

向posts集合中插入如下一条数据：

```
{
  "title": "文章标题",
  "content": "文章内容"
}
```

插入数据后的数据库内容如下：

```
{
  "title": "文章标题",
  "content": "文章内容"
}
```

由于posts集合权限规则中定义".read"操作类型为true，所以可以读取所有数据。

默认规则

对于新创建的服务，小程序Serverless会自动提供一个默认规则。默认没有任何读写权限，请先修改权限规则。

操作详情，请参见[设置数据表权限](#)。

```
{
  "db": {
    "*": {
      "*": false
    }
  }
}
```

```
}
```

5.4 控制台操作指南

5.4.1 创建数据表

小程序Serverless服务使用的是分布式文件存储数据库MongoDB，以JSON格式存储数据。数据库中的每条记录都是一个JSON格式的对象，一个数据库可以包含多个集合（相当于关系型数据库中的表）。您可以在控制台上创建数据表。

前提条件

创建服务空间

操作步骤

1. 登录[小程序云控制台](#)。
2. 在左侧导航栏，选择小程序Serverless > 云数据库。
3. 在页面右上角，选择要关联的环境。
4. 单击添加图标 ，输入数据表名称，最后单击确定。

相关文档

数据结构设计

数据存储服务是基于MongoDB托管在云端的数据库，数据以JSON格式存储。作为开发者，您可以在客户端内直接操作数据，也可以在云函数中读写数据。

5.4.2 添加记录

数据库中的每条记录都是一个JSON格式的对象，您可以直接在控制台上添加一条记录。

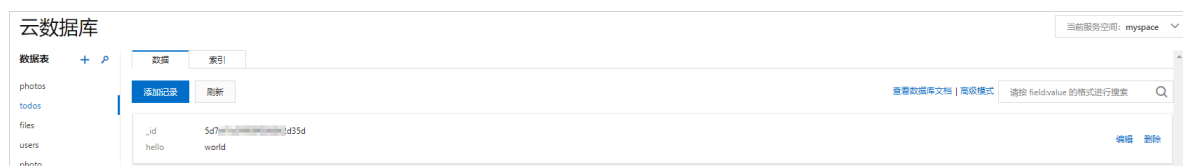
前提条件

创建数据表

操作步骤

1. 在数据库页面，单击目标数据表，然后单击添加记录。
2. 在添加记录页面，以JSON格式输入记录内容，然后单击确定。

每条记录都有一个唯一的ID。添加成功后您可以删除或修改该条记录。



相关文档

insertOne

在集合中添加一条记录。

[insertMany](#)

在集合中添加一批记录。

5.4.3 设置数据表权限

小程序Serverless会为每个新建的数据表自动提供一个默认权限规则。默认新建数据表无任何操作权限，您可以通过添加权限规则管理数据表的权限。

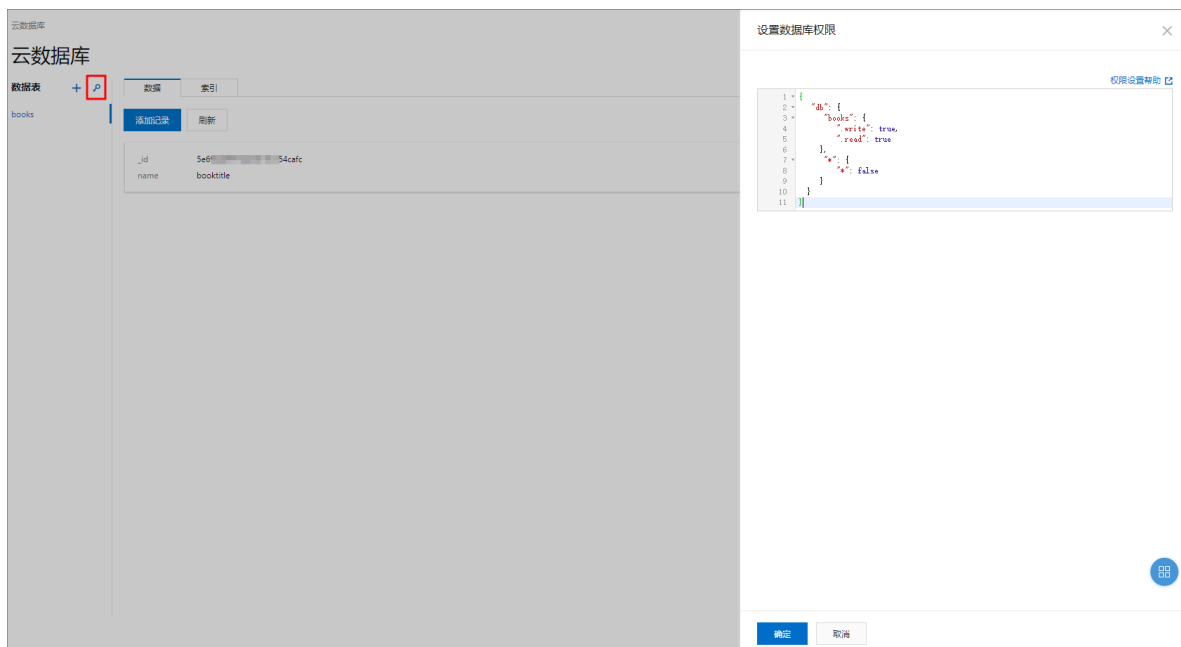
前提条件

确保您已经了解小程序云Serverless服务的数据库权限设置规则。

小程序Serverless提供了一套简明易懂的JSON语法用来控制用户对资源的访问，类似于身份验证体系里的IAM或者网络安全中的ACL。您可以通过修改权限规则来控制数据表权限。更多详细信息，请参见[数据权限管理](#)。

操作步骤

1. 在[云数据库](#)页面，单击权限图标 .
2. 在设置数据库权限页面，设置权限规则。然后直接修改安全规则。
关于权限规则的语法要求，请参见[规则语法](#)。
3. 修改完成后，单击确定。



5.4.4 使用高级模式

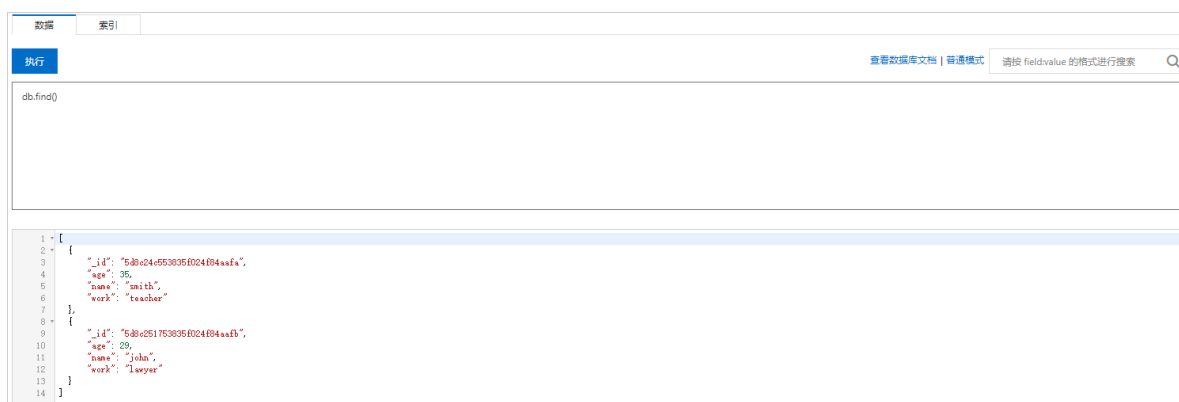
在控制台的云数据库管理页面中，可以切换到高级模式直接编写数据库语句的执行脚本，对数据库进行增删查改操作。

前提条件

创建数据集合

操作步骤

1. 在数据库页面，单击目标数据表，然后选择页面右上角的高级模式。
2. 在输入框中输入执行脚本。
3. 单击执行按钮，输出执行结果。



说明:

高级模式支持的操作同客户端，请参见[数据库API](#)。

5.4.5 设置数据库索引

使用数据库时，对成为查询条件的字段设置索引可以有效提高查询效率，更快的获取信息。缺省情况下会对_id_字段默认创建一个非唯一索引。

前提条件

- [创建数据集合](#)
- [添加记录](#)

操作步骤

1. 在数据库页面，单击目标数据表，然后单击索引页签进入索引设置页面。
2. 单击添加索引进入索引设置页面。

3. 填写索引名称，设置索引属性、索引字段。

- 索引属性：可以设置为唯一索引或非唯一索引。

将字段设置为唯一索引后，可以防止不同记录的被索引键上存储相同值。



注意：

如果一个记录在一个唯一索引中没有索引字段的值，则该索引将为该记录存储一个空值。由于唯一约束限制，只会允许一个记录缺少索引字段。对多于一个以上的记录没有索引字段的值或缺少索引字段，索引构建将失败，提示重复键错误。

- 索引字段：支持单键索引和复合索引设置，字段按升序或降序排列。

单键索引是最常见的索引形式，针对一个指定字段建立索引。对于单字段索引，升序、降序的查询效果一样。

复合索引是单键索引的升级版，针对多个字段联合创建索引，先按照第一个字段排列，第一个字段相同的记录按第二个字段排列，依次类推。

4. 单击保存按钮，保存索引设置。

数据表			
+ /			
test			
todos			
user			
添加索引 刷新			
索引名称	索引属性	索引字段	操作
_id	非唯一	_id (升序)	删除
name	唯一	name (升序)	删除

5.5 数据管理最佳实践

5.5.1 管理数据集

您可以通过小程序Serverless提供的数据存储服务API对集合（相当于MySQL中的表）进行数据读取、写入、更新、删除等操作。

读取数据

以下代码示例展示了适用于不同场景的数据读取方法：

- 从集合中读取所有数据。

例如，查询所有图片：

```
mpserverless.db.collection('images').find();
```

- 根据查询条件从集合中读取指定数据。

例如，查询特定用户添加的图片：

```
mpserverless.db.collection('images').find({ 'userId': userId });
```

- 读取数据并对获取的数据进行排序。

例如，将获取的图片按照添加时间进行排序：

```
mpserverless.db.collection('images').find({ 'userId': userId }, { 'uploadTime': -1 });
```

- 读取数据并对获取的数据进行操作。

例如，读取用户的图片列表，并显示出来。其中，`_getImages(user)` 是封装好的方法，用来获取用户对应的图片列表，并更新小程序中的数据，此时页面内容也会刷新。在`onShow()`方法中，当用户信息获取成功，便去获取用户的图片列表并显示到页面上。

```
// client/index/index.js

// 显示指定当前用户的图片列表
onShow() {
  mpserverless.user.getInfo().then((user) => {
    this._getImages(user);
  }).catch(console.log);
},

// 获取特定用户的图片列表
_getImages(user) {
  mpserverless.db.collection('images')
    .find({ userId: user.userId }, { uploadTime: -1 })
    .then(({ result: images }) => {
      images.map((item) => {
        item.uploadTime = new Date(item.uploadTime).toString();
        return item;
      });
      this.setData({ images });
    })
    .catch(console.log);
},
```

写入数据

您可以通过以下两种方式写入数据：

- 通过insertOne()方法逐条向集合写入数据。

例如，使用insertOne()方法为用户添加一个文档：

```
mpserverless.db.collection('images').insertOne({
  text: inputText,
  url: imageUrl,
});
```

- 通过insertMany()方法批量写入数据。

例如，使用insertMany()方法为用户添加一个文档：

```
// client/add-image/add-image.js

// 将新的任务添加到当前用户的图片列表中
add() {
  mpserverless.user.getInfo().then((user) => {
    mpserverless.db.collection('images').insertOne({
      text: this.data.inputValue,
      url: this.data.imageUrl ? this.data.imageUrl : false,
      userId: user.userId,
      uploadTime: new Date(),
    }).then(() => {
      my.navigateBack();
    }).catch(console.log);
  }).catch(console.log);
},
```

更新数据

您可以通过以下两种方式更新数据：

- 通过updateOne()方法更新符合条件的第一条数据。

例如，使用updateOne()方法更新一个文档：

```
mpserverless.db.collection('images').updateOne(
  { _id: imageId },
  {
    $set: {
      text: result.inputValue,
    }
  }
);
```

其中，\$set是更新记录的处理方式。

- 通过updateMany()方法更新符合条件的所有数据。

以一个图片管理的小程序为例，用户触发重命名图片名称操作时，显示对话框让用户输入新的图片名称，执行更新文档中的名称字段：

```
// client/index/index.js

// 变更图片名称的事件处理
```

```

rename(e) {
  const dataset = e.target.dataset;

  my.prompt({
    title: '修改名称',
    message: '请输入新的图片名称:',
    placeholder: '',
    okButtonText: '确定',
    cancelButtonText: '取消',
    success: (result) => {
      if (result.ok) {
        mpserverless.db.collection('images').updateOne(
          { _id: dataset.itemId },
          {
            $set: {
              text: result.inputValue,
            }
          }
        ).then(() => {
          this._getImages(this.data.user);
        }).catch(console.log);
      }
    }
  });
},
});
},

```

删除数据

您可以通过以下两种方式删除数据：

- 通过`deleteOne()`方法删除第一个符合条件数据。

例如，使用`deleteOne()`方法删除一个文档：

```
mpserverless.db.collection('image').deleteOne({ _id: imageId });
```

- 通过`deleteMany()`方法删除所有符合条件数据。

以一个图片管理的小程序为例，当用户单击示例小程序图片删除按钮时，唤起删除对话框，确认后执行删除和刷新图片列表：

```

// client/index/index.js

// 删除图片的事件处理
delete(e) {
  const dataset = e.target.dataset;

  // 确认和删除图片
  my.confirm({
    title: '删除图片',
    content: '是否确认删除该图片?',
    confirmButtonText: '删除',
    cancelButtonText: '取消',
    success: (result) => {
      if (result.confirm) {
        mpserverless.db.collection('images').deleteOne({
          '_id': dataset.itemId,
          'userId': this.data.user.userId,
        }).then(() => {
          // 刷新任务列表

```

```
        this._getImages(this.data.user);
    }).catch(console.log);
  },
},
});
},
```

5.5.2 原子操作

目前数据存储服务使用MongoDB 3.4，不具备处理事务的能力。您可以参考本文的示例对数据进行原子并发操作。

原子插入

要保证同一条信息不可插入两次，在集合上新增一个字段并且设置unique index，如下所示：

```
// 创建唯一索引
mpserverless.db.collection('order').createIndex(
  { 'orderId': 1 },
  { 'unique': true }
);

// 插入时验证
function placeOrder(items) {
  const orderId = uuid();
  mpserverless.db.collection('order').insertOne({
    orderId,
    items,
  }).then(res => {
    console.log('order placed successful');
  }).catch(err => {
    console.error('insert failed');
  });
}
```

原子更新

原子更新可以通过判断当前的最后更新时间来匹配是否可以更新。例如，在发送一个礼物时，A和B同时领取，在不支持原子更新的情况下，可能导致两个用户都抢到的情况。

```
mpserverless.db.collection('gifts').createIndex('inventory.itemId');

function giveOutGift(owner, item) {
  // 发礼物
  return mpserverless.db.collection('inventory').updateOne({
    user: owner,
  }, {
    $addToSet: { // 保证唯一性, subdocument 不能建立唯一索引
      inventory: {
        item,
        itemId: item.id,
        status: 'unclaimed',
      },
    },
  }, { upsert: true });
}

function freezeGift(item) {
  return mpserverless.db.collection('gifts').updateOne({
```

```

    inventory: {
      $elemMatch: { // 找到数组里对应的礼物
        itemId: item.id,
        status: 'unclaimed',
      },
    },
  }, {
    $set: {
      'inventory.$.status': { status: 'transferring' }, // 设置成转移中
    },
  }).then(res => {
    if (res.affectedDocs !== 1) {
      return Promise.reject();
    }
  });
}

function claimGift(to, item) {
  return mpserverless.db.collection('gifts').updateOne({
    user: to,
  }, {
    $addToSet: {
      inventory: {
        item,
        itemId: item.id,
        status: 'owned'
      },
    },
  }, {
    upsert: true,
  }).then(res => {
    if (res.affectedDocs !== 1) {
      return Promise.reject();
    }
  });
}

function claimUserGift(from, to, item) {
  return freezeGift().then(claimGift).then(() => mpserverless.db.
collection('gifts').updateOne({
  user: from,
  inventory: {
    $elemMatch: {
      itemId: item.id,
      status: 'transferring',
    },
  },
}, {
  $set: {
    'inventory.$.status': 'claimed',
  },
}));
}

giveOutGift('alice', {
  name: 'umbrella',
  id: '5b9642e109d54b4c12d68c7e',
}).then(res => claimUserGift('alice', 'bob', {
  name: 'umbrella',
  id: '5b9642e109d54b4c12d68c7e',
}));

```


6 文件存储

6.1 使用文件存储服务

文件存储服务支持文本、图片和其他由用户生成的内容存储到云端。您可以通过`mpserverless.file`对象调用文件存储相关的API，也可以通过云函数直接调用同一个云服务的文件存储服务。

SDK调用示例

以下代码展示了如何将小程序客户端中选中的图片上传，获得图片URL。

```
// 选择并上传图片，获得图片 URL
attach() {
  my.chooseImage({
    chooseImage: 1,
    success: res => {
      const path = res.apFilePaths[0];
      const options = {
        filePath: path,
        headers: {
          contentDisposition: 'attachment',
        },
      },
    },
  });

  mpserverless.file.uploadFile(options).then((image) => {
    console.log(image);
    this.setData({
      imageUrl: image.fileUrl,
    });
  }).catch(console.log);
},
});
},
```

云函数调用示例

以下代码展示了如何通过云函数删除客户端中上传的图片。

```
module.exports = async (ctx) => {
  // ctx.args 是从客户端传过来的参数
  const res = await ctx.mpserverless.file.deleteFile({ fileUrl: ctx.args.fileUrl });
  return res;
};
```

更多信息

[安装客户端SDK](#)

[文件存储API参考](#)

6.2 文件管理最佳实践

`mpserverless.file`对象提供`uploadFile`和`deleteFile`方法管理文件。上传的文件将通过CDN进行网络加速。

文件格式

文件存储服务目前支持上传图片、音频和视频文件。文件格式要求如下表所示。

文件类型	文件格式
图片	jpg/jpeg、png、gif、bmp、wbmp、webp、svg
音频	mp3
视频	mp4、ogg、webm

上传文件

以下代码展示了如何通过`uploadFile`方法以指定路径的方式上传一张公开的图片。

```
const options = {
  // env: public, // 不指定存储方式 env 的情况下，默认为公开 (public)
  filePath: path,
  headers: {
    contentDisposition: 'attachment',
  },
};

mpserverless.file.uploadFile(options);
```

删除文件

以下代码展示了如何通过`deleteFile`方法根据文件的URL删除一个文件。

```
const options = {
  filePath: path,
  headers: {
    contentDisposition: 'attachment',
  },
};

mpserverless.file.uploadFile(options).then((image) => {
  mpserverless.file.deleteFile({ filePath: image.filePath });
});
```

示例教程

以一个图片管理小程序为例，当用户新增一个图片条目的时候，需要上传一个图片作为附件。那么您需要让用户在添加图片时，在客户端选择一个图片上传到Serverless文件存储服务，并将得到的图片路径记录在数据存储中。

首先，增加attach()方法，作为上传图片的事件处理，用于上传图片和获取图片路径。

```
// client/add-image/add-image.js

// 选择并上传图片，获得图片 URL
attach() {
  my.chooseImage({
    chooseImage: 1,
    success: res => {
      const path = res.apFilePaths[0];
      const options = {
        filePath: path,
        headers: {
          contentDisposition: 'attachment',
        },
      };

      mpserverless.file.uploadFile(options).then((image) => {
        console.log(image);
        this.setData({
          imageUrl: image.fileUrl,
        });
      }).catch(console.log);
    },
  });
},
```

然后，提交新增图片条目时，将图片URL作为数据一并存储到数据存储中。

```
// client/add-image/add-image.js

// 将新的图片内容添加到当前用户的图片列表中
add() {

  // 如果图片名称没有填，或者没有上传图片，则进行提示
  if (this.data.inputValue == '' || !this.data.imageUrl) {
    my.alert({
      title: '添加失败',
      content: '请填写图片名称和上传图片。',
      buttonText: '我知道了',
    });
  }

  // 正常情况，写入数据存储
} else {
  mpserverless.user.getInfo().then((user) => {
    mpserverless.db.collection('images').insertOne({
      text: this.data.inputValue,
      url: this.data.imageUrl ? this.data.imageUrl : false,
      userId: user.userId,
      uploadTime: new Date(),
    }).then(() => {
      my.navigateBack();
    }).catch(console.log);
  }).catch(console.log);
}
```

```
},
```

6.3 文件权限管理

小程序Serverless提供了一套简明易懂的JSON语法用来控制用户对资源的访问，类似于身份验证体系里的IAM或者网络权限中的ACL。您可以在控制台上修改文件权限。

规则语法

一组权限规则由多个子规则组成，每个子规则包含以下信息：

- **范围**：用于定义规则生效范围。

在数据存储的安全规则中，每一个规则的范围是一个文件（路径）。您可以通过文件路径指定一个特定的文件，也可以通过JavaScript的正则表达式指代一批文件或使用星号（*）指代所有文件。以下是三种范围书写方式示例：

- **文件路径**：`xxx/yyy/zzz.jpg`。范围是这个特定文件。
- **正则表达式**：`/.*\.\.png/`。范围是所有后缀为`.png`的文件。
- **所有集合**：`*` 指代所有集合。
- **操作**：用于定义生效范围内的特定操作。所有人都具有文件的读权限。

文件存储提供以下操作定义规则：

- `.write`：写操作
- `*`：所有操作
- **策略**：用于确定生效范围内的特定操作是否允许。默认没有标记为允许的操作，都是不允许的。

策略是操作是否被允许的标记，可以是一个布尔值，或者是一个表达式。当策略冲突时，以最先出现的规则策略为准。

表 6-1: 表达式通用对象

顶级对象	字段名称	描述
<code>request.auth</code>	<code>userId</code>	请求用户ID。

顶级对象	字段名称	描述
resource.auth	userId	资源所属用户ID。

userId 指的是通过mpserverless.user.getInfo()获取的userId字段。

表 6-2: 表达式运算符

操作符	描述	匹配类型
==	等于	string number
!=	不等于	string number
>	大于	number
>=	大于等于	number
<	小于	number
<=	小于等于	number
!	非	string number

默认规则

对于新创建的服务，小程序Serverless会自动提供一个默认规则。默认规则规定所有文件都只有资源所有者可以进行写操作，所有人可读。

```
{
  "file": {
    "*": {
      ".write": "request.auth.userId == resource.auth.userId"
    }
  }
}
```

规则示例

例如，您可以创建如下安全规则，满足不同的图片访问需求：

- 对于.png格式的图片只有图片所有者可以写，所有人可读。
- 某个特殊路径的图片只有所有者可以读写。
- 其他文件所有人都可读写。



注意:

JSON结构的安全规则配置暂不支持//注释。为例方便阅读，在文档中使用//进行说明，在实际使用时请去掉//注释。

```
{
  "file": {
    "/*\\.png/": {
      ".write": "request.auth.userId == resource.auth.userId",
      "*": true
    },
    "xxx/yyy/zzz.jpg": {
      "*": "request.auth.userId == resource.auth.userId"
    },
    "*": {
      "*": true
    }
  }
}
```

其中:

- 第一个规则是针对.png格式的图片文件。/*\\.png/是一个正则，匹配所.png图片作为规则范围。.read是具体操作描述，*是全部其他操作的描述。规则内容为.png格式的图片只有资源所有者可以进行文件写操作，所有人可以进行读取操作。
- 第二个规则是针对xxx/yyy/zzz.jpg这个特定路径的图片，只有资源所有者可以进行文件读写操作。
- 第三个规则是针对除了前两个规则匹配到的文件之外的所有文件，所有人都可以进行读写操作。

修改文件权限

详细说明，请参见[设置文件权限](#)。

6.4 控制台操作指南

6.4.1 上传文件

您可以直接在控制台上上传小程序所需的文件。

前提条件

上传文件前，确保您以满足以下条件：

- [创建服务空间](#)。

- 支持上传图片、音频和视频文件，格式要求如下：
 - 图片：.jpg .jpeg .png .gif .bmp .wbmp .webp .svg .image
 - 音频：.audio .au .mp2 .mp3
 - 视频：.3gpp .mp4 .mpeg .flv .avi .wmv .mpg .mpga
- 要上传的图片名称必须满足以下条件：； 区分大小写；；
 - 使用 UTF-8 编码。
 - 图片名称区分大小写。
 - 长度必须在 1-1023 字节之间。
 - 不能以 / 或者 \ 字符开头。

操作步骤

1. 登录[小程序云控制台](#)。
2. 在左侧导航栏，选择小程序Serverless > 云存储。
3. 单击上传文件。
4. 单击直接上传或将文件拖拽至上传区域。

文件上传过程中，请勿刷新或关闭页面，否则上传任务会被中断且列表会被清空。

预期结果

上传成功后，您可以单击详情查看图片下载地址。



相关文档

[uploadFile](#)

上传文件。

6.4.2 删除图片

您可以直接在控制台上删除不需要的图片。

操作步骤

1. 登录[小程序云控制台](#)。

2. 在左侧导航栏，选择小程序Serverless > 云存储。
3. 找到目标图片，然后单击删除。
4. 在弹出的对话框中，单击确定。

相关文档

[deleteFile](#)

删除之前上传的文件。

6.4.3 设置文件权限

小程序Serverless会为每个新建的文件自动提供一个默认权限规则。默认规则规定所有文件都只有资源所有者可以进行写操作，所有人可读。您可以在控制台上修改文件权限。

前提条件

确保您已经了解小程序云Serverless服务的文件权限设置规则。

小程序Serverless提供了一套简明易懂的JSON语法用来控制用户对资源的访问，类似于身份验证体系里的IAM或者网络安全中的ACL。您可以通过修改权限规则来管理文件权限。更多详细信息，请参见[文件权限管理](#)。

操作步骤

1. 登录[小程序云控制台](#)。
2. 在左侧导航栏，选择小程序Serverless > 云存储。
3. 单击权限页签。
4. 修改权限规则，然后单击保存。

7 扩展能力

7.1 云调用概述

云调用是基于小程序Serverless的云函数来使用支付宝小程序开放接口的能力，可以方便的让开发者在小程序中直接调用支付宝的后端开放接口，不需要关注服务端的相关配置，极大的减小了接入的流程，进一步降低了支付宝小程序的开发门槛。

目前云调用涵盖了支付宝开放的基础能力、营销能力、会员能力、支付能力、安全能力，以及阿里云开放的短信能力。各项能力涉及的API列表如下所示。



说明:

目前，仅支持在支付宝小程序中使用如下扩展能力。

基础能力

API	描述
base.oauth.getToken	获取授权访问令牌。
base.oauth.refreshToken	刷新授权访问令牌。
base.qrcode.create	创建小程序二维码。

营销能力

API	描述
marketing.templateMessge.send	发送模板消息。

会员能力

API	描述
member.identification.init	身份认证初始化。
member.identification.certify	生成认证链接。
member.identification.query	身份认证记录查询。

支付能力

API	描述
payment.common.create	创建交易。

API	描述
payment.common.query	查询交易。
payment.common.refund	交易退款。
payment.common.close	关闭交易。
payment.common.cancel	撤销交易。
payment.huabei.create	创建花呗分期交易。
payment.faceToFace.pay	当面付交易付款。

安全能力

API	描述
security.textRisk.detect	检测内容风险。

阿里云能力

API	描述
aliyun.shortMessage.send	发送短信。
aliyun.shortMessage.batchSend	批量发送。
aliyun.shortMessage.query	查询发送记录。
aliyun.shortMessage.preciseQuery	精准查询。

7.2 使用云调用

要使用小程序云提供的扩展能力，您需要先在控制台开通云调用功能，然后在小程序中安装云调用SDK并进行初始化。



说明：

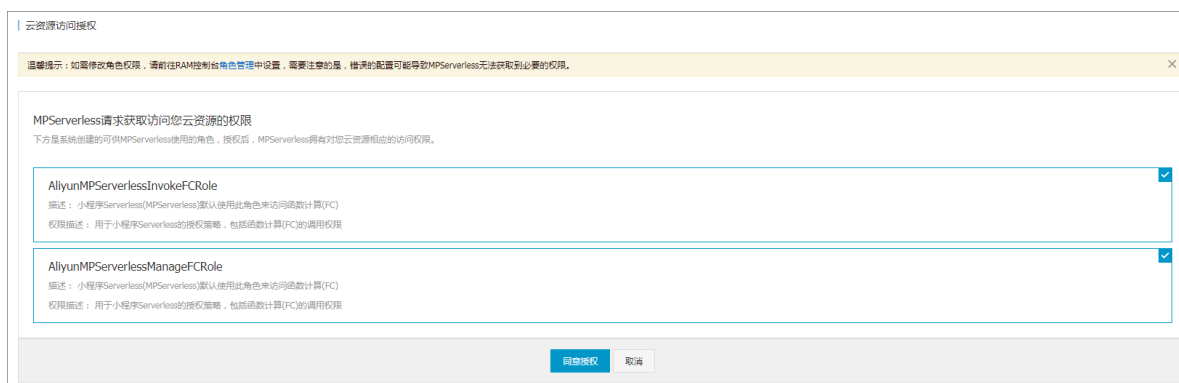
目前，仅支持在支付宝小程序中使用云调用功能。

步骤一：开通云调用功能

完成以下操作，在小程序云控制台开通云调用功能：

1. 登录[小程序云控制台](#)。
2. 在左侧导航栏，选择小程序Serverless > 扩展能力。
3. 在扩展能力页面，单击立即开通。

4. 在弹出的对话框，单击前往RAM进行授权，然后单击同意授权允许小程序访问函数计算服务。



5. 返回扩展能力页面，查看开通状态。



步骤二：安装并初始化SDK

完成以下操作，安装并初始化云调用SDK：

1. 在小程序项目的根目录执行以下命令安装SDK。

```
npm install alipay-cloud-sdk --save
```

2. 执行以下代码初始化小程序云客户端SDK。

```
// 依赖 SDK
import MPServerless from '@alicloud/mpserverless-sdk';

// 初始化依赖的对象
const serverless = new MPServerless({
  uploadFile: my.uploadFile,
  request: my.request,
  getAuthCode: my.getAuthCode
}, {
```

```
appId: '', // 小程序应用标识
spaceId: '', // 服务空间标识
clientSecret: '', // 服务空间 secret key
endpoint: '' // 服务空间地址, 从小程序Serverless控制台处获得
});
```

其中:

- appId是小程序的ID。您可以在支付宝小程序控制台获得。
- spaceId、clientSecret和endpoint在小程序Serverless控制台创建服务空间后可以获得。详情参见[创建服务空间](#)。

3. 执行以下代码初始化alipay-cloud-sdk。

```
import cloud from 'alipay-cloud-sdk';

// 调用 .init 进行初始化, 之后直接使用 cloud 对象即可
// serverless 对象是 @alicloud/mpserverless-sdk 的实例
cloud.init(serverless, {
  // 配置消息回调相关的配置
  notify: {
    url: '' // 消息回调的地址, exp: https://callback.com
  }
});
```

7.3 公共数据结构

返回数据结构

下表列举了alipay-cloud-sdk的公共返回数据。

字段名	数据类型	说明
requestId	String	请求ID。
success	Boolean	请求是否成功。
result	Any	请求响应数据, 结构与具体API有关。

result公共数据如下表所示。

字段名	数据类型	说明
code	String	响应码。
msg	String	响应主消息。
subCode	String	响应子码, 一般是子系统的响应码。
subMsg	String	响应子消息, 一般是子系统的返回消息

公共错误码

下表列举了公共错误码。

错误码	说明
SYS_INIT	SDK未初始化。 使用SDK之前需要调用 <code>cloud.init</code> 方法初始化。详情请参见 使用云调用 。
FEATURE_NOT_SUPPORT	API暂未支持。
INVALID_JSON_OBJECT	JSON解析失败。
NOT_FOUND_PARAM	缺少必要参数，请检查必填参数。
INVALID_PARAM	参数不合法，请检查参数类型等是否正确。

8 统计分析

8.1 集成统计分析服务

8.1.1 支付宝小程序集成

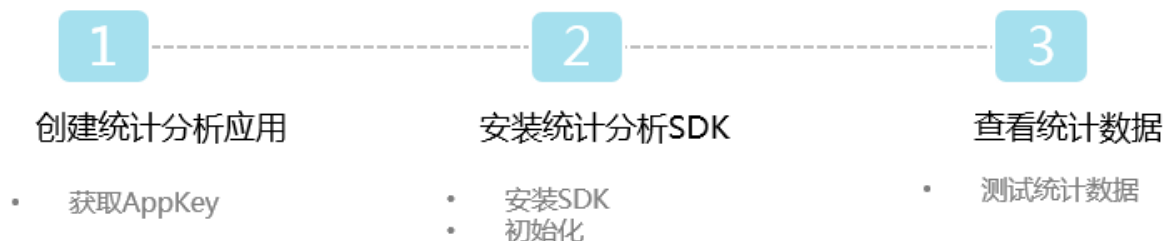
要为支付宝小程序接入统计分析服务，您要先在控制台创建应用，然后在小程序项目中安装统计分析服务SDK。

前提条件

- 确保支付宝小程序版本不低于10.1.52，低版本将获取不到统计信息。
- [创建服务空间](#)。

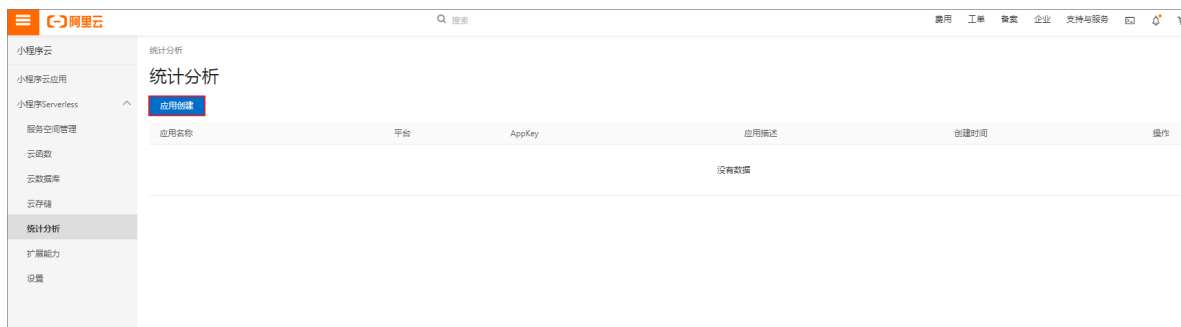
背景信息

要接入支付宝小程序进行用户使用数据统计分析，您需要完成以下操作：



步骤一：创建统计分析应用

1. 登录[小程序云管理](#)控制台。
2. 选择服务空间。
3. 在左侧导航栏，单击统计分析，然后单击创建应用。



4. 根据提示配置应用信息，然后单击确定。

应用创建

* 应用名称

支付宝ToDoApp

* 平台选择

支付宝

* 一级分类

实用工具

* 二级分类

生活工具

描述

请输入应用的相关描述

确定

取消

5. 记录生成的AppKey。

统计分析						
应用创建		当前服务空间: doc				
应用名称	平台	AppKey	应用描述	创建时间	操作	
支付宝ToDoApp	支付宝	5d887ce95*****0002a8		2019-09-24 05:06:02	查看数据 编辑 删除	

步骤二：安装SDK

1. 在小程序项目的根目录执行以下命令安装统计分析SDK。

```
npm i @alicloud/mpserverless-analyse-alipaySdk -- save
```

2. 执行以下命令安装小程序Serverless 客户端SDK。

```
npm install @alicloud/mpserverless-sdk@2.2.0 --save
```



说明：

客户端SDK版本不能低于2.2.0。

3. 在支付宝小程序项目的`app.js`文件中添加如下代码：

在使用该代码时请注意：



说明：

- 示例中的3个方法是SDK最基本的统计方法，请务必都引入。
- 请在小程序最初初始化时调用`MPAnalyse.init()`方法，请勿出现任何异步方法阻塞`init`方法的调用。
- 请在`init`方法中填入在统计分析页面中生成的AppKey，否则无法在统计分析页面中看到正确数据。
- 慎重调用`clearStorage()`以及`clearStorageSync()`接口。SDK会将用户相关操作数据缓存在客户端数据存储模块，在特定时间启动发送策略。若调用该接口可导致数据统计不准确的问题。
- 若用户在使用小程序过程强制关闭支付宝有可能会造成统计数据丢失的情况。

```
import MPAnalyse from '@alicloud/mpserverless-analyse-alipaySdk';
import MPServerless from '@alicloud/mpserverless-sdk'; ///SDK版本必须
为2.2.0及以上

const serverless = new MPServerless({
  uploadFile: my.uploadFile,
  request: my.request,
  getAuthCode: my.getAuthCode
}, {
  appId: '1234456789', // 小程序应用标识
  spaceId: 'db4dd657-7041-470a-90xxxxx', // 服务空间标识
  clientSecret: '6c3c86xxxx6', // 服务空间 secret key
  endpoint: 'https://endpoint' // 服务空间地址，从小程序Serverless控制台
处获得
});

App({
  onLaunch(options) {
    MPAnalyse.init('YOUR_APP_KEY', my, { // 务必填入已注册的统计分析
      appKey, 不然将无法统计
    });
  }
});
```

```
mpServerless: serverless,
MPServerless后的serverless
platformParam: options
暂不传入
});
},
onShow() {
    MPAnalyse.resume();
},
onHide() {
    MPAnalyse.pause();
}
});
```

其中：

- `appId`：小程序的ID。您可以在[支付宝小程序开放平台](#)获取小程序的App ID。
- `spaceId``clientSecret`和`endpoint`：在小程序Serverless控制台创建服务空间后可以获得。详情参见[创建服务空间](#)。
- `MPAnalyse.init`：YOUR_APP_KEY是[步骤一：创建统计分析应用](#)中创建应用生成的AppKey。

统计分析						当前服务空间: doc
应用创建						
应用名称	平台	AppKey	应用描述	创建时间	操作	
支付宝ToDApp	支付宝	5d887ce95*****9002a8		2019-09-24 05:06:02	查看数据 编辑 删除	

- platformParam: options: 如果需要进行渠道分析功能，必须引入该参数。



注意:

渠道分析将会统计用户首次进入小程序页面时带的渠道参数，请保证首次进入的url参数中带有`sdk_channel`字段。例如首次进入的URL为：`pages/index/index?sdk_channel=test`，`test`将会作为此次渠道的值。

4. 打开调试工具中的console选项，若看到有 [SDK] -- 集成SDK成功提示。即表示SDK集成成功。



说明:

该方案仅提供验证是否集成成功的方法，由于在开发环境下可能会存在数据不完整的情况，请您正式发布小程序后在验证数据正确性。

步骤三：查看统计数据

1. 登录小程序云控制台。
2. 在左侧导航栏，单击统计分析。

3. 单击目标统计分析应用的查看数据链接进入统计分析页面。



8.1.2 微信小程序集成

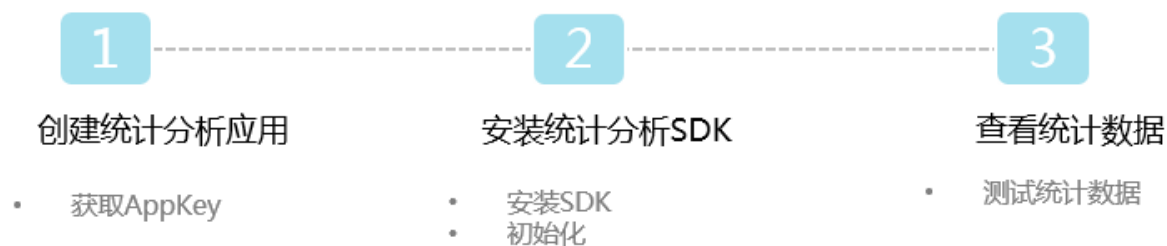
要为微信小程序接入统计分析服务，您要先在控制台创建应用，然后在小程序项目中安装统计分析服务SDK。

前提条件

[创建服务空间](#)。

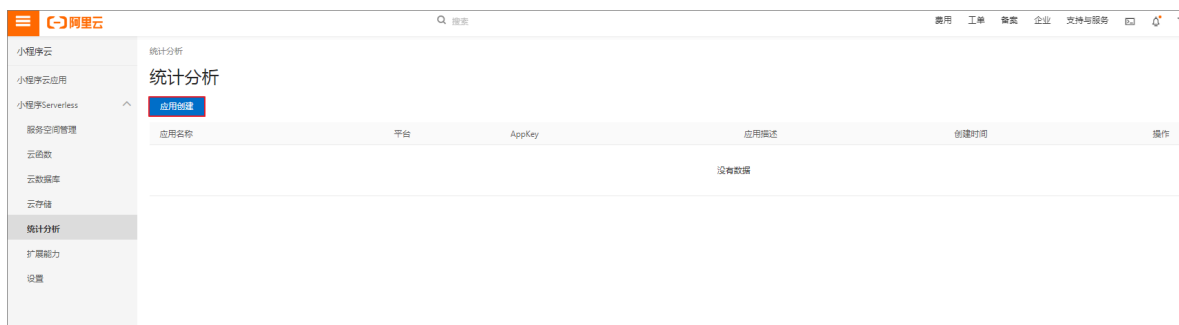
背景信息

要接入微信小程序进行用户使用数据统计分析，您需要完成以下操作：



步骤一：创建统计分析应用

1. 登录[小程序云管理控制台](#)。
2. 在左侧导航栏，单击统计分析，然后单击创建应用。



3. 根据提示配置应用信息，然后单击确定。

应用创建

* 应用名称

微信ToDoApp

* 平台选择

微信

* 一级分类

实用工具

* 二级分类

生活工具

描述

请输入应用的相关描述

确定

取消

4. 记录生成的AppKey。

统计分析						
应用创建						
应用名称	平台	AppKey	应用描述	创建时间	操作	
微信ToDoApp	微信	5a8922-36a		2019-09-25 00:55:51	查看数据 编辑 删除	

步骤二：安装SDK

1. 在小程序项目的根目录执行以下命令安装统计分析SDK。

```
npm i @alicloud/mpserverless-analyse-wxSdk -- save
```

2. 单击[这里](#)下载小程序Serverless客户端SDK文件，将下载后的`mpserverless.js`文件添加到小程序项目目录下。

说明：

客户端SDK版本不能低于2.2.0。

3. 在微信小程序项目的`app.js`文件中添加如下代码：

在使用该代码时请注意：



说明：

- 示例中的3个方法是SDK最基本的统计方法，请务必都引入。
- 请在小程序最初初始化时调用`MPAnalyse.init()`方法，请勿出现任何异步方法阻塞`init`方法的调用。
- 请在`init`方法中填入在统计分析页面中生成的AppKey，否则无法在统计分析页面中看到正确数据。
- 慎重调用`clearStorage()`以及`clearStorageSync()`接口。SDK会将用户相关操作数据缓存在客户端数据存储模块，在特定时间启动发送策略。若调用该接口可导致数据统计不准确的问题。
- 若用户在使用小程序过程强制关闭微信有可能会造成统计数据丢失的情况。

```
const MPAnalyse = require('@alicloud/mpserverless-analyse-wxSdk');
const MPServerless = require('/sdk/mpserverless.js'); //客户端SDK版本
必须为2.2.0及以上

const serverless = new MPServerless({
  uploadFile: wx.uploadFile,
  request: wx.request,
  getAuthCode: wx.login,
  getFileInfo: wx.getFileInfo,
  getImageInfo: wx.getImageInfo
}, {
  appId: '1234456789', // 小程序应用标识
  spaceId: 'db4dd657-7041-470a-90xxxxxx', // 服务空间标识
  clientSecret: '6c3c86xxxx6', // 服务空间 secret key
  endpoint: 'https://endpoint', // 服务空间地址，从小程序Serverless控制台
  处获得
});

App({
  onLaunch: function (options) {
```

```
MPAnalyse.init('YOUR_APP_KEY', { // 务必填入已注册的统计分
析appKey, 不然将无法统计
  mpServerless: serverless,      // 务必传入实例化
  MPServerless后的serverless
  platformParam: options         // 若不需要渠道分析功能, 可
  以暂不传入
});
},

onShow (options) {
  MPAnalyse.resume();           // 请务必引入
},

onHide () {
  MPAnalyse.pause();           // 请务必引入
}
});
```

其中:

- appId: 小程序的ID。您可以在[微信小程序开放平台](#)获取小程序的App ID。
- spaceIdclientSecret和endpoint: 在小程序Serverless控制台创建服务空间后可以获得。详情参见[创建服务空间](#)。
- MPAnalyse.init : YOUR_APP_KEY是步骤一中创建统计分析应用生成的AppKey。

统计分析					
应用创建					
当前服务空间: doc					
应用名称	平台	AppKey	应用描述	创建时间	操作
支付宝ToDoApp	支付宝	5c887ce95*****90002a8		2019-09-24 05:06:02	查看数据 编辑 删除

- platformParam: options: 如果需要渠道分析功能, 必须引入参数。



注意:

渠道分析将会统计用户首次进入小程序页面时带的渠道参数, 请保证首次进入的url参数中带有sdk_channel字段。例如首次进入的URL为: pages/index/index?sdk_channel=test, test将会作为此次渠道的值。

4. 打开调试工具中的console选项, 若看到有 [SDK] -- 集成SDK成功提示。即表示SDK集成成功。



说明:

该方案仅提供验证是否集成成功的方法, 由于在开发环境下可能会存在数据不完整的情况, 请您正式发布小程序后在验证数据正确性。

步骤三: 查看统计数据

1. 登录[小程序云](#)控制台。
2. 在左侧导航栏, 单击统计分析。

3. 单击目标统计分析应用的查看数据链接进入统计分析页面。



8.2 查看统计数据

接入统计分析服务后，您可以在小程序控制台中查看小程序运行的用户、渠道等数据，帮助您制定小程序运营策略。

前提条件

在查看统计数据前，确保您已经完成小程序的接入：

- [支付宝小程序集成](#)
- [微信小程序集成](#)

操作步骤

1. 登录[小程序云](#)控制台。
2. 在左侧导航栏，单击统计分析。
3. 单击目标统计分析应用的查看数据链接进入统计分析页面。



8.3 基本统计指标说明

指标	说明
新增用户	第一次启动小程序的用户（以设备为判断标准）。
启动次数	打开小程序视为启动，完全退出或退至后台即视为启动结束。
活跃用户	启动过小程序的用户（去重），启动过一次的用户即视为活跃用户，包括新用户与老用户。
累计用户	截止到当前时间，启动过小程序的所有独立用户（去重，以设备为判断标准）。
平均日使用时长	日使用时长的均值，即全部用户的日使用时长/总活跃用户。
平均日启动次数	用户一天内启动小程序的次数，即总启动次数/活跃用户。

指标	说明
活跃用户构成	每日活跃用户中新用户与老用户的分布。
新增用户次日留存率	该日的新增用户在次日仍启动小程序的用户占该日新增的比例。
平均单次使用时长	单次使用时长的均值，即全部用户的日使用时长/总启动次数。

8.4 概况分析

8.4.1 整体趋势

整体趋势页面通过展示累计用户、活跃用户等基本指标的趋势图和TOP渠道数据，帮助您了解小程序使用的整体趋势和规模。

基本指标说明

指标	说明
新增用户	第一次启动小程序的用户（以设备为判断标准）。
启动次数	打开小程序视为启动，完全退出或退至后台即视为启动结束。
活跃用户	启动过小程序的用户（去重），启动过一次的用户即视为活跃用户，包括新用户与老用户。
累计用户	截止到当前时间，启动过小程序的所有独立用户（去重，以设备为判断标准）。
平均日使用时长	日使用时长的均值，即全部用户的日使用时长/总活跃用户。
平均日启动次数	用户一天内启动小程序的次数，即总启动次数/活跃用户。
活跃用户构成	每日活跃用户中新用户与老用户的分布。
新增用户次日留存率	该日的新增用户在次日仍启动小程序的用户占该日新增的比例。
平均单次使用时长	单次使用时长的均值，即全部用户的日使用时长/总启动次数。

整体趋势概览

提供截止至昨日小程序整体情况的7个指标数据，并计算这些指标的同比变化。



说明：

同比变化指和上一个周期（上个7日内）的数据变化。



表 8-1: 整体趋势指标说明

指标	说明
新增用户（7日平均）	最近7日（不含今日）每日新增用户的平均值。
新用户次日留存率（7日平均）	最近7日次日留存率的平均值。
活跃用户（7日平均）	最近7日（不含今日）每日活跃用户的平均值。
使用时长（7日平均）	最近7日（不含今日）用户每日使用时长的平均值。
近7日总活跃用户数	最近7日（不含今日）活跃用户的总数（去重）。
累计用户数	截止到当前时间，启动过应用的所有独立用户（去重，以设备为判断标准）。
近30日总活跃用户数	单最近30日（不含今日）活跃用户的总数（去重）。

趋势变化与对比

提供新增用户、活跃用户、启动次数等9个常用指标的线图变化趋势，支持时间对比。默认只展示8个指标的趋势图，您可以单击编辑指标选择要查看的指标趋势。



整体趋势图支持添加时间对比。时间对比可以帮助您准确的获取对比时间内的各项指标的变化趋势，分析用户使用行为。

- 1. 单击趋势图上方的日历图标，选择趋势图的时间粒度。例如选择展示过去7天的数据趋势。
- 2. 选择要查看的指标。
- 3. 单击+选择时间然后选择对比时间。

说明:

对比时间的时间粒度和您步骤 1 中选择的时间粒度相同。

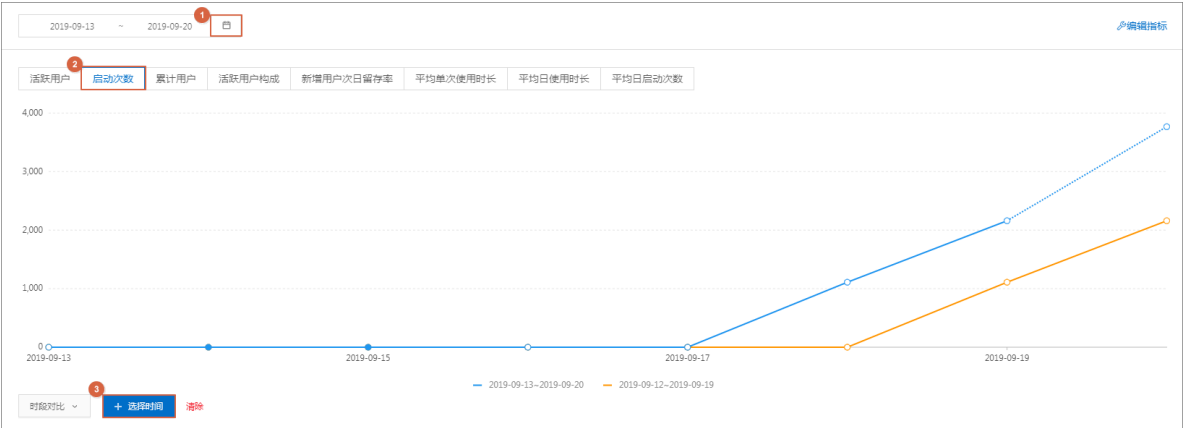
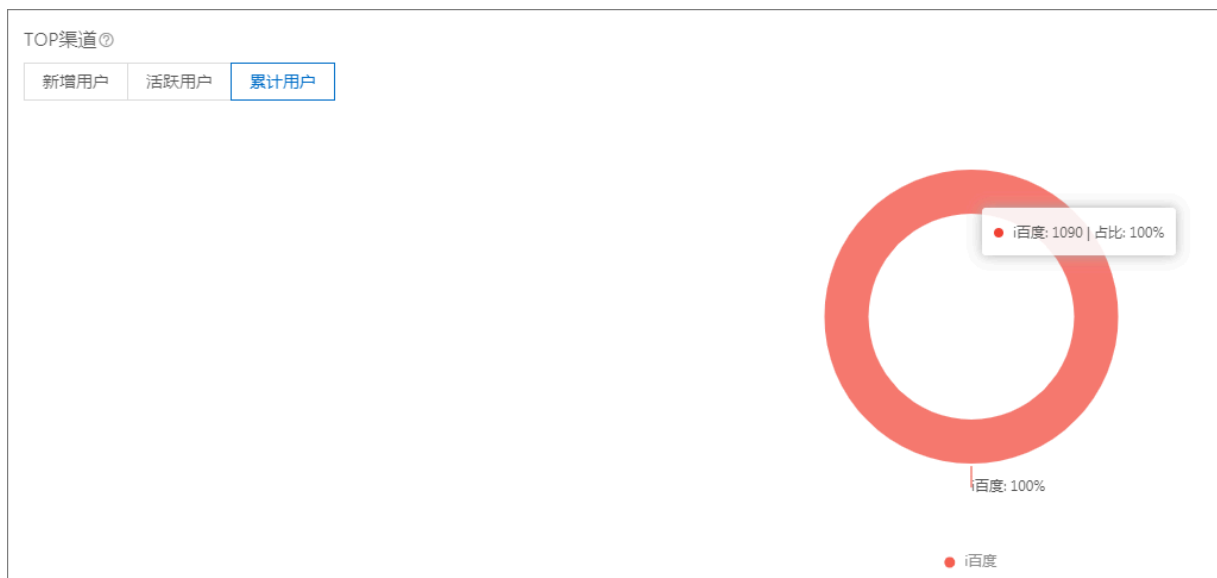


表 8-2: 对比趋势指标说明

指标	说明
新增用户	第一次启动小程序的用户（以设备为判断标准）。
启动次数	打开小程序视为启动，完全退出或退至后台即视为启动结束。
活跃用户	启动过小程序的用户（去重），启动过一次的用户即视为活跃用户，包括新用户与老用户。
累计用户	截止到当前时间，启动过小程序的所有独立用户（去重，以设备为判断标准）。
平均日使用时长	日使用时长的均值，即全部用户的日使用时长/总活跃用户。
平均日启动次数	用户一天内启动小程序的次数，即总启动次数/活跃用户。
活跃用户构成	每日活跃用户中新用户与老用户的分布。
新增用户次日留存率	该日的新增用户在次日仍启动小程序的用户占该日新增的比例。
平均单次使用时长	单次使用时长的均值，即全部用户的日使用时长/总启动次数。

TOP渠道

提供昨日新增、活跃、累计用户排名Top 10渠道和其它渠道的占比情况。



- 新增用户：按照今日新增排名的Top 10渠道的今日与昨日的新增用户数量的占比。
- 活跃用户：按照今日活跃排名的Top 10渠道的今日与昨日的活跃用户数量的占比。
- 累计用户：按照累计用户排名的Top 10渠道的累计用户数量的占比。

8.4.2 实时统计

实时统计页面主要展示今日实时的新增用户、启动次数等数据。

基本指标

展示截止至今日上个整点时刻的各指标的当日总数据，以及相对于昨日相同整点时刻的数据的变化率。



指标	说明
新增用户	第一次启动小程序的用户（以设备为判断标准）。
启动次数	打开小程序视为启动，完全退出或退至后台运行超过30秒后再次进入小程序，视为一次新启动。

指标	说明
时段累计日活	启动过小程序的用户（去重），启动过一次的用户即视为活跃用户，包括新用户与老用户。
分时活跃用户	活跃用户在24小时中的分布情况（每小时去重）。

8.5 用户分析

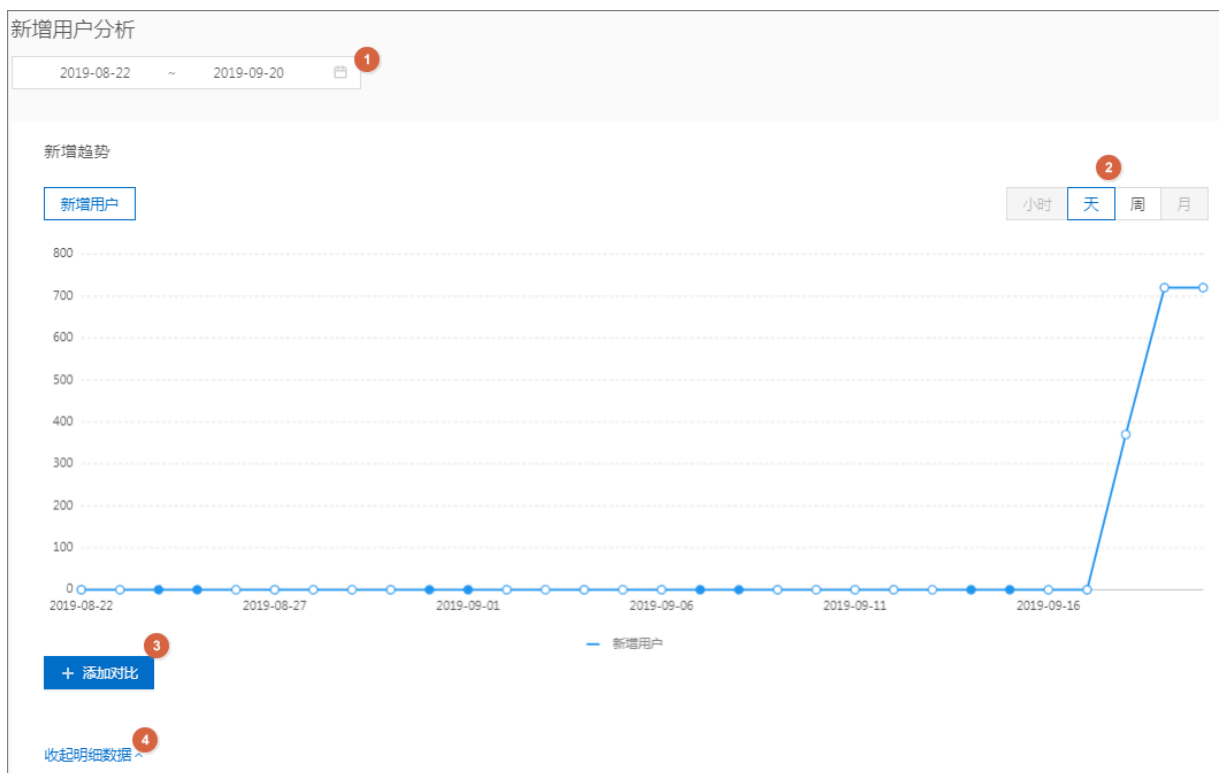
8.5.1 新增用户

新增用户页面展示实时的新增用户数据和次日留存率，帮助您了解新增用户质量。

新增用户分析

统计口径：首次启动进入小程序的设备视为一个新增用户。

您可以筛选查看不同时间段的新增用户，并添加时间对比数据。



配置	说明
1. 时段选择	选择特定时间维度，目前支持选择最大时间筛选为732天。

配置	说明
2. 时间颗粒度选择	按照不同颗粒度查看数据： · 当时间维度在0~7天时，可按小时查看。 小时数据最多展示7天并暂时不支持筛选。 · 当时间维度大于8天时，可按周查看。 按周显示新增用户时，界面上用每周的周日来代表该周。 · 当时间维度大于30天时，可按月查看。 按月显示新增用户时，界面上用每个月的第一日来代表该月。
3. 添加对比	支持渠道、时段对比。 选择时间对比可对所选时段的数据做上周同期以及上月同期数据对比。
4. 展开明细数据	单击此处可查看所选时间范围内的明细数据包括新增用户总数和新增用户占比。 新增用户占比指时段内新增用户占该时段内活跃用户的比例。

次日留存率

统计口径：该日的新增用户在次日仍启动小程序的用户占该日新增的比例。

您可以通过观测次日留存率了解新增用户质量。

日期	新增用户 (占比)
2019-09-19	720 (100%)
2019-09-18	370 (100%)
2019-09-17	0 (0%)
2019-09-16	0 (0%)
2019-09-15	0 (0%)
2019-09-14	0 (0%)
2019-09-13	0 (0%)
2019-09-12	0 (0%)

8.5.2 活跃用户

活跃用户页面展示活跃用户趋势、活跃用户构成等数据。

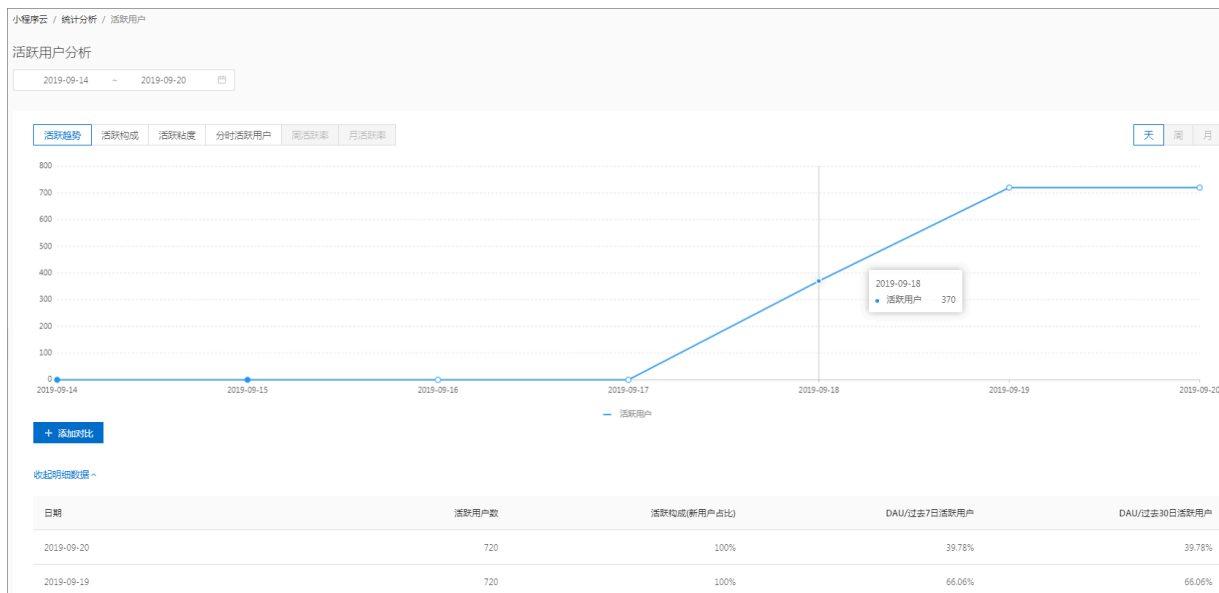
统计口径

所选时间段内，启动过小程序的用户（去重），启动过一次的用户即被视为活跃用户，包括新用户和老用户。

更新周期

- 日活跃用户实时更新。
- 周活跃用户每周日更新上周数据。
- 月活跃用户每月1日更新上月数据。

详细说明



配置	说明
1. 时段选择	选择特定时间维度，目前支持选择最大时间筛选为732天。
2. 指标选择	提供活跃用户、活跃用户构成、分时活跃用户、活跃粘度、用户周活跃率、用户月活跃率指标。您可以单击不同指标进行切换。  说明： · 分时活跃用户：只有所选时间段小于或等于7天时可查看整点的活跃用户数据。 · 周活跃率：只有所选时间段大于或等于8天时可查看。 · 月活跃率：只有所选时间段大于30天时可查看。
3. 时间颗粒度选择	按照不同颗粒度查看数据： · 当时间维度在0~7天时，可按小时查看。 小时数据最多展示7天并暂时不支持筛选。 · 当时间维度大于8天时，可按周查看。 按周显示新增用户时，界面上用每周的周日来代表该周。 · 当时间维度大于30天时，可按月查看。 按月显示新增用户时，界面上用每个月的第一日来代表该月。

配置	说明
4. 添加对比	支持渠道、时段对比。 选择时间对比可对所选时段的数据做上周同期以及上月同期数据对比。
5. 展开明细数据	单击此处可查看所选时间范围内的明细数据。 活跃用户占比指该日（周/月）活跃用户占所查看时段内每日（周/月）活跃用户加和的比例。

常见问题

1. 为什么活跃用户概况中的过去7天活跃用户数少于活跃用户明细中，每天活跃用户的加和总数？

7天活跃用户数是对这一周的活跃用户进行去重。而活跃用户明细中，是按天进行去重的。如果一个用户每天都使用小程序，那么每天加和的数量就要比7天活跃用户数多6个。

2. 我昨天夜里测试的小程序，但是为什么算在了今天的活跃中？

我们按零点划分日期的。在零点之后使用，则会被算为第二天的活跃用户。

8.5.3 活跃用户

活跃用户页面展示活跃用户趋势、活跃用户构成等数据。

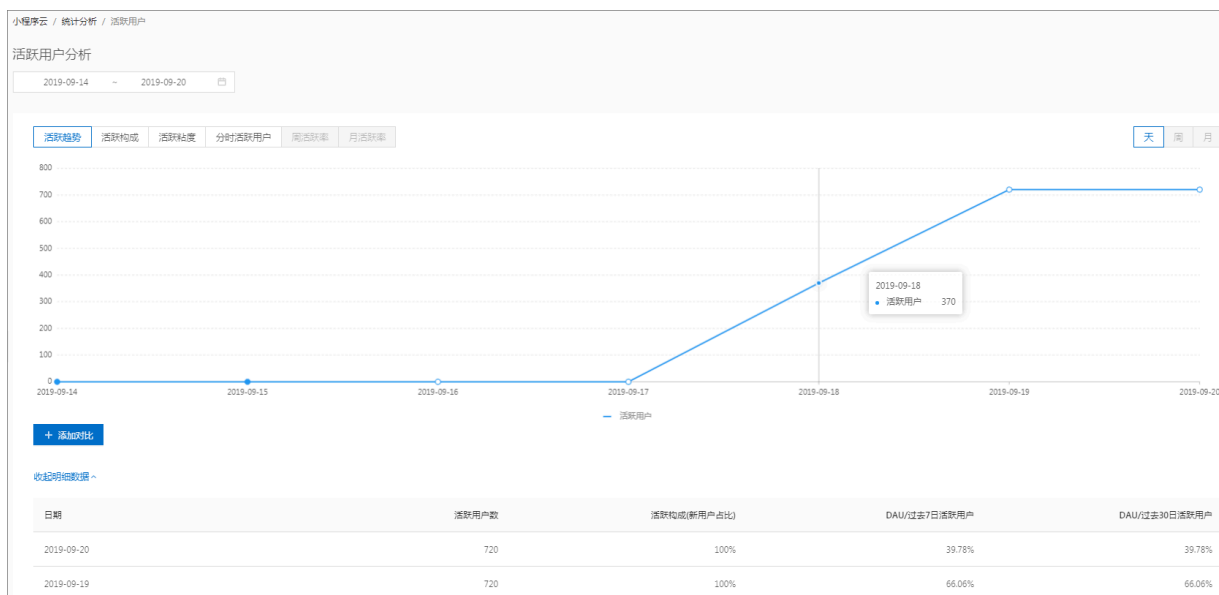
统计口径

所选时间段内，启动过小程序的用户（去重），启动过一次的用户即被视为活跃用户，包括新用户和老用户。

更新周期

- 日活跃用户实时更新。
- 周活跃用户每周日更新上周数据。
- 月活跃用户每月1日更新上月数据。

详细说明



配置	说明
1. 时段选择	选择特定时间维度，目前支持选择最大时间筛选为732天。
2. 指标选择	提供活跃用户、活跃用户构成、分时活跃用户、活跃粘度、用户周活跃率、用户月活跃率指标。您可以单击不同指标进行切换。  说明： · 分时活跃用户：只有所选时间段小于或等于7天时可查看整点的活跃用户数据。 · 周活跃率：只有所选时间段大于或等于8天时可查看。 · 月活跃率：只有所选时间段大于30天时可查看。
3. 时间颗粒度选择	按照不同颗粒度查看数据： · 当时间维度在0~7天时，可按小时查看。 小时数据最多展示7天并暂时不支持筛选。 · 当时间维度大于8天时，可按周查看。 按周显示新增用户时，界面上用每周的周日来代表该周。 · 当时间维度大于30天时，可按月查看。 按月显示新增用户时，界面上用每个月的第一日来代表该月。
4. 添加对比	支持渠道、时段对比。 选择时间对比可对所选时段的数据做上周同期以及上月同期数据对比。

配置	说明
5. 展开明细数据	单击此处可查看所选时间范围内的明细数据。 活跃用户占比指该日（周/月）活跃用户占所查看时段内每日（周/月）活跃用户加和的比例。

常见问题

1. 为什么活跃用户概况中的过去7天活跃用户数少于活跃用户明细中，每天活跃用户的加和总数？

7天活跃用户数是对这一周的活跃用户进行去重。而活跃用户明细中，是按天进行去重的。如果一个用户每天都使用小程序，那么每天加和的数量就要比7天活跃用户数多6个。

2. 我昨天夜里测试的小程序，但是为什么算在了今天的活跃中？

我们按零点划分日期的。在零点之后使用，则会被算为第二天的活跃用户。

8.6 活跃用户

活跃用户页面展示活跃用户趋势、活跃用户构成等数据。

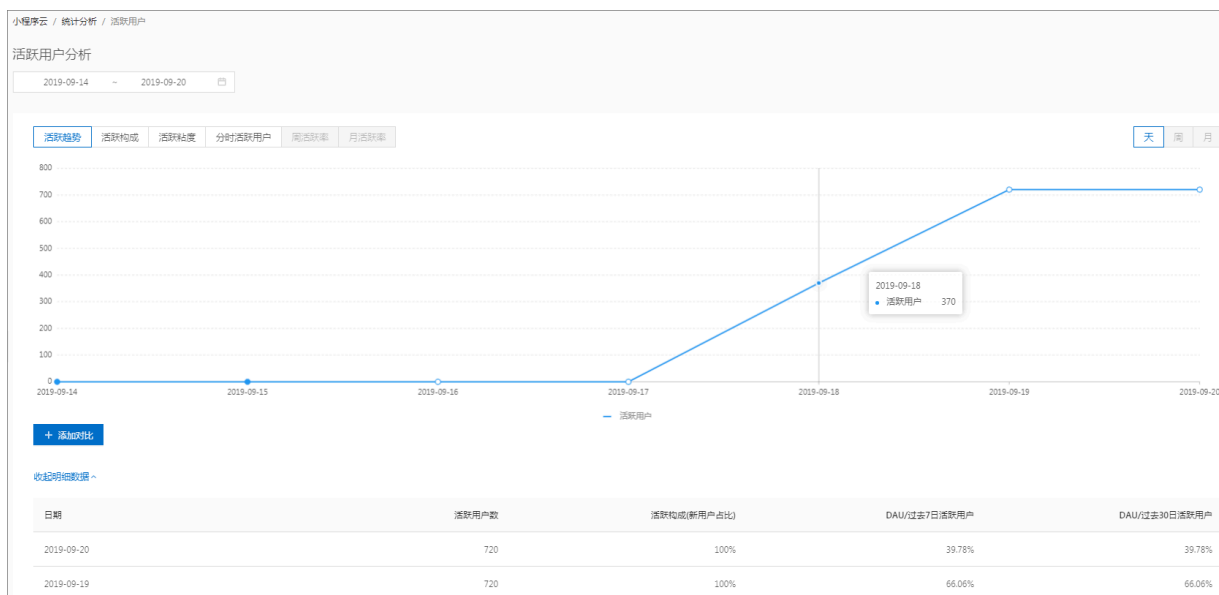
统计口径

所选时间段内，启动过小程序的用户（去重），启动过一次的用户即被视为活跃用户，包括新用户和老用户。

更新周期

- 日活跃用户实时更新。
- 周活跃用户每周日更新上周数据。
- 月活跃用户每月1日更新上月数据。

详细说明



配置	说明
1. 时段选择	选择特定时间维度，目前支持选择最大时间筛选为732天。
2. 指标选择	提供活跃用户、活跃用户构成、分时活跃用户、活跃粘度、用户周活跃率、用户月活跃率指标。您可以单击不同指标进行切换。  说明： · 分时活跃用户：只有所选时间段小于或等于7天时可查看整点的活跃用户数据。 · 周活跃率：只有所选时间段大于或等于8天时可查看。 · 月活跃率：只有所选时间段大于30天时可查看。
3. 时间颗粒度选择	按照不同颗粒度查看数据： · 当时间维度在0~7天时，可按小时查看。 小时数据最多展示7天并暂时不支持筛选。 · 当时间维度大于8天时，可按周查看。 按周显示新增用户时，界面上用每周的周日来代表该周。 · 当时间维度大于30天时，可按月查看。 按月显示新增用户时，界面上用每个月的第一日来代表该月。
4. 添加对比	支持渠道、时段对比。 选择时间对比可对所选时段的数据做上周同期以及上月同期数据对比。

配置	说明
5. 展开明细数据	单击此处可查看所选时间范围内的明细数据。 活跃用户占比指该日（周/月）活跃用户占所查看时段内每日（周/月）活跃用户加和的比例。

常见问题

1. 为什么活跃用户概况中的过去7天活跃用户数少于活跃用户明细中，每天活跃用户的加和总数？

7天活跃用户数是对这一周的活跃用户进行去重。而活跃用户明细中，是按天进行去重的。如果一个用户每天都使用小程序，那么每天加和的数量就要比7天活跃用户数多6个。

2. 我昨天夜里测试的小程序，但是为什么算在了今天的活跃中？

我们按零点划分日期的。在零点之后使用，则会被算为第二天的活跃用户。

8.6.1 活跃用户

活跃用户页面展示活跃用户趋势、活跃用户构成等数据。

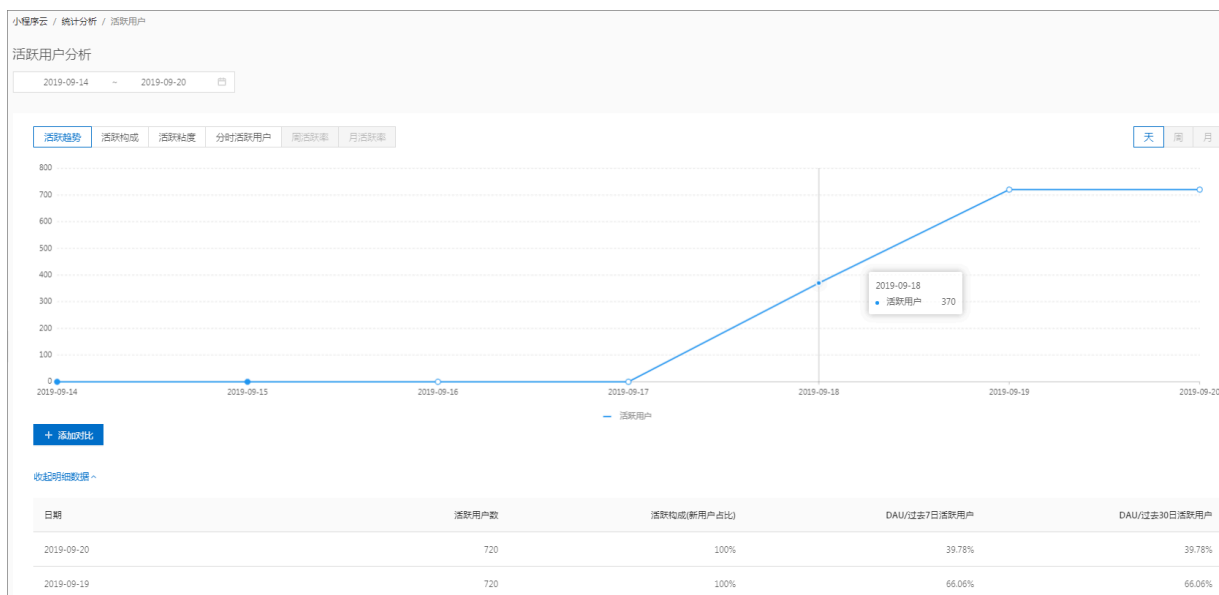
统计口径

所选时间段内，启动过小程序的用户（去重），启动过一次的用户即被视为活跃用户，包括新用户和老用户。

更新周期

- 日活跃用户实时更新。
- 周活跃用户每周日更新上周数据。
- 月活跃用户每月1日更新上月数据。

详细说明



配置	说明
1. 时段选择	选择特定时间维度，目前支持选择最大时间筛选为732天。
2. 指标选择	提供活跃用户、活跃用户构成、分时活跃用户、活跃粘度、用户周活跃率、用户月活跃率指标。您可以单击不同指标进行切换。  说明： · 分时活跃用户：只有所选时间段小于或等于7天时可查看整点的活跃用户数据。 · 周活跃率：只有所选时间段大于或等于8天时可查看。 · 月活跃率：只有所选时间段大于30天时可查看。
3. 时间颗粒度选择	按照不同颗粒度查看数据： · 当时间维度在0~7天时，可按小时查看。 小时数据最多展示7天并暂时不支持筛选。 · 当时间维度大于8天时，可按周查看。 按周显示新增用户时，界面上用每周的周日来代表该周。 · 当时间维度大于30天时，可按月查看。 按月显示新增用户时，界面上用每个月的第一日来代表该月。
4. 添加对比	支持渠道、时段对比。 选择时间对比可对所选时段的数据做上周同期以及上月同期数据对比。

配置	说明
5. 展开明细数据	单击此处可查看所选时间范围内的明细数据。 活跃用户占比指该日（周/月）活跃用户占所查看时段内每日（周/月）活跃用户加和的比例。

常见问题

1. 为什么活跃用户概况中的过去7天活跃用户数少于活跃用户明细中，每天活跃用户的加和总数？

7天活跃用户数是对这一周的活跃用户进行去重。而活跃用户明细中，是按天进行去重的。如果一个用户每天都使用小程序，那么每天加和的数量就要比7天活跃用户数多6个。

2. 我昨天夜里测试的小程序，但是为什么算在了今天的活跃中？

我们按零点划分日期的。在零点之后使用，则会被算为第二天的活跃用户。

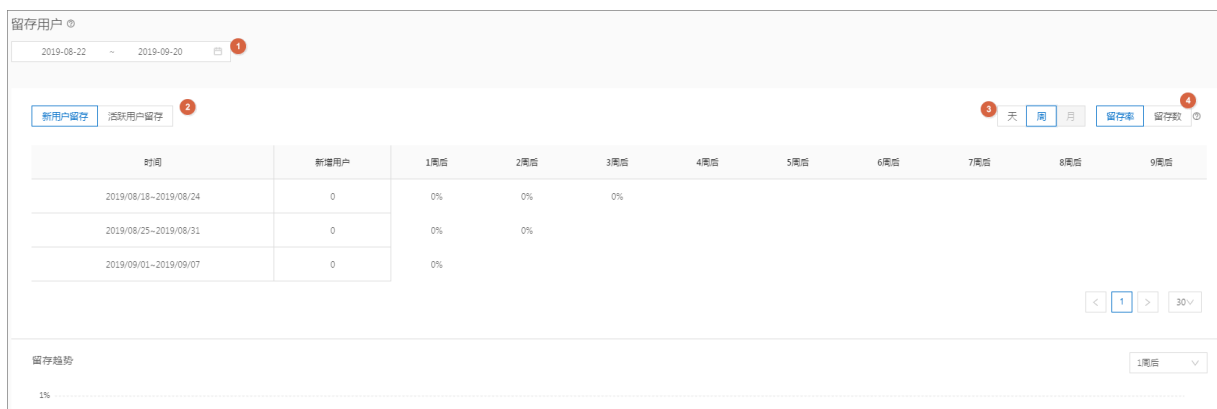
8.7 留存用户

留存用户页面展示新用户留存和活跃用户留存数据。

更新周期

留存用户的数据每日更新一次。

详细说明



配置	说明
1. 时段选择	选择特定时间维度，目前支持选择最大时间筛选为732天。
2. 留存用户选择	- 新用户留存：某段时间内的新增用户，经过一段时间后，又继续使用小程序的被认作是留存用户。 - 活跃用户留存：某段时间内的活跃用户，经过一段时间后，又继续使用小程序的被认作是留存用户。

配置	说明
3. 时间颗粒度选择	按照不同颗粒度查看数据。选择周（月）时，展示的是该自然周（月）的新增用户，在之后的自然周（月）的留存情况。
4. 选择数据粒度	- 留存率：某段时间内的新增用户（活跃用户），经过一段时间后，又继续使用小程序的比例。 - 留存数：某段时间内的新增用户（活跃用户），经过一段时间后，又继续使用小程序的用户数。 例如，5月份新增用户200，这200人在6月份启动过小程序的有100人，7月份启动过小程序的有80人，8月份启动过小程序的有50人；则5月新增用户一个月后的留存率是50%，两个月后的留存率是40%，三个月后的留存率是25%。

常见问题

1. 留存用户列表里不同的颜色表示什么？

留存率越高，标注的颜色越深，以此直观区分留存的变化情况。

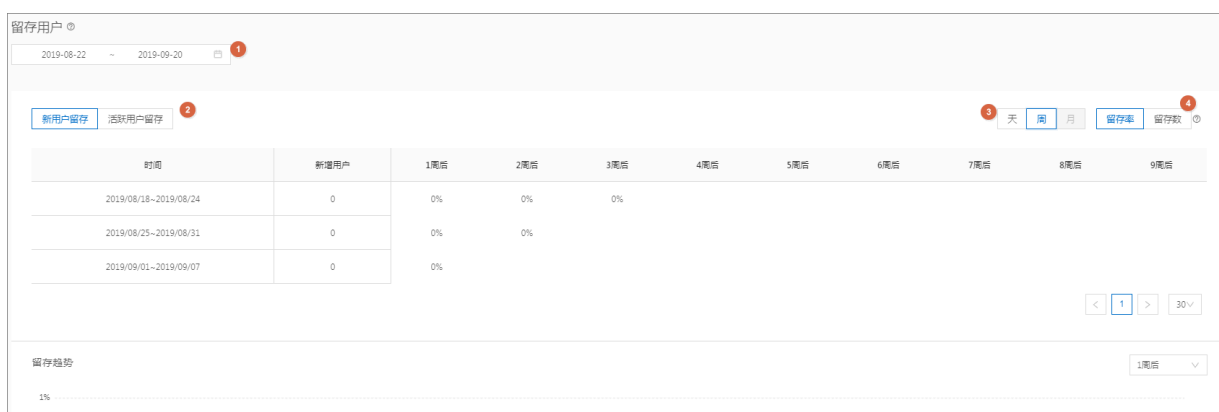
8.7.1 留存用户

留存用户页面展示新用户留存和活跃用户留存数据。

更新周期

留存用户的数据每日更新一次。

详细说明



配置	说明
1. 时段选择	选择特定时间维度，目前支持选择最大时间筛选为732天。

配置	说明
2. 留存用户选择	· 新用户留存：某段时间内的新增用户，经过一段时间后，又继续使用小程序的被认作是留存用户。 · 活跃用户留存：某段时间内的活跃用户，经过一段时间后，又继续使用小程序的被认作是留存用户。
3. 时间颗粒度选择	按照不同颗粒度查看数据。选择周（月）时，展示的是该自然周（月）的新增用户，在之后的自然周（月）的留存情况。
4. 选择数据粒度	· 留存率：某段时间内的新增用户（活跃用户），经过一段时间后，又继续使用小程序的比例。 · 留存数：某段时间内的新增用户（活跃用户），经过一段时间后，又继续使用小程序的用户数。 例如，5月份新增用户200，这200人在6月份启动过小程序的有100人，7月份启动过小程序的有80人，8月份启动过小程序的有50人；则5月新增用户一个月后的留存率是50%，两个月后的留存率是40%，三个月后的留存率是25%。

常见问题

1. 留存用户列表里不同的颜色表示什么？

留存率越高，标注的颜色越深，以此直观区分留存的变化情况。

8.8 渠道

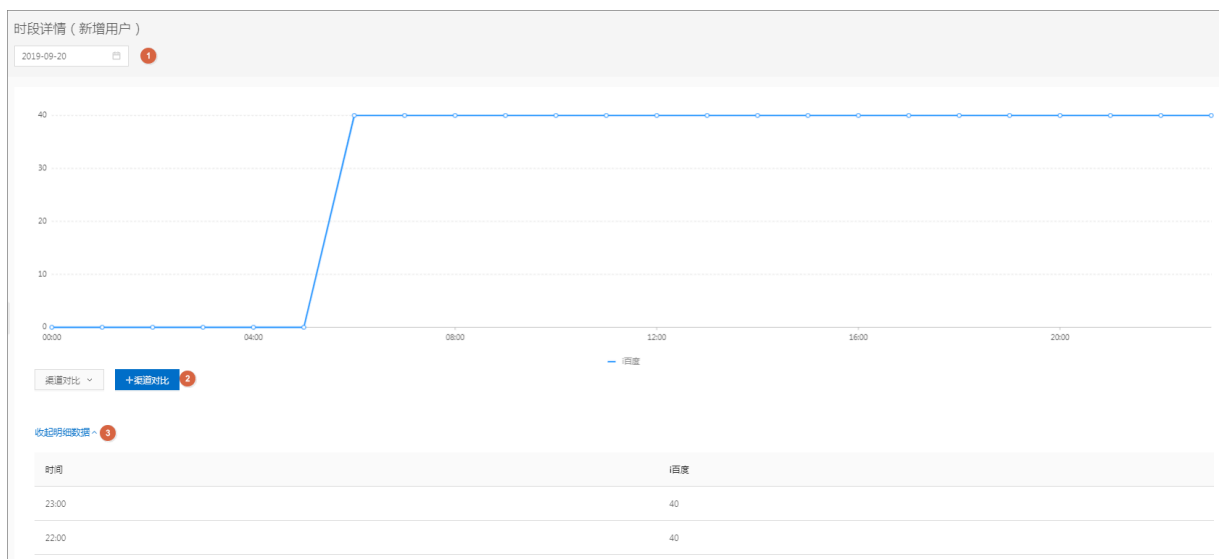
8.8.1 时段详情

时段详情页面展示不同渠道在选定时间内的新增用户数据。

统计口径

首次启动进入小程序的设备视为一个新增用户。

详细说明

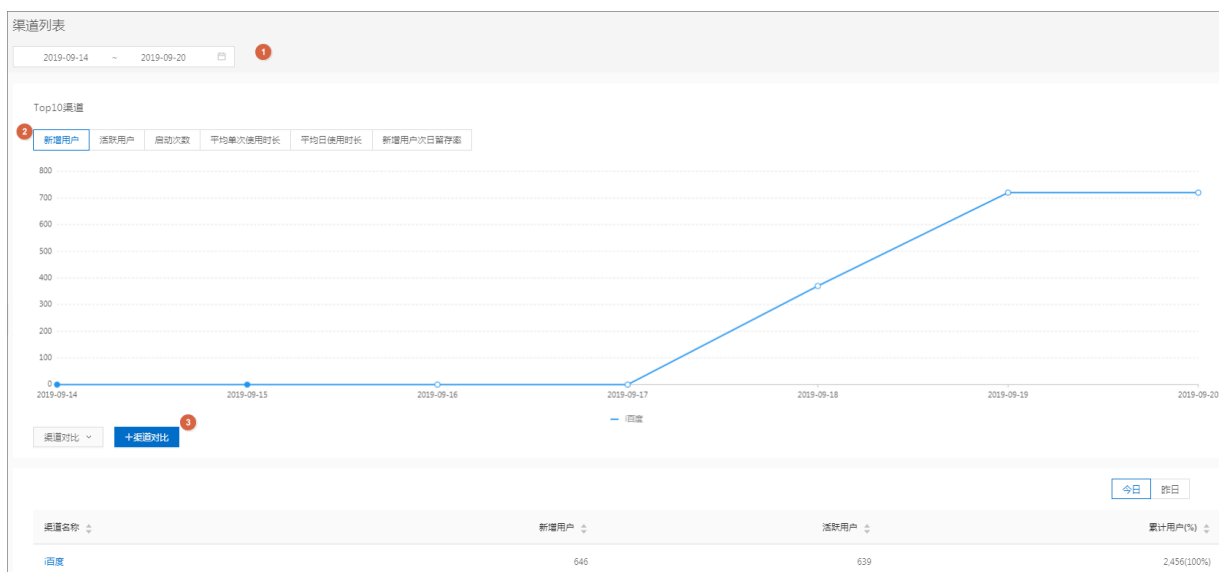


配置	说明
1. 时段选择	选择时间段，支持选择任意一天。
2. 渠道对比	对比任意所选渠道在特定时间内的新增用户数据。
3. 明细数据	各渠道的新增用户明细。

8.8.2 渠道列表

渠道列表页面展示Top 10渠道的新增、活跃等数据，帮助您了解该渠道用户的粘着度、忠诚度，进而了解渠道推广的质量，为您调整渠道推广策略提供依据。

详细说明



配置	说明
1. 时段选择	选择特定时间维度，目前支持选择最大时间筛选为732天。
2. 指标选择	· 新用户：第一次启动小程序的用户（以设备为判断标准）。 · 活跃用户：启动过小程序的用户（去重），启动过一次的用户即视为活跃用户，包括新用户与老用户。 · 平均单次使用时长：单次使用时长的均值，即全部用户的日使用时长/总启动次数。 · 平均日使用时长：日使用时长的均值，即全部用户的日使用时长/总活跃用户。 · 新增用户次日留存率：该日的新增用户在次日仍启动小程序的用户占该日新增的比例。
3. 渠道对比	可选择其他渠道来进行数据对比。

9 跨端配置指南

9.1 uni-app跨平台开发扩展使用教程

uni-app跨平台开发扩展支持在阿里云小程序开发者工具中将uni-app工程编译为微信、百度、字节跳动小程序。

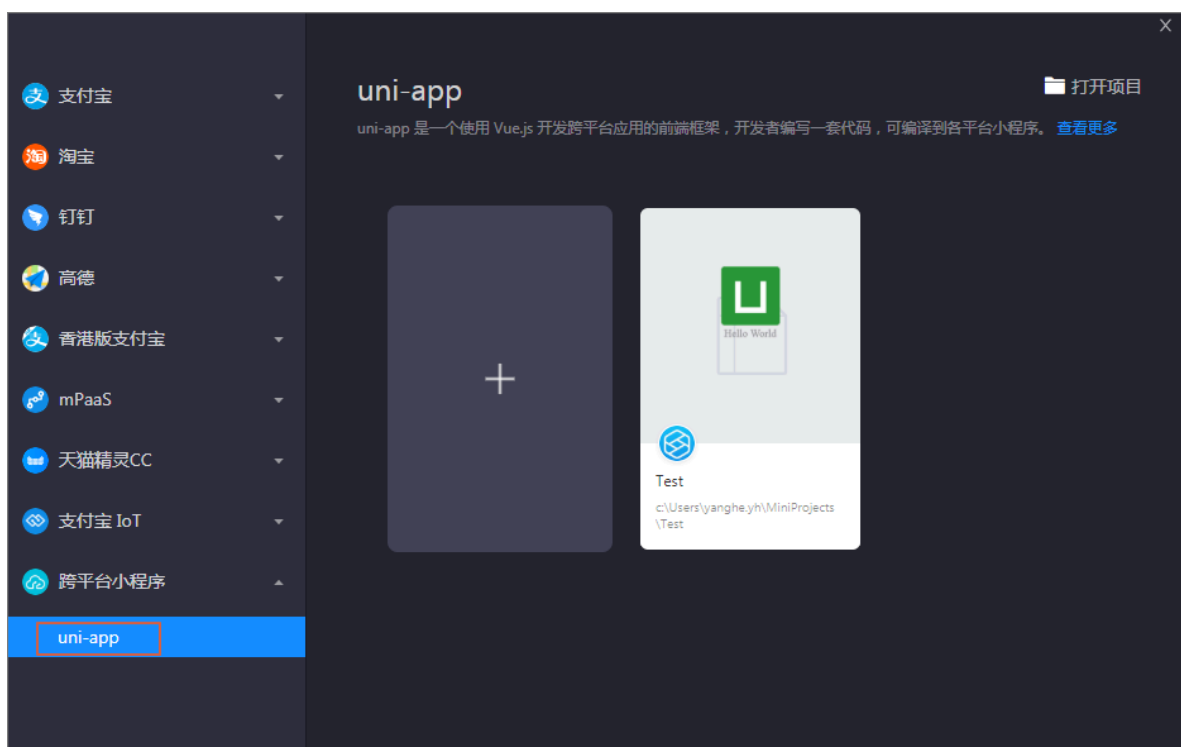
前提条件

确保您安装的是最新版本小程序开发者工具。访问这里下载最新版本：<https://docs.alipay.com/mini/ide/download>

步骤一：安装uni-app跨平台开发扩展

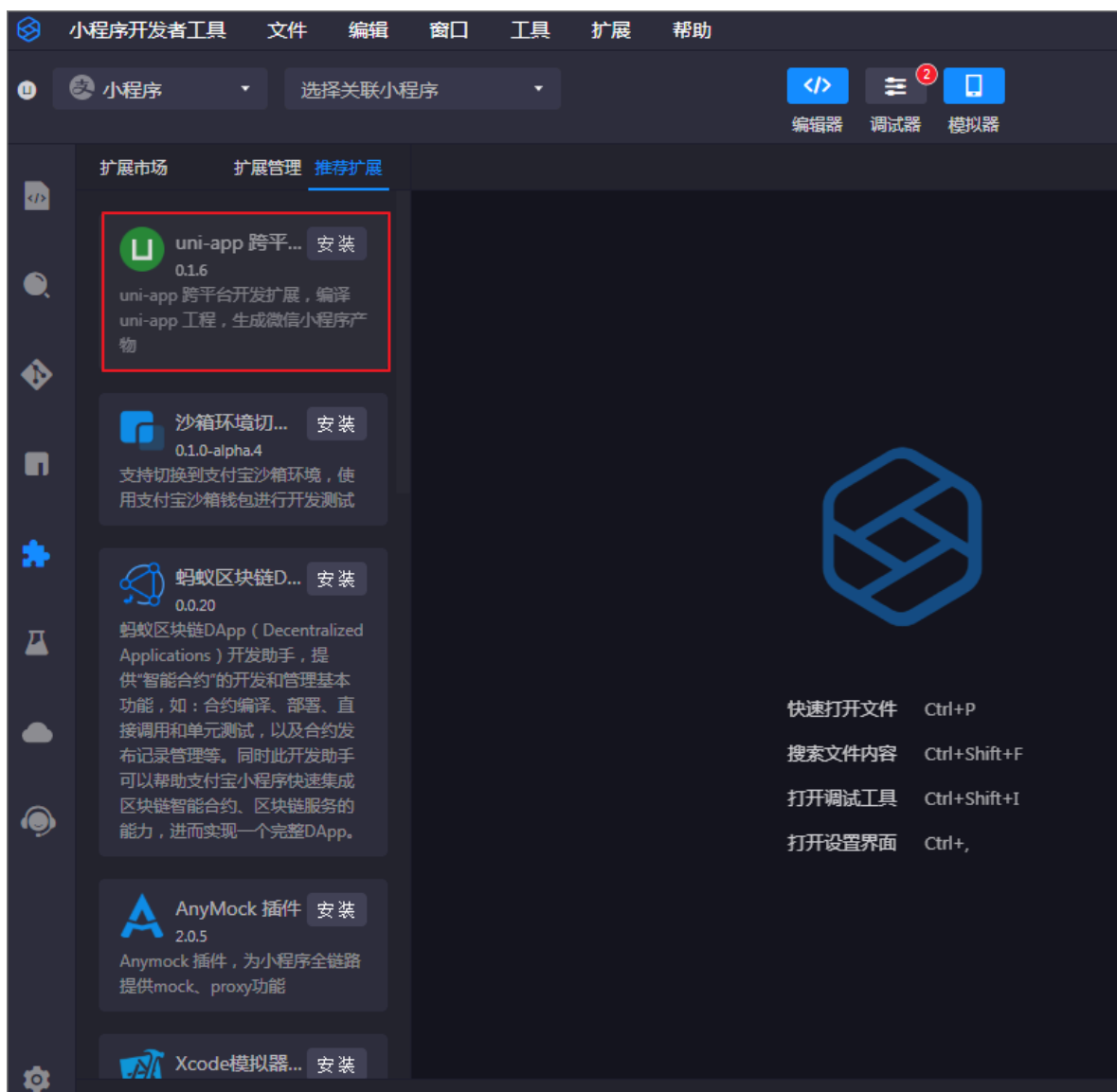
完成以下操作，安装uni-app跨平台开发扩展：

1. 打开小程序开发者工具，选择跨平台小程序 > uni-app。



2. 单击+创建小程序项目。

3. 在左侧工具栏，单击插件市场图标，单击uni-app跨平台开发扩展的安装按钮。



4. 安装完成后，单击启用。

步骤二：开启微信小程序编译

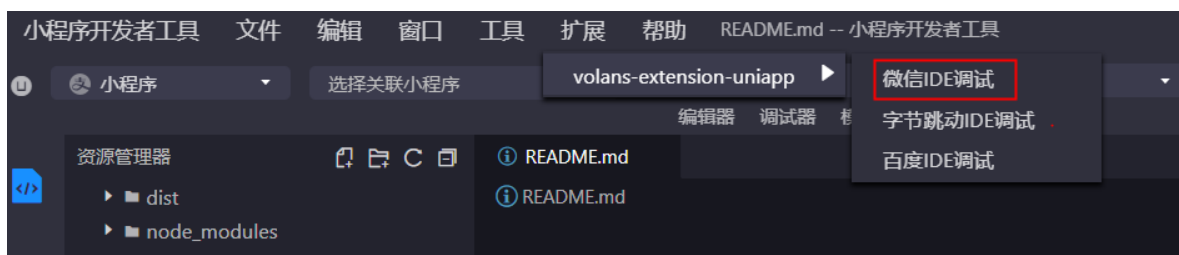
启用后，完成以下步骤开启微信小程序编译：

1. 在IDE上部菜单栏，单击扩展 > volans-extension-uniapp > 微信IDE调试。



说明：

确保您打开的是小程序uni-app项目工程，否则无法打开uni-app跨平台开发扩展配置页面。



2. 在微信开发者工具文本框中输入微信开发者工具的安装路径，然后单击保存。

微信开发者工具的默认安装路径如下，请根据实际情况输入。

- **MacOS:** `/Applications/wechatwebdevtools.app`
- **Windows:** `C:/Program Files (x86)/Tencent/##web####`



注意:

- 只有首次使用时需要配置IDE安装路径。
- 微信开发者工具必须打开服务端口，否则无法唤醒IDE。详细信息，请参见[微信 IDE - 安全设置](#)。



3. 单击开始编译。

编译后，若IDE路径配置正确，并自动打开微信小程序IDE，并加载编译后的微信小程序产物。

步骤三：开启字节跳动小程序编译

完成以下操作，开启字节跳动编译：

1. 在跨平台开发扩展页面，选择字节跳动。



跨平台开发扩展

我们支持将 uni-app 工程编译成微信、字节跳动、百度等小程序代码工程。

1. 请选择平台: ☐ 微信 ☒ 字节跳动 ☐ 百度

2. 依赖检查: 已安装依赖包

3. 编译: 开发编译 生产编译

4. 产物路径: c:\Users\yanghe.yh\MiniProjects\uniappLoginTemplate\dist\dev\mp-toutiao

2. 单击点击安装依赖包进行安装。
3. 根据需要单击开发编译或生产编译。

开发编译和生产编译，产物路径是相同的。

步骤四：开启百度小程序编译

完成以下操作，开启百度编译：

1. 在跨平台开发扩展页面，选择百度。



跨平台开发扩展

我们支持将 uni-app 工程编译成微信、字节跳动、百度等小程序代码工程。

1. 请选择平台: ☐ 微信 ☐ 字节跳动 ☒ 百度

2. 依赖检查: 点击安装依赖包

3. 编译: 开发编译 生产编译

4. 产物路径: c:\Users\yanghe.yh\MiniProjects\uniappLoginTemplate\dist\dev\mp-baidu

运行步骤

- 编译生成百度小程序 未开始
- 打开百度开发者工具 未开始

2. 单击安装依赖包进行安装。
3. 根据需要单击开发编译或生产编译。

开发编译和生产编译，产物路径是相同的。

9.2 开发多端小程序教程

本教程指导您在阿里云小程序开发者工具中使用uni-app跨平台开发扩展插件开发多端小程序。

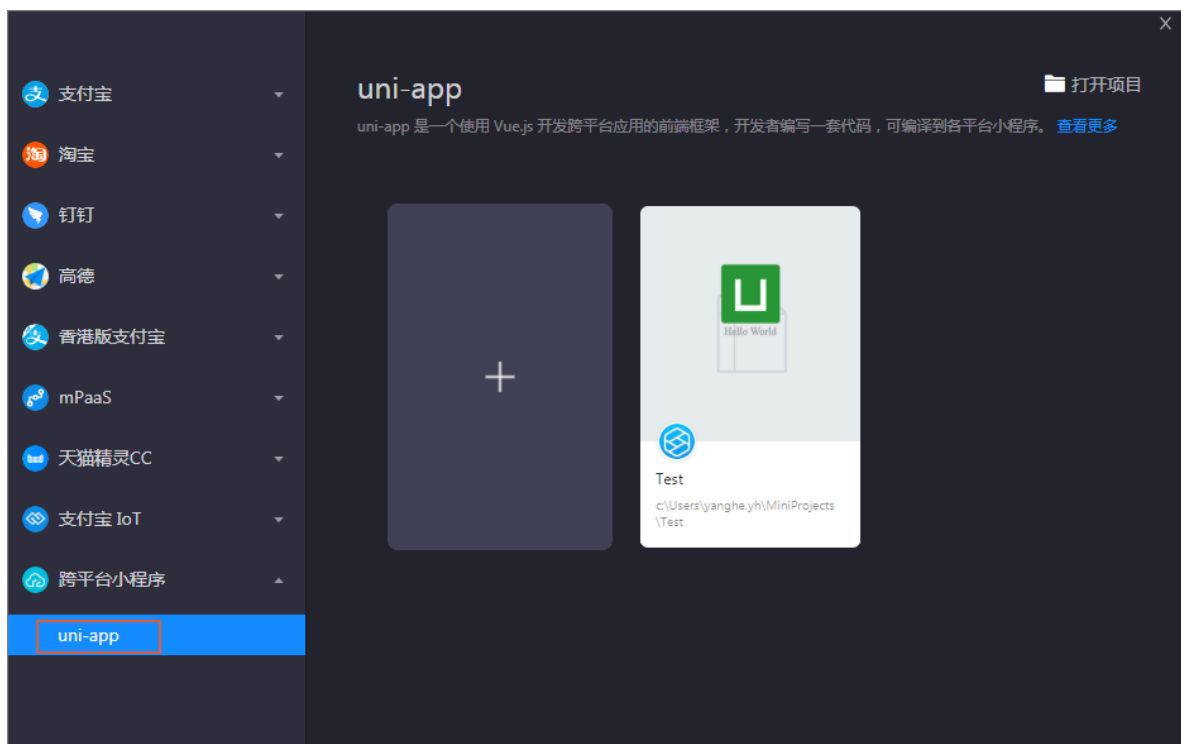
前提条件

- 确保您安装的是最新版本小程序开发者工具。访问这里下载最新版本：<https://docs.alipay.com/mini/ide/download>
- 完成uni-app跨平台开发扩展的安装和编译。详情请参见[uni-app跨平台开发扩展使用教程](#)。

步骤一：创建和配置小程序项目

完成以下操作，创建小程序项目：

1. 打开小程序开发者工具，选择跨平台小程序 > uni-app。



2. 单击+创建小程序项目。
3. 在左侧工具栏单击npm模块，然后选择安装全部项目依赖。

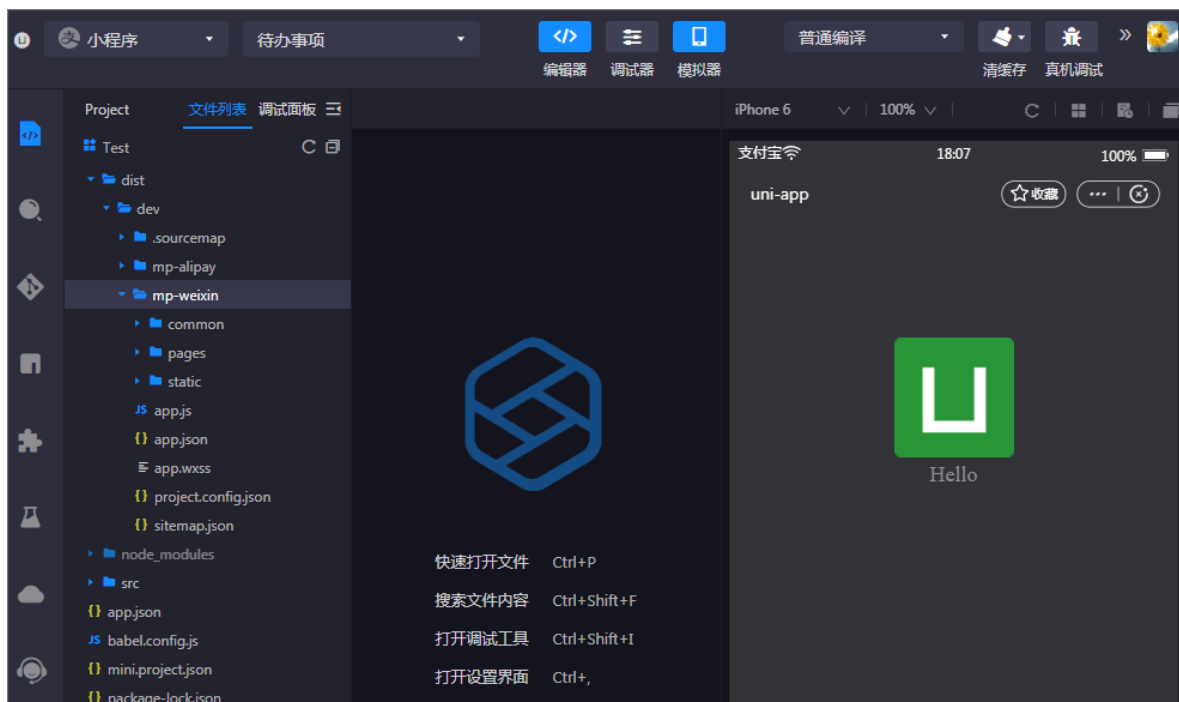


说明：

依赖安装时间可能较长，和网络环境有关，请耐心等待。



依赖安装成功后，编译报错消失，如下图所示。



uni-app项目创建成功后，您就可以开发uni-app跨端工程了。主要项目目录如下：

- `src/`：uni-app工程的源码目录。详细信息，请参见[uni-app 工程目录结构](#)。

· `dist/`: 小程序构建的文件:

- `dev/mp-alipay`: 支付宝小程序结构文件。
- `dev/mp-weixin`: 微信小程序结构文件, 需要结合uni-app插件使用。详细使用说明, 请参见[uni-app跨平台开发扩展使用教程](#)。



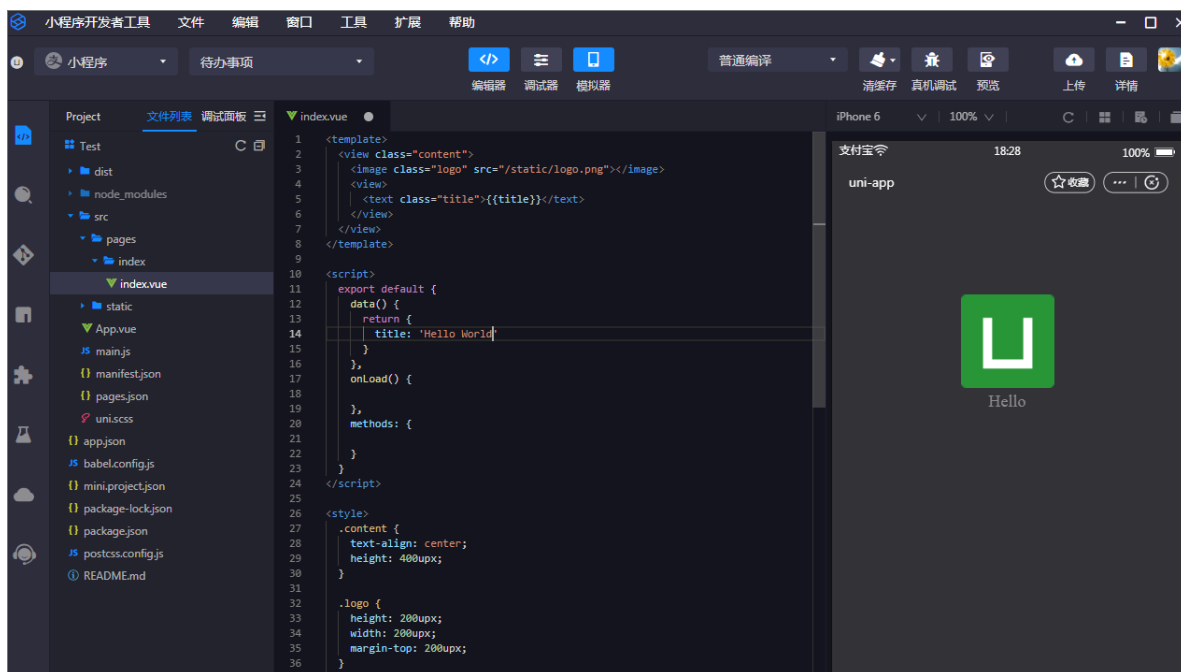
说明:

目前仅支持开发支付宝和微信端小程序。

步骤二: 开发小程序

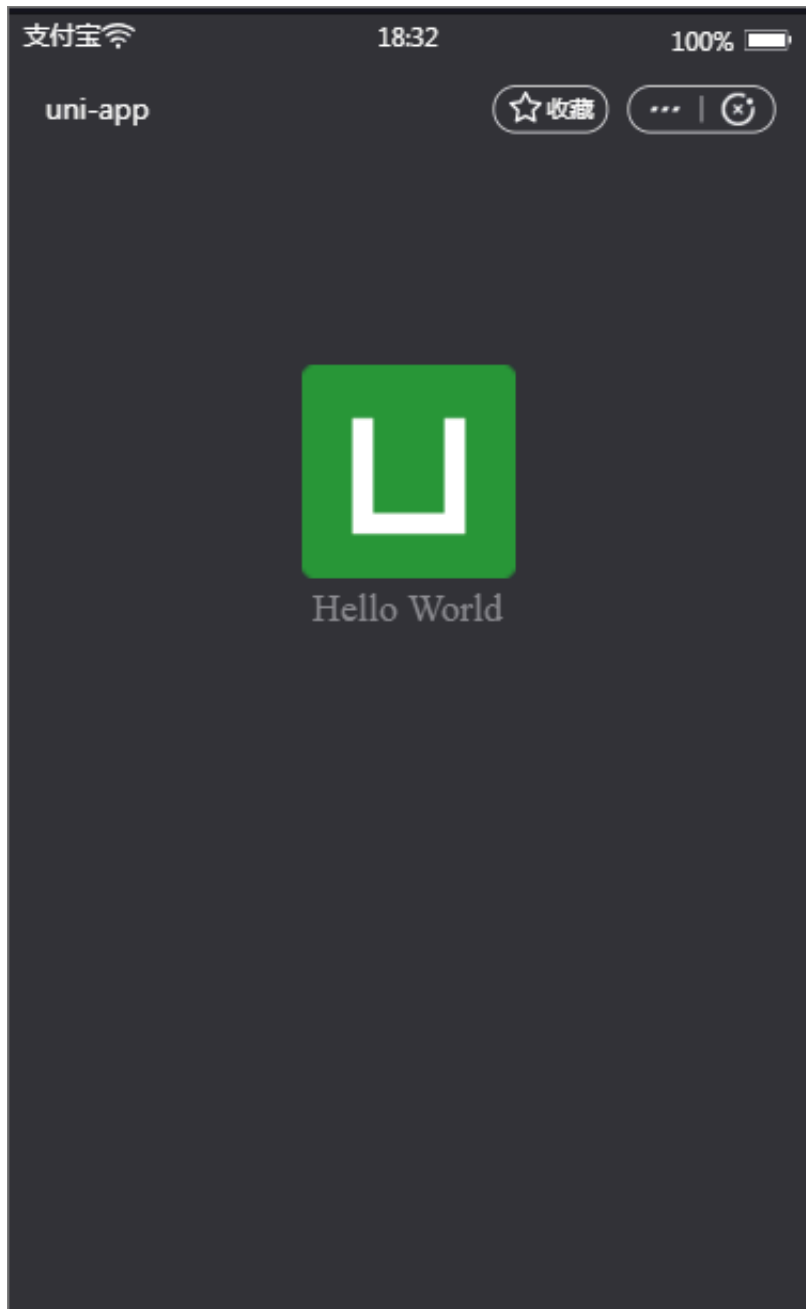
您可以在小程序开发者工具中直接进行代码开发。本教程中以小程序开发者工具中自带的小程序为例。

1. 在小程序开发者工具中, 打开`src/pages/index/index.vue`文件, 并将title修改为Hello World。



2. 保存修改。

IDE会触发自动编译，编译完成后的模拟器效果，如下图所示。



步骤三：预览和调试

完成以下操作，调试小程序：

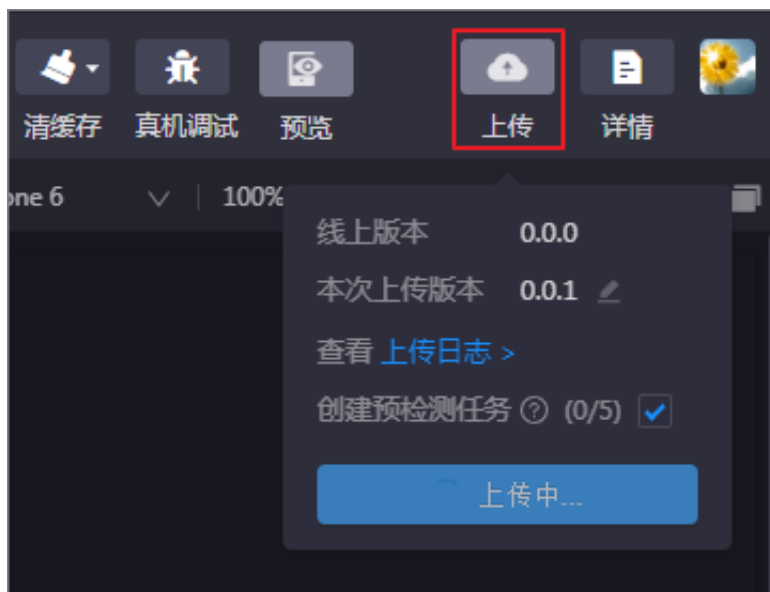
1. 在小程序开发者工具顶部菜单栏，单击预览，默认选中自动预览。



2. 编译完成后，请保持手机支付宝在前台运行，小程序会自动显示。您也可以单击扫码预览，然后使用支付宝客户端扫描二维码预览小程序。

步骤四：上传小程序

小程序调试完成后，您可以直接单击上传将开发好的小程序上传至开放平台。



使用咨询

如果在跨端开发中遇到任何问题，您可以通过搜索钉钉群（阿里小程序跨端IDE体验群，群号：23356164）或使用钉钉扫描二维码入群咨询。

阿里小程序跨端IDE体...



 扫一扫群二维码，立刻加入该群。

9.3 开发者工具Vue扩展使用教程

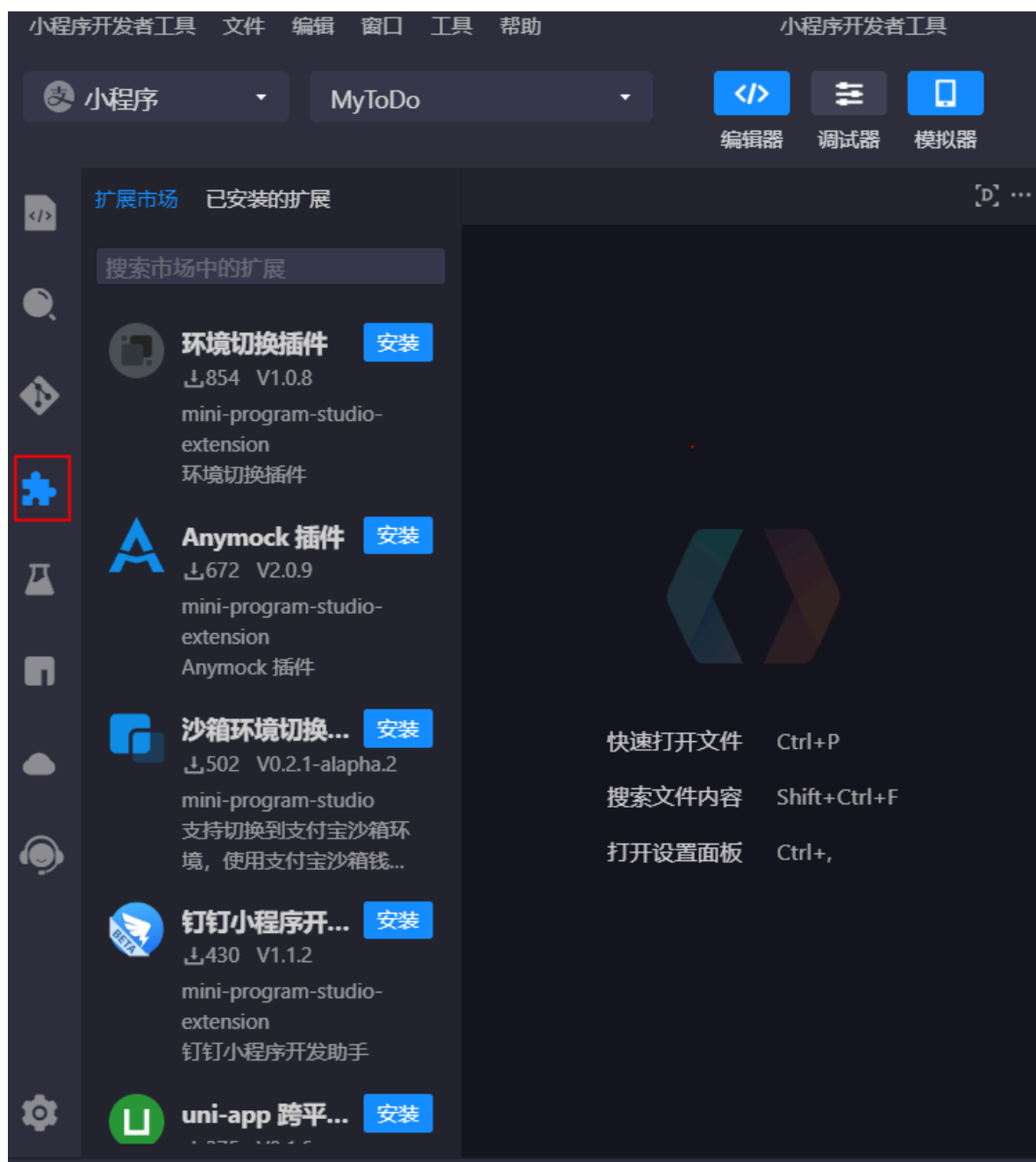
小程序开发者工具1.0版本，需要安装IDE扩展以便对Vue文件有更好的语法支持。只有首次使用时需要安装。

背景信息

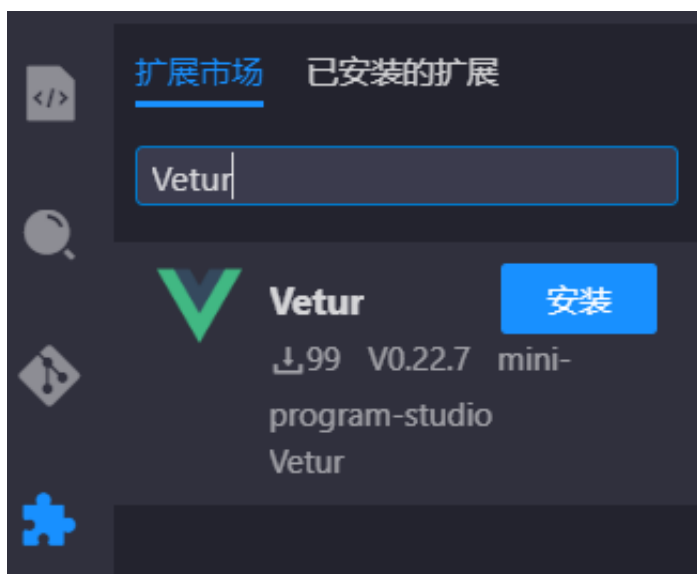
访问[这里](#)下载并安装阿里云小程序开发者工具1.0版本。

操作步骤

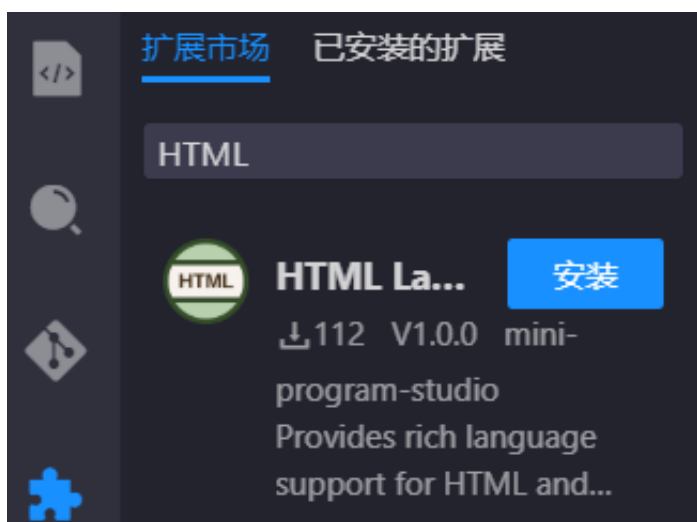
1. 打开阿里云小程序开发者工具，然后在左侧导航栏单击扩展市场图标。



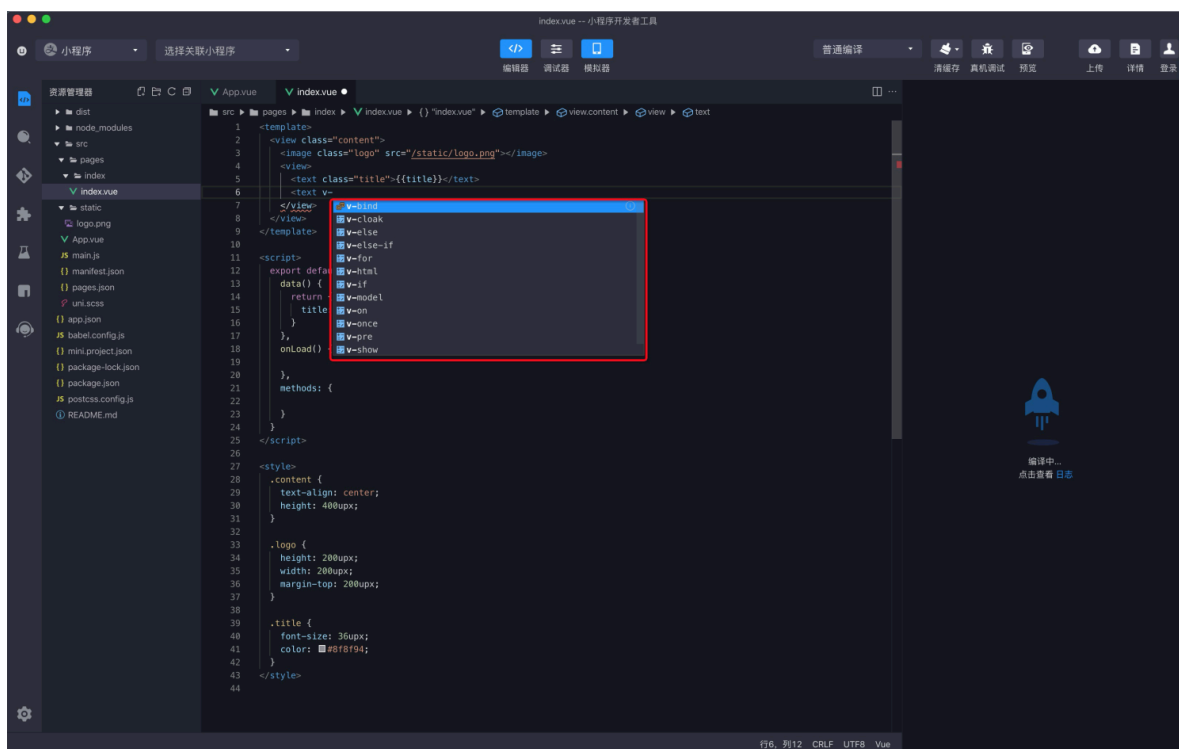
2. 搜索Vetur，安装Vetur插件。



3. 搜索HTML，安装HTML Language Features插件。



4. 重启IDE，使安装的扩展生效。进入Vue文件编译，即可体验语法高亮、语法提示等支持。



10 设置小程序

10.1 添加支付宝小程序

10.1.1 添加支付宝小程序

在使用小程序Serverless开发支付宝小程序前，您需要在控制台配置支付宝小程序信息。

前提条件

您已经在[蚂蚁金服开放平台](#)创建了小程序，并为支付宝小程序配置了公钥。

操作步骤

1. 登录[小程序云控制台](#)。
2. 在左侧导航栏，单击小程序Serverless > 设置。
3. 单击支付宝页签，然后单击添加。
4. 输入您的私钥。

您可以通过RSA密钥生成工具生成密钥，详情请参见[生成支付宝RSA密钥](#)。

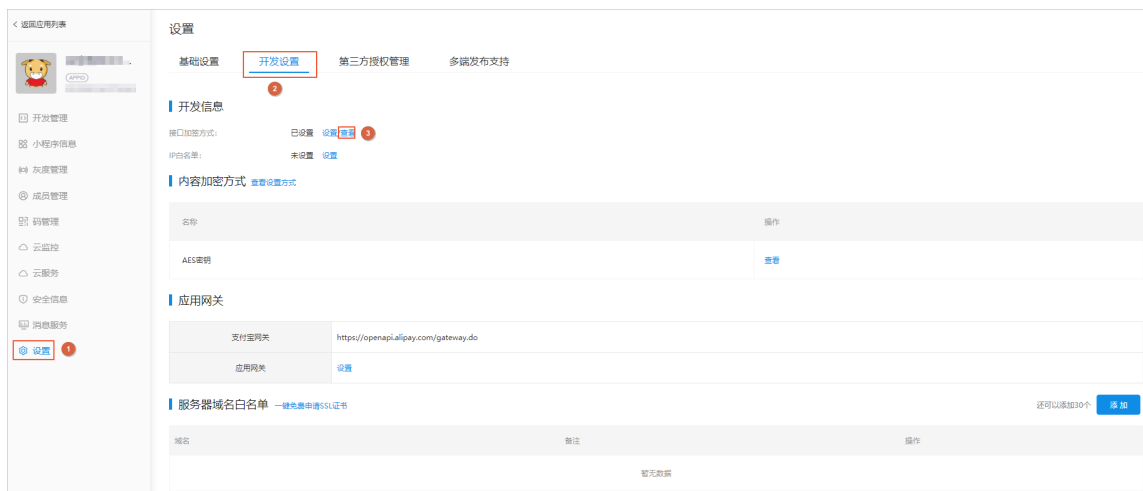


注意：

密钥格式必须为PKCS8（Java适用）。

5. 输入支付宝小程序配置的公钥。参考以下步骤查看公钥：

- 登录[蚂蚁金服开放平台](#)。
- 在支付宝小程序菜单栏单击设置，然后单击开发设置。
- 在接口加签方式区域，单击查看。



- 复制支付宝公钥信息，然后粘贴到阿里云支付宝小程序的配置中。



10.1.2 生成支付宝RSA密钥

支付宝提供一键生成RSA密钥工具，方便您在开发时使用。

操作步骤

1. 下载密钥生成工具。

- Windows: [密钥生成工具](#)



说明:

Windows版本工具请不要安装在含有空格的目录路径下。

- MacOS: [密钥生成工具](#)

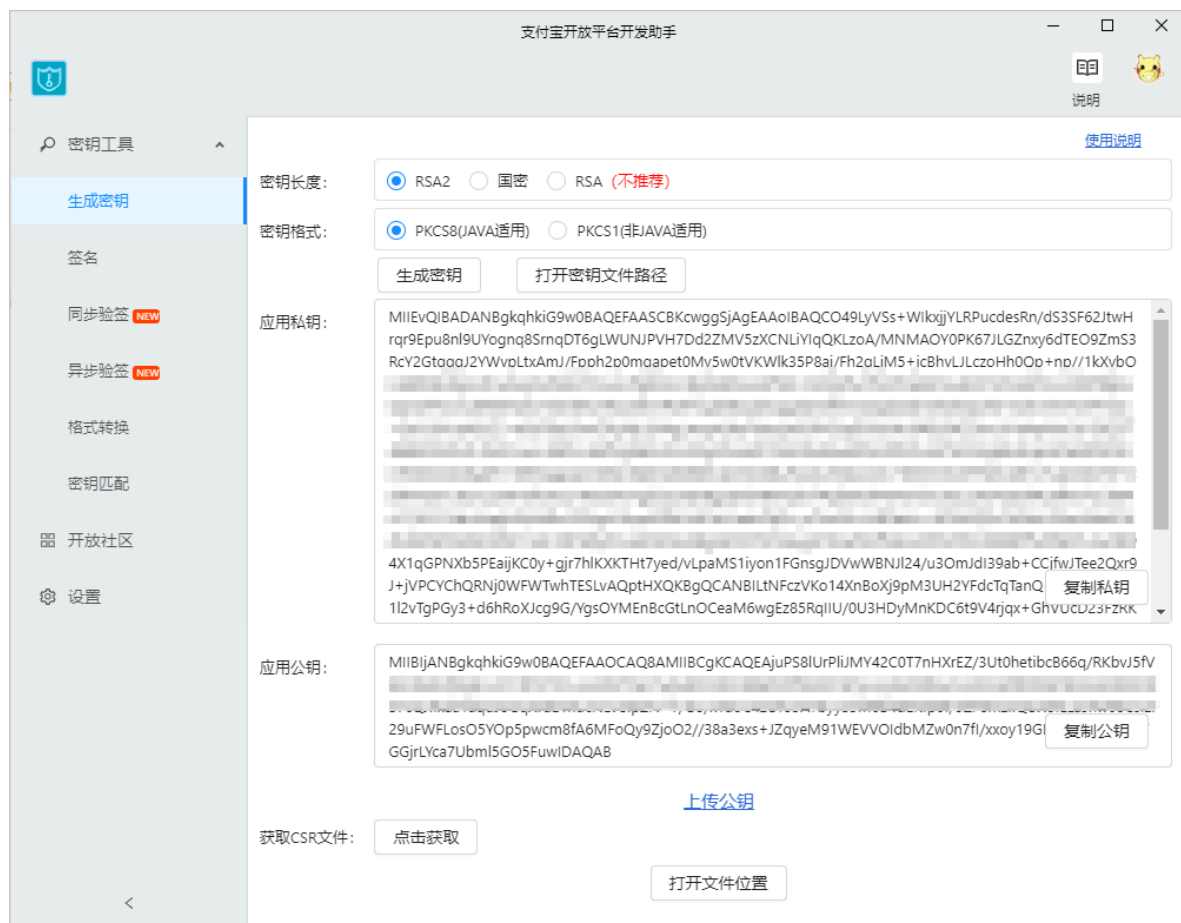
2. 运行AlipayDevelopmentAssistant文件。

3. 在生成密钥页面，密钥长度选择RSA2，密钥格式选择PKCS8（Java适用），然后单击生成密钥。



说明:

推荐使用RSA2加密，有关RSA2和RSA算法请参见[RSA](#) 和 [RSA2 签名算法区别](#)。



4. 单击打开密钥文件路径查看生成的公钥和密钥。

应用公钥2048.txt	2019/6/24 14:10	文本文档	1 KB
应用私钥2048.txt	2019/6/24 14:10	文本文档	2 KB

10.2 添加微信小程序

在使用小程序Serverless开发微信小程序前，您需要在控制台配置微信小程序信息。

前提条件

已完成微信小程序注册并完成认证。详细信息请参见[注册微信小程序](#)。

操作步骤

1. 登录[小程序云控制台](#)。
2. 在左侧导航栏，单击小程序Serverless > 设置。
3. 单击微信页签，然后单击添加密钥。
4. 在添加微信密钥页面，输入以下信息，然后单击确定。

- AppID：输入微信小程序的App ID。

登录[微信公众平台](#)，在开发设置页面查看App ID。



- App Secret：输入微信小程序的密钥。

添加微信密钥

* AppID

请输入小程序的 AppID

0/128

* App Secret

请输入小程序的 App Secret

后续步骤

[在微信小程序中使用SDK](#)

11 SDK 文档

11.1 变更历史

记录各版本SDK的新增特性。

版本	版本号	说明
客户端SDK 2.2	@alicloud/mpserverless-sdk@2.2.0	增加统计分析功能。更多信息请参见 安装客户端SDK2.2版本 。
客户端SDK 2.1	@alicloud/mpserverless-sdk@2.1.0	增加匿名用户授权方式。 更多信息，请参见 #unique_88 。
客户端SDK 2.0	@alicloud/mpserverless-sdk@2.0.0	支持在微信小程序客户端中使用小程序云Serverless服务。 更多信息，请参见 在微信小程序中使用SDK 。
客户端SDK 1.0	@alicloud/mpserverless-sdk@1.0.3	通过小程序SDK可以直接在支付宝IDE中调用小程序云的云函数、云存储和数据库等服务。

11.2 安装客户端SDK2.2版本

通过小程序云SDK，您可以在小程序中直接访问小程序云Serverless服务。2.2版本开始支持统计分析功能。

前提条件

在首次使用小程序云服务前，您需要开通小程序云服务。详细信息，请参见[#unique_15](#)。

在支付宝小程序中使用SDK

完成以下操作，在支付宝小程序中使用SDK：

1. 在支付宝小程序项目的根目录执行以下命令安装SDK。

```
npm install @alicloud/mpserverless-sdk@2.2 --save
```

2. 在小程序项目中添加以下代码，初始化SDK。

```
import MPServerless from '@alicloud/mpserverless-sdk';

const mpServerless = new MPServerless({
  uploadFile: my.uploadFile,
  request: my.request,
  getAuthCode: my.getAuthCode,
```

```
getFileInfo: my.getFileInfo,  
getImageInfo: my.getImageInfo  
}, {  
  appId: '1234456789', // 小程序应用标识  
  spaceId: 'db4dd657-7041-470a-90xxxxx', // 服务空间标识  
  clientSecret: '6c3c86xxxx6', // 服务空间 secret key  
  endpoint: 'https://endpoint', // 服务空间地址, 从小程序Serverless控制台  
  处获得  
});
```

其中:

- appId是小程序的ID。您可以在[支付宝小程序开放平台](#)获取小程序的App ID。
- spaceId、clientSecret和endpoint在小程序Serverless控制台创建服务空间后可以获得。详情参见[创建服务空间](#)。

3. 调用user.authorize方法完成授权。

```
const res = await mpServerless.user.authorize({  
  authProvider: 'alipay_openapi'  
});  
if (res.success) {  
  console.log('授权成功');  
}
```

其中:

- authProvider: 指定授权方, 取值alipay_openapi和wechat_openapi。
- authType: 指定授权方式。取值anonymous表示通过匿名方式授权。



说明:

- 如果同时指定了authProvider和authType, authType配置优先。
- 如果将authType设置为anonymous, 则不需要设置authProvider。
- 如果没有设置authType, 则必须设置authProvider。
- 匿名授权获取的用户身份是一个临时身份, 每次匿名授权得到的用户标识是随机不固定的, 若需要获取固定用户标识, 则需要通过三方授权获取。

在微信小程序中使用SDK

完成以下操作, 在微信小程序中使用小程序云SDK:

1. 单击[这里](#)下载SDK文件。
2. 将下载后的mpserverless.js文件添加到小程序项目目录下, 然后在app.js或其他需要的页面中引入SDK文件。

```
//app.js  
const MPServerless = require('/sdk/mpserverless.js');
```

```
const mpServerless = new MPServerless({
  uploadFile: wx.uploadFile,
  request: wx.request,
  getAuthCode: wx.login,
  getFileInfo: wx.getFileInfo,
  getImageInfo: wx.getImageInfo,
}, {
  appId: 'appId', // 小程序应用标识
  spaceId: '', // 服务空间标识
  clientSecret: '', // 服务空间 secret key
  endpoint: '', // 服务空间地址, 从小程序 serverless 控制台处获得
});
```

其中:

- appId是小程序的ID。您可以在[微信公众平台](#)获取小程序的App ID。
- spaceId、clientSecret和endpoint在小程序Serverless控制台创建服务空间后可以获得。详情参见[创建服务空间](#)。

3. 调用user.authorize方法完成授权

```
const res = await mpServerless.user.authorize({
  authProvider: 'wechat_openapi'
});
if (res.success) {
  console.log('授权成功');
}
```

其中:

- authProvider: 指定授权方, 取值alipay_openapi和wechat_openapi。
- authType: 指定授权方式。取值anonymous表示通过匿名方式授权。



说明:

- 如果同时指定了authProvider和authType, authType配置优先。
- 如果将authType设置为anonymous, 则不需要设置authProvider。
- 如果没有设置authType, 则必须设置authProvider。
- 匿名授权获取的用户身份是一个临时身份, 每次匿名授权得到的用户标识是随机不固定的, 若需要获取固定用户标识, 则需要通过三方授权获取。

在钉钉小程序中使用SDK

完成以下操作, 在第三方个人应用的钉钉小程序中使用小程序云SDK:

1. 在钉钉小程序项目的根目录执行以下命令安装SDK。

```
npm install @alicloud/dt-mpserverless-sdk@0.0.2 --save
```

2. 在小程序项目中添加以下代码，初始化SDK。

```
import MPServerlessSDK from '@alicloud/dt-mpserverless-sdk';
const mpServerless = new MPServerlessSDK({
  uploadFile: dd.uploadFile,
  request: dd.httpRequest,
  getAuthCode: dd.getAuthCode
}, {
  appId: '1234456789', // 小程序应用标识
  spaceId: 'db4dd657-7041-470a-90xxxxx', // 服务空间标识
  clientSecret: '6c3c86xxxx6', // 服务空间 secret key
  endpoint: 'https://endpoint', // 服务空间地址，从小程序Serverless控制台处获得
})
```

其中：

- appId是小程序的ID。您可以在[钉钉开放平台](#)获取小程序的App ID。
- spaceId、clientSecret和endpoint在小程序Serverless控制台创建服务空间后可以获得。详情参见[创建服务空间](#)。

3. 调用user.authorize方法完成授权。

```
const res = await mpServerless.user.authorize({
  authProvider: 'dingtalk_openapi'
});
if (res.success) {
  console.log('授权成功');
}
```

其中：

- authProvider：指定授权方，钉钉SDK仅支持取值dingtalk_openapi。
- authType：指定授权方式。取值anonymous表示通过匿名方式授权。



说明：

- 如果同时指定了authProvider和authType，authType配置优先。
- 如果将authType设置为anonymous，则不需要设置authProvider。
- 如果没有设置authType，则必须设置authProvider。
- 匿名授权获取的用户身份是一个临时身份，每次匿名授权得到的用户标识是随机不固定的，若需要获取固定用户标识，则需要通过三方授权获取。

11.3 安装客户端SDK2.2版本

通过小程序云SDK，您可以在小程序中直接访问小程序云Serverless服务。2.2版本开始支持统计分析功能。

前提条件

在首次使用小程序云服务前，您需要开通小程序云服务。详细信息，请参见[#unique_15](#)。

在支付宝小程序中使用SDK

完成以下操作，在支付宝小程序中使用SDK：

1. 在支付宝小程序项目的根目录执行以下命令安装SDK。

```
npm install @alicloud/mpserverless-sdk@2.2 --save
```

2. 在小程序项目中添加以下代码，初始化SDK。

```
import MPServerless from '@alicloud/mpserverless-sdk';

const mpServerless = new MPServerless({
  uploadFile: my.uploadFile,
  request: my.request,
  getAuthCode: my.getAuthCode,
  getFileInfo: my.getFileInfo,
  getImageInfo: my.getImageInfo
}, {
  appId: '1234456789', // 小程序应用标识
  spaceId: 'db4dd657-7041-470a-90xxxxx', // 服务空间标识
  clientSecret: '6c3c86xxxx6', // 服务空间 secret key
  endpoint: 'https://endpoint', // 服务空间地址，从小程序Serverless控制台处获得
});
```

其中：

- appId是小程序的ID。您可以在[支付宝小程序开放平台](#)获取小程序的App ID。
- spaceId、clientSecret和endpoint在小程序Serverless控制台创建服务空间后可以获得。详情参见[创建服务空间](#)。

3. 调用user.authorize方法完成授权。

```
const res = await mpServerless.user.authorize({
  authProvider: 'alipay_openapi'
});
if (res.success) {
  console.log('授权成功');
```



```
}
```

其中：

- **authProvider**：指定授权方，取值`alipay_openapi`和`wechat_openapi`。
- **authType**：指定授权方式。取值`anonymous`表示通过匿名方式授权。



说明：

- 如果同时指定了`authProvider`和`authType`，`authType`配置优先。
- 如果将`authType`设置为`anonymous`，则不需要设置`authProvider`。
- 如果没有设置`authType`，则必须设置`authProvider`。
- 匿名授权获取的用户身份是一个临时身份，每次匿名授权得到的用户标识是随机不固定的，若需要获取固定用户标识，则需要通过三方授权获取。

在微信小程序中使用SDK

完成以下操作，在微信小程序中使用小程序云SDK：

1. 单击[这里](#)下载SDK文件。
2. 将下载后的`mpserverless.js`文件添加到小程序项目目录下，然后在`app.js`或其他需要的页面中引入SDK文件。

```
//app.js
const MPServerless = require('/sdk/mpserverless.js');

const mpServerless = new MPServerless({
  uploadFile: wx.uploadFile,
  request: wx.request,
  getAuthCode: wx.login,
  getFileInfo: wx.getFileInfo,
  getImageInfo: wx.getImageInfo,
}, {
  appId: 'appId', // 小程序应用标识
  spaceId: '', // 服务空间标识
  clientSecret: '', // 服务空间 secret key
  endpoint: '', // 服务空间地址，从小程序 serverless 控制台处获得
});
```

其中：

- `appId`是小程序的ID。您可以在[微信公众平台](#)获取小程序的App ID。
- `spaceId`、`clientSecret`和`endpoint`在小程序Serverless控制台创建服务空间后可以获得。详情参见[创建服务空间](#)。

3. 调用`user.authorize`方法完成授权

```
const res = await mpServerless.user.authorize({
  authProvider: 'wechat_openapi'
```

```
});  
if (res.success) {  
  console.log('授权成功');  
}
```

其中：

- authProvider：指定授权方，取值alipay_openapi和wechat_openapi。
- authType：指定授权方式。取值anonymous表示通过匿名方式授权。



说明：

- 如果同时指定了authProvider和authType，authType配置优先。
- 如果将authType设置为anonymous，则不需要设置authProvider。
- 如果没有设置authType，则必须设置authProvider。
- 匿名授权获取的用户身份是一个临时身份，每次匿名授权得到的用户标识是随机不固定的，若需要获取固定用户标识，则需要通过三方授权获取。

在钉钉小程序中使用SDK

完成以下操作，在第三方个人应用的钉钉小程序中使用小程序云SDK：

1. 在钉钉小程序项目的根目录执行以下命令安装SDK。

```
npm install @alicloud/dt-mpserverless-sdk@0.0.2 --save
```

2. 在小程序项目中添加以下代码，初始化SDK。

```
import MPServerlessSDK from '@alicloud/dt-mpserverless-sdk';  
const mpServerless = new MPServerlessSDK({  
  uploadFile: dd.uploadFile,  
  request: dd.httpRequest,  
  getAuthCode: dd.getAuthCode  
}, {  
  appId: '1234456789', // 小程序应用标识  
  spaceId: 'db4dd657-7041-470a-90xxxxxx', // 服务空间标识  
  clientSecret: '6c3c86xxxx6', // 服务空间 secret key  
  endpoint: 'https://endpoint', // 服务空间地址，从小程序Serverless控制台  
  处获得  
})
```

其中：

- appId是小程序的ID。您可以在[钉钉开放平台](#)获取小程序的App ID。
- spaceId、clientSecret和endpoint在小程序Serverless控制台创建服务空间后可以获得。详情参见[创建服务空间](#)。

3. 调用user.authorize方法完成授权。

```
const res = await mpServerless.user.authorize({
```

```
    authProvider: 'dingtalk_openapi'
  });
  if (res.success) {
    console.log('授权成功');
  }
}
```

其中：

- **authProvider**：指定授权方，钉钉SDK仅支持取值dingtalk_openapi。
- **authType**：指定授权方式。取值anonymous表示通过匿名方式授权。



说明：

- 如果同时指定了authProvider和authType，authType配置优先。
- 如果将authType设置为anonymous，则不需要设置authProvider。
- 如果没有设置authType，则必须设置authProvider。
- 匿名授权获取的用户身份是一个临时身份，每次匿名授权得到的用户标识是随机不固定的，若需要获取固定用户标识，则需要通过三方授权获取。

11.4 安装客户端SDK2.0版本

通过小程序云SDK，您可以在小程序中直接访问小程序云Serverless服务。2.0版本支持支付宝和微信小程序，在安装SDK后，您需要调用user.authorize进行授权。

前提条件

在首次使用小程序云服务前，您需要开通小程序云服务。详细信息，请参见[#unique_15](#)。

在支付宝小程序中使用SDK

完成以下操作，在支付宝小程序中使用SDK：

1. 在支付宝小程序项目的根目录执行以下命令安装SDK。

```
npm install @alicloud/mpserverless-sdk@2.0.0 --save
```

2. 在小程序项目中添加以下代码，初始化SDK。

```
import MPServerless from '@alicloud/mpserverless-sdk';

const mpServerless = new MPServerless({
  uploadFile: my.uploadFile,
  request: my.request,
  getAuthCode: my.getAuthCode
}, {
  appId: '1234456789', // 小程序应用标识
  spaceId: 'db4dd657-7041-470a-90xxxxx', // 服务空间标识
  clientSecret: '6c3c86xxxx6', // 服务空间 secret key
  endpoint: 'https://endpoint', // 服务空间地址，从小程序Serverless控制台获得
}
```

```
});
```

其中：

- appId是小程序的ID。您可以在[支付宝小程序开放平台](#)获取小程序的App ID。
- spaceIdclientSecret和endpoint在小程序Serverless控制台创建服务空间后可以获得。详情参见[创建服务空间](#)。

3. 调用user.authorize方法完成授权。

```
const res = await mpServerless.user.authorize({
  authProvider: 'alipay_openapi'
});
if (res.success) {
  console.log('授权成功');
}
```

在微信小程序中使用SDK

完成以下操作，在微信小程序中使用小程序云SDK：

1. 单击[这里](#)下载SDK文件。
2. 将下载后的mpserverless.js文件添加到小程序项目目录下，然后在app.js或其他需要的页面中引入SDK文件。

```
//app.js
const MPServerless = require('/sdk/mpserverless.js');

const mpServerless = new MPServerless({
  uploadFile: wx.uploadFile,
  request: wx.request,
  getAuthCode: wx.login,
  getFileInfo: wx.getFileInfo,
  getImageInfo: wx.getImageInfo,
}, {
  appId: 'appId', // 小程序应用标识
  spaceId: '', // 服务空间标识
  clientSecret: '', // 服务空间 secret key
  endpoint: '', // 服务空间地址，从小程序 serverless 控制台处获得
});
```

其中：

- appId是小程序的ID。您可以在[微信公众平台](#)获取小程序的App ID。
- spaceIdclientSecret和endpoint在小程序Serverless控制台创建服务空间后可以获得。详情参见[创建服务空间](#)。

3. 调用user.authorize方法完成授权

```
const res = await mpServerless.user.authorize({
  authProvider: 'wechat_openapi'
});
if (res.success) {
```

```
console.log('授权成功');
}
```

11.5 安装客户端SDK1.0版本

通过小程序云SDK，您可以在小程序中直接访问小程序云Serverless服务。

前提条件

在首次使用小程序云服务前，您需要开通小程序云服务。详细信息，请参见[#unique_15](#)。

操作步骤

1. 在小程序项目的根目录执行以下命令安装SDK。

```
npm install @alicloud/mpserverless-sdk@1.0.3 --save
```

2. 执行以下代码完成初始化。

```
import MPServerless from '@alicloud/mpserverless-sdk';
const mpServerless = new MPServerless({
  uploadFile: my.uploadFile,
  request: my.request,
  getAuthCode: my.getAuthCode
}, {
  appId: '1234456789', // 小程序应用标识
  spaceId: 'db4dd657-7041-470a-90xxxxx', // 服务空间标识
  clientSecret: '6c3c86xxxx6', // 服务空间 secret key
  endpoint: 'https://endpoint', // 服务空间地址，从小程序Serverless控制台处获得
});
```

其中：

- appId是小程序的ID。您可以在支付宝小程序控制台获得。
- spaceIdclientSecret和endpoint在小程序Serverless控制台创建服务空间后可以获得。详情参见[创建服务空间](#)。

12 客户端API参考

12.1 云函数

12.1.1 invoke

调用云服务。

方法定义

该方法的定义如下：

```
mpserverless.function.invoke(functionName: string, options?: object):  
Promise<Result>
```

返回参数

字段名	类型	说明
success	Boolean	执行状态。
requestId	String	请求ID。
result	Any	接口返回内容，由开发者代码和请求参数 header 的 content-type 决定，默认 content-type 为 application/json。

示例

```
mpserverless.function.invoke('dataAnalytics', {  
  range: 30,  
}).then((res) => {  
  console.log(res);  
}).catch(console.error);
```

12.2 数据存储

12.2.1 aggregate

对数据库执行聚合查询。

方法定义

该方法的定义如下：

```
aggregate(pipeline: object | object[], options?: object): Promise<MongoResult>
```

请求参数

该方法接收11个参数，其定义如下：

字段名	类型	必填	说明
pipeline	Array	是	聚合查询对象。
options	Object	否	控制项。
options. explain	Boolean	否	是否返回执行计划。
options. allowDiskUse	Boolean	否	是否在聚合查询执行的过程中使用磁盘存储临时数据。
options. maxTimeMS	Number	否	执行查询的超时时间。
options. bypassDocumentValidation	Boolean	否	是否允许绕过文档验证。
options. raw	Boolean	否	是否将结果作为 BSON Buffer 返回。
options. promoteLongs	Boolean	否	是否将 long 数据类型提升为 Number。
options. promoteValues	Boolean	否	是否将 BSON 值的类型提升为本地类型。
options. promoteBuffers	Boolean	否	是否将 BSON Buffer 转换为 Node Buffer。

字段名	类型	必填	说明
options.collation	Object	否	指定更新的排序顺序。

示例

```
mpserverless.db.collection('users').aggregate([
  {
    $match: {
      _id: {
        $in: insertRet.insertedIds
      }
    },
  },
  {
    $project: {
      name: 1,
      address: 1
    }
  },
  {
    $group: {
      _id: '$name',
      count: {$sum: 1}
    }
  },
  {
    $match: {
      count: {
        $gt: 1
      }
    }
  }
]);
```

12.2.2 count

获取集合中符合条件的记录数量。

方法定义

该方法的定义如下：

```
count(query: object, options?: object): Promise<MongoResult>
```

请求参数

该方法接收 5 个参数，其定义如下：

字段名	类型	必填	说明
query	Object	否	数据库操作时的过滤条件。
options	String	否	控制项。

字段名	类型	必填	说明
options.limit	Boolean	否	是否限制 count 的文档数量。
options.skip	Number	否	count 前跳过的文档数量。
options.maxTimeMS	Number	否	超时时间。

示例

```
mpserverless.db.collection('users').count({age: {$gt: 18}}).then((res) => {
  console.log(res);
}).catch(console.error);
```

12.2.3 deleteOne

删除集合中的一条记录。如果没有查询条件，则默认删除第一行。

方法定义

该方法的定义如下：

```
deleteOne(filter: object): Promise<MongoResult>
```

请求参数

该方法接收一个参数，其定义如下：

字段名	类型	必填	说明
filter	Object	否	数据库操作时的过滤条件

示例

```
mpserverless.db.collection('users').deleteOne({age: {$lt: 18}}).then((res) => {
  const hasDeleted = res.affectedDocs > 0;
```

```
}).catch(console.error);
```

12.2.4 deleteMany

删除集合中的一批记录。

方法定义

该方法的定义如下：

```
deleteMany(filter: object): Promise<MongoResult>;
```

请求参数

字段名	类型	必填	说明
filter	Object	否	数据库操作时的过滤条件。

示例

```
mpserverless.db.collection('users').deleteMany({age: {$lt: 18}}).then((res) => {
  const hasDeleted = res.affectedDocs > 0;
}).catch(console.error);
```

12.2.5 distinct

获取某个属性去重后的所有记录。

方法定义

该方法的定义如下：

```
distinct(key: string, query: object, options?: object): Promise<MongoResult>
```

请求参数

该方法接收 3 个参数，其定义如下：

字段名	类型	必填	说明
key	String	是	待获取的属性名。
query	Object	是	数据库操作时的查询条件。
options	Object	否	控制项。

示例

```
mpserverless.db.collection('users').distinct('name', {age: {$gt: 18}})
```

```
.then((res) => {})  
.catch(console.error);
```

12.2.6 find

查找集合中符合条件的所有记录。

方法定义

该方法的定义如下：

```
find(query?: object, options?: object): Promise<MongoResult>
```

请求参数

该方法接收 8 个参数，其定义如下：

字段名	类型	必填	说明
query	Object	否	数据库操作时的查询条件。
options	Object	否	控制项。
options.limit	Number	否	查询的文档数量限制。
options.skip	Number	否	跳过的文档数量。
options.maxTimeMS	Number	否	超时时间。
options.sort	Object	否	指定排序的字段，并使用 1 和 -1 来指定排序的方式。其中： · 1：表示升序排列 · -1：表示降序排列
options.projection	Object	否	使用投影操作符指定返回的键，值设置为1的字段返回，值为0的字段隐藏。
options.hint	Object	否	指定查询时使用的索引。

示例

```
mpserverless.db.collection('users').find({  
  age: {$gt: 18}  
}, {  
  projection: {name: 1},  
  limit: 10,  
  skip: 10,  
  sort: {name: 1},  
})
```

```
    hint: {name: 1, _id: 0}
  })
  .then(res => {})
  .catch(console.error);
```

12.2.7 findOne

查询单条记录。

方法定义

该方法的定义如下：

```
findOne(query?: object, options?: object): Promise<MongoResult>
```

请求参数

该方法接收 8 个参数，其定义如下：

字段名	类型	必填	说明
query	Object	否	数据库操作时的查询条件。
options	String	否	控制项。
options.limit	Number	否	查询的文档数量限制。
options.skip	Number	否	跳过的文档数量。
options.maxTimeMS	Number	否	超时时间。
options.sort	Object	否	排序规则。
options.projection	Object	否	查询后过滤的字段。
options.hint	Object	否	指定查询时使用的索引。

示例

```
mpserverless.db.collection('users')
  .findOne({
    age: {$gt: 18}
  }, {
    projection: {name: 1},
    limit: 10,
    skip: 10,
  })
  .then(res => {})
```

```
.catch(console.error)
```

12.2.8 findOneAndDelete

查询并删除一条记录。

方法定义

该方法的定义如下：

```
findOneAndDelete(query?: object, options?: object): Promise<MongoResult>
```

请求参数

该方法接收 5 个参数，其定义如下：

字段	类型	必填	说明
query	Object	否	数据库操作时的查询条件。
options	String	否	控制项。
options.maxTimeMS	Number	否	超时时间。
options.sort	Object	否	排序规则。
options.projection	Object	否	查询后过滤的字段。

示例

```
mpserverless.db.collection('users')
  .findOneAndDelete({
    age: 18
  })
  .then(res => {})
  .catch(console.error)
```

12.2.9 findOneAndReplace

查询并整体替换一条记录。

方法定义

该方法的定义如下：

```
findOneAndReplace(query?: object, options?: object): Promise<MongoResult>
```

请求参数

该方法接收 6 个参数，其定义如下：

字段	类型	必填	描述
query	Object	否	数据库操作时的查询条件。
options	String	否	控制项。
options.maxTimeMS	Number	否	超时时间。
options.sort	Object	否	排序规则
options.upsert	Boolean	否	如果查找不到对应文档，是否插入。默认值：false。
options.projection	Object	否	查询后过滤的字段。

示例

```
mpserverless.db.collection('users')
  .findOneAndReplace({
    age: 18
  })
  .then(res => {})
  .catch(console.error)
```

12.2.10 findOneAndUpdate

查询并更新记录。

方法定义

该方法的定义如下：

```
findOneAndUpdate(filter: object, update: object, options?: object):
Promise<MongoResult>
```

请求参数

该方法接收 7 个参数，其定义如下：

字段	类型	必填	说明
filter	Object	是	数据库操作时的过滤条件。
update	Object	是	数据库操作时的更新对象。
options	String	否	控制项。
options.maxTimeMS	Number	否	超时时间。

字段	类型	必填	说明
options.sort	Object	否	排序规则。
options.upsert	Boolean	否	如果查找不到对应文档，是否插入。默认值：false。
options.projection	Object	否	查询后过滤的字段。

示例

```
mpserverless.db.collection('users')
  .findOneAndUpdate({
    username: "zhangsan"
  },
  {
    $set: {
      age: 18
    }
  })
  .then(res => {})
  .catch(console.error)
```

12.2.11 replaceOne

查询并整体替换这条记录。

方法定义

该方法的定义如下：

```
replaceOne(doc: object, filter: object, options?: object): Promise<MongoResult>
```

请求参数

该方法接收 4 个参数，其定义如下：

字段名	类型	必填	说明
filter	Object	是	数据库操作时的过滤条件。
doc	Object	是	替换的目标文档。
options	Object	否	控制项。
options.upsert	Boolean	否	如果匹配不到，是否插入。默认值：false。

示例

```
mpserverless.db.collection('users')
  .replaceOne({
    name: 'tom'
    age: 18
  }, {
    name: 'jerry'
  })
  .then(res => {})
  .catch(console.error);
```

12.2.12 insertMany

在集合中添加一批记录。

方法定义

该方法的定义如下：

```
insertMany(docs: object[]): Promise<MongoResult>
```

请求参数

该方法接收 1 个参数，其定义如下：

字段名	类型	必填	说明
docs	Array	是	待插入的数据，只能为数组。

示例

```
mpserverless.db.collection('users').insertMany([
  {
    name: 'tom',
    age: 1
  }, {
    name: 'jerry',
    age: 2
  })
  .then(res => {})
```



```
.catch(console.error)
```

12.2.13 insertOne

在集合中添加一条记录。

方法定义

该方法的定义如下：

```
insertOne(doc: object): Promise<MongoResult>
```

请求参数

该方法接收 1 个参数，其定义如下：

字段名	类型	必填	说明
doc	Object	是	待插入的数据

示例

```
mpserverless.db.collection('users').insertOne({  
  name: 'tom',  
  age: 1  
})  
.then(res => {})  
.catch(console.error)
```

12.2.14 updateMany

更新集合中的一批记录。

方法定义

该方法的定义如下：

```
updateMany(filter: object, update: object, options?: object): Promise<MongoResult>
```

请求参数

该方法接收 4 个参数，其定义如下：

字段名	类型	必填	说明
filter	Object	是	过滤条件。
update	Object	是	更新的文档
options	Object	否	控制项

字段名	类型	必填	说明
options.upsert	Boolean	否	不匹配时是否直接插入文档。默认值：false。

示例

```
mpserverless.db.collection('users').updateMany({
  name: 'jerry'
}, {
  $set: {
    age: 10
  }
})
.then(res => {})
.catch(console.error)
```

12.2.15 updateOne

更新集合中的一条记录。

方法定义

该方法的定义如下：

```
updateOne(filter: object, update: object, options?: object): Promise<
MongoResult>
```

请求参数

该方法接收 4 个参数，其定义如下：

字段名	类型	必填	说明
filter	Object	是	过滤条件。
update	Object	是	更新的文档。
options	Object	否	控制项。
options.upsert	Boolean	否	不匹配时是否直接插入文档。默认值：false。

示例

```
mpserverless.db.collection('users').updateOne({
  name: 'jerry'
}, {
  $set: {
    age: 10
  }
})
```

```
.then(res => {})  
.catch(console.error)
```

12.3 文件存储

12.3.1 uploadFile

上传文件。

方法定义

该方法的定义如下：

```
mpserverless.file.uploadFile(options: object): Promise<Result>
```

请求参数

该方法接收 6 个参数，其定义如下：

字段名	类型	必填	说明
options	Object	是	上传控制项。
options. fileSize	String	否	上传的文件大小。
options. extension	String	否	上传的文件的扩展名。 目前支持上传以下格式的文件： · .jpg · .jpeg · .png · .gif · .image
options. filePath	String	是	本地文件路径，通常可以从图片文件描述中获取文件路径。可上传文件大小限制在 100 MB 以内。
options. env	String	否	文件的获取方式。 唯一可选值 public：可公开访问的文件。
options. timeout	Number	否	超时时间，以毫秒为单位。默认值：60000。

字段名	类型	必填	说明
options.headers	Object	否	文件响应头键值对，可定义如下内容： • cacheControl：缓存策略。 • contentDisposition：文件形式。 • contentEncoding：文件内容编码。 • expires：缓存有效时长。
options.meta	Object	否	自定义文件响应头键值对。例如，自定义 userId: halo 获得响应头 x-meta-user-id: halo。

返回参数

字段名	类型	说明
filePath	String	本地文件路径。
fileUrl	String	上传文件后获得的文件链接地址。

示例

小程序选取文件上传示例。

```
my.chooseImage({
  chooseImage: 1,
  success: res => {
    const path = res.apFilePaths[0];
    const options = {
      filePath: path,
      headers: {
        contentDisposition: 'attachment',
      },
    };

    mpserverless.file.uploadFile(options).then(console.log).catch(
      console.error);
  },
});
```

12.3.2 deleteFile

删除之前上传的文件。

方法定义

该方法的定义如下：

```
mpserverless.file.deleteFile(url: string): Promise<Result>
```

请求参数

该方法接收 1 个参数，其定义如下：

字段名	类型	必填	说明
url	String	是	线上完整路径

示例

```
mpserverless.file.deleteFile('https://resource.bspapp.com/xxx-xx/4b82ded0-0118-4de4-9f50-ab13110a1ffb.jpg').then(res => {
  console.log(res);
}).catch(err => {
  console.error(err);
});
```

12.4 用户管理

12.4.1 getInfo

获取用户信息。

方法定义

该方法的定义如下：

```
mpserverless.user.getInfo(options: object): Promise<Result>
```

请求参数

该方法接收以下请求参数。

字段名	类型	必填	说明
options	Object	否	authProvider{String}指定客户端类型，取值： • alipay_openapi：支付宝（默认值） • wechat_openapi：微信 • dingtalk_openapi：钉钉

返回参数

字段名	类型	说明
success	Boolean	操作是否成功。
result	Object	用户信息： • spaceId{String}：服务空间ID。 • userId{String}：用户ID。 • oAuthUserId{String}：三方授权用户标识。只有指定了authProvider时才会返回该参数。

示例

- 匿名授权方式获取用户信息。

```
const res = await mpServerless.user.authorize({
  authType: 'anonymous'
});
if (res.success) {
  console.log('授权成功');
}
mpServerless.user.getInfo().then(user => {
  this.setData(userInfo: user);
}).catch(console.error);
```

- 非匿名授权方式获取用户信息。

- 支付宝授权:

```
const res = await app.mpServerless.user.authorize({
  authProvider: 'alipay_openapi'
});
if (res.success) {
  console.log('授权成功');
}
mpServerless.user.getInfo().then(user => {
  this.setData(userInfo: user);
}).catch(console.error);
```

- 微信授权:

```
const res = await app.mpServerless.user.authorize({
  authProvider: 'wechat_openapi'
});
if (res.success) {
  console.log('授权成功');
}
mpServerless.user.getInfo().then(user => {
  this.setData(userInfo: user);
}).catch(console.error);
```

- 钉钉授权:

```
const res = await app.mpServerless.user.authorize({
  authProvider: 'dingtalk_openapi'
});
if (res.success) {
  console.log('授权成功');
}
mpServerless.user.getInfo().then(user => {
  this.setData(userInfo: user);
}).catch(console.error);
```



说明:

应用获取授权后方能获取用户信息，获取授权接口信息参见[authorize](#)。

12.4.2 authorize

获取用户授权。

方法定义

该方法的定义如下：

```
authorize(options: AuthorizeOptions): Promise<{success: boolean}>
```

请求参数

字段名	类型	必填	说明
options	Object	否	- authProvider{String}：客户端类型。取值： - alipay_opennapi（默认值） - wechat_opennapi - authType{String}：授权方式。取值： - 空字符串（默认值）。 - anonymous：匿名方式



说明：

- 如果同时指定了authProvider和authType，authType配置优先。
- 如果将authType设置为anonymous，则不需要设置authProvider。
- 如果没有设置authType，则必须设置authProvider。

返回参数

字段名	类型	说明
success	Boolean	操作是否成功。

示例

• 非匿名授权：

```
const res = await mpServerless.user.authorize({
  authProvider: 'alipay_openapi'
});
if (res.success) {
  console.log('授权成功');
}
```

• 匿名授权：

```
const res = await mpServerless.user.authorize({
  authType: 'anonymous'
});
if (res.success) {
  console.log('授权成功');
}
```



说明：

匿名授权获取的用户身份是一个临时身份，每次匿名授权得到的用户标识是随机不固定的，若需要获取固定用户标识，则需要通过三方授权获取。

13 钉钉隐藏暂时
