短信

SDK 文档

产品文档

腾讯云

文档目录

SDK 文档
SDK 下载

最近更新时间：2019-09-23 11:55:00

目前短信提供国内短信和国际/港澳台短信服务。

- 国内短信提供单发，群发，指定模板单发，指定模板群发以及短信回执与回复拉取。
- 国际/港澳台短信可以直接使用国内单发和群发接口，只需替换相应的国家（或地区）码与手机号码。

SDK 名称	SDK 文档	github 地址
Java SDK	短信 Java SDK 文档	短信 Java SDK
PHP SDK	短信 PHP SDK 文档	短信 PHP SDK
Python SDK	短信 Python SDK 文档	短信 Python SDK
JavaScript SDK	短信 Node.js SDK 文档	短信 Node.js SDK
C# SDK	短信 C# SDK 文档	短信 C# SDK
小程序	短信小程序开发指南	云短信体验版 Demo

Java SDK

最近更新时间：2019-10-29 16:15:02

SDK 功能简介

目前腾讯云短信为客户提供**国内短信**和**国际/港澳台短信**服务，腾讯云短信 SDK 支持以下操作：

国内短信	国际/港澳台短信
• 指定模板单发短信 • 指定模板群发短信 • 拉取短信回执和短信回复状态	• 指定模板单发短信 • 指定模板群发短信 • 拉取短信回执

说明：

- 群发短信
  一次群发请求最多支持200个号码，如对号码数量有特殊需求请联系腾讯云短信技术支持（QQ：3012203387）。
- 拉取短信回执
  该功能默认关闭。您可以根据实际需求联系腾讯云短信技术支持（QQ：3012203387）开通，实现批量拉取短信回执。

SDK 使用指南

相关资料

各个接口及其参数的详情介绍请参见 API 指南 、SDK 文档 和 错误码。

前提条件

在使用 SDK 前，您需要准备以下信息：

- **获取 SDKAppID 和 AppKey**
  云短信应用 **SDKAppID** 和 **AppKey** 可在 短信控制台 的应用信息里获取。如您尚未添加应用，请登录 短信控制台 添加应用。
- **申请签名并确认审核通过**
  一个完整的短信由短信**签名**和**短信正文内容**两部分组成，短信**签名**需申请和审核，**签名**可在 短信控制台 的相应服务模块【内容配置】中进行申请，详细申请操作请参见 创建签名。发送国际/港澳台短信时，允许不携带签名。

- **申请模板并确认审核通过**
  短信正文内容**模板**需申请和审核，**模板**可在 短信控制台 的相应服务模块【内容配置】中进行申请，详细申请操作请参见 创建正文模板。

配置 SDK

qcloudsms_java 可以采用多种方式进行安装，我们提供以下三种方法供用户使用：

- **maven**
  使用 qcloudsms_java 功能前，需要先在 pom.xml 中添加如下依赖：

```
<dependency>
<groupId>com.github.qcloudsms</groupId>
<artifactId>qcloudsms</artifactId>
<version>1.0.6</version>
</dependency>
```

- **sbt**

```
libraryDependencies += "com.github.qcloudsms" % "sms" % "1.0.6"
```

- **其他**
  - 方法1
    将 源代码 直接引入到项目工程中。

    > 说明：
    > 由于 qcloudsms_java 依赖四个依赖项目 library： org.json，httpclient，httpcore 和 httpmime。若采用方法1，需要将以上四个 jar 包一并导入工程。

  - 方法2
    将 JAR 包 直接引入到您的工程中。

示例代码

> 说明：
> 所有示例代码仅作参考，无法直接编译和运行，需根据实际情况进行修改。

- **准备必要参数**

```
// 短信应用 SDK AppID
int appid = 1400009099; // SDK AppID 以1400开头
// 短信应用 SDK AppKey
String appkey = "9ff91d87c2cd7cd0ea762f141975d1df37481d48700d70ac37470aefc60f9bad";
// 需要发送短信的手机号码
String[] phoneNumbers = {"21212313123", "12345678902", "12345678903"};
// 短信模板 ID，需要在短信应用中申请
int templateId = 7839; // NOTE: 这里的模板 ID`7839`只是示例，真实的模板 ID 需要在短信控制台中申请
// 签名
String smsSign = "腾讯云"; // NOTE: 签名参数使用的是`签名内容`，而不是`签名ID`。这里的签名"腾讯云"只是示例，真实的签名需要在短信控制台申请
```

- **指定模板 ID 单发短信**

```
import com.github.qcloudsms.SmsSingleSender;
import com.github.qcloudsms.SmsSingleSenderResult;
import com.github.qcloudsms.httpclient.HTTPException;
import org.json.JSONException;
import java.io.IOException;
try {
String[] params = {"5678"};
SmsSingleSender ssender = new SmsSingleSender(appid, appkey);
SmsSingleSenderResult result = ssender.sendWithParam("86", phoneNumbers[0],
templateId, params, smsSign, "", "");
System.out.println(result);
} catch (HTTPException e) {
// HTTP 响应码错误
e.printStackTrace();
} catch (JSONException e) {
// JSON 解析错误
e.printStackTrace();
} catch (IOException e) {
// 网络 IO 错误
e.printStackTrace();
}
```

- **指定模板 ID 群发短信**

```
import com.github.qcloudsms.SmsMultiSender;
import com.github.qcloudsms.SmsMultiSenderResult;
```

```
import com.github.qcloudsms.httpclient.HTTPException;
import org.json.JSONException;
import java.io.IOException;
try {
String[] params = {"5678"};
SmsMultiSender msender = new SmsMultiSender(appid, appkey);
SmsMultiSenderResult result = msender.sendWithParam("86", phoneNumbers,
templateId, params, smsSign, "", "");
System.out.println(result);
} catch (HTTPException e) {
// HTTP 响应码错误
e.printStackTrace();
} catch (JSONException e) {
// JSON 解析错误
e.printStackTrace();
} catch (IOException e) {
// 网络 IO 错误
e.printStackTrace();
}
```

- **拉取短信回执以及回复**

```
import com.github.qcloudsms.SmsStatusPuller;
import com.github.qcloudsms.SmsStatusPullCallbackResult;
import com.github.qcloudsms.SmsStatusPullReplyResult;
import com.github.qcloudsms.httpclient.HTTPException;
import org.json.JSONException;
import java.io.IOException;
try {
// Note: 短信拉取功能需要联系腾讯云短信技术支持（QQ：3012203387）开通权限
int maxNum = 10; // 单次拉取最大量
SmsStatusPuller spuller = new SmsStatusPuller(appid, appkey);

// 拉取短信回执
SmsStatusPullCallbackResult callbackResult = spuller.pullCallback(maxNum);
System.out.println(callbackResult);

// 拉取回复，国际/港澳台短信不支持回复功能
SmsStatusPullReplyResult replyResult = spuller.pullReply(maxNum);
System.out.println(replyResult);
} catch (HTTPException e) {
// HTTP 响应码错误
e.printStackTrace();
} catch (JSONException e) {
```

```
// JSON 解析错误
e.printStackTrace();
} catch (IOException e) {
// 网络 IO 错误
e.printStackTrace();
}
```

- **拉取单个手机短信状态**

```
import com.github.qcloudsms.SmsMobileStatusPuller;
import com.github.qcloudsms.SmsStatusPullCallbackResult;
import com.github.qcloudsms.SmsStatusPullReplyResult;
import com.github.qcloudsms.httpclient.HTTPException;
import org.json.JSONException;
import java.io.IOException;
try {
int beginTime = 1511125600; // 开始时间（UNIX timestamp）
int endTime = 1511841600; // 结束时间（UNIX timestamp）
int maxNum = 10; // 单次拉取最大量
SmsMobileStatusPuller mspuller = new SmsMobileStatusPuller(appid, appkey);

// 拉取短信回执
SmsStatusPullCallbackResult callbackResult = mspuller.pullCallback("86",
phoneNumbers[0], beginTime, endTime, maxNum);
System.out.println(callbackResult);

// 拉取回复，国际/港澳台短信不支持回复功能
SmsStatusPullReplyResult replyResult = mspuller.pullReply("86",
phoneNumbers[0], beginTime, endTime, maxNum);
System.out.println(replyResult);
} catch (HTTPException e) {
// HTTP 响应码错误
e.printStackTrace();
} catch (JSONException e) {
// JSON 解析错误
e.printStackTrace();
} catch (IOException e) {
// 网络 IO 错误
e.printStackTrace();
}
```

- **发送国际/港澳台短信**

  发送国际/港澳台短信与发送国内短信类似，只需替换相应的国家码或地区码。详细示例请参考：

- 指定模板单发短信
- 指定模板群发短信
- 拉取短信回执

- **使用代理**

  部分环境需要使用代理才能上网，可使用 ProxyHTTPClient 来发送请求，示例如下：

```
import com.github.qcloudsms.SmsSingleSender;
import com.github.qcloudsms.SmsSingleSenderResult;
import com.github.qcloudsms.httpclient.HTTPException;
import com.github.qcloudsms.httpclient.ProxyHTTPClient;
import org.json.JSONException;
import java.io.IOException;
try {
// 创建一个代理 httpclient
ProxyHTTPClient httpclient = new ProxyHTTPClient("127.0.0.1", 8080, "http");

String[] params = {"5678"};
SmsSingleSender ssender = new SmsSingleSender(appid, appkey, httpclient);
SmsSingleSenderResult result = ssender.sendWithParam("86", phoneNumbers[0],
templateId, params, smsSign, "", ""); // 签名参数未提供或者为空时，会使用默认签名发送短信
System.out.println(result);
} catch (HTTPException e) {
// HTTP 响应码错误
e.printStackTrace();
} catch (JSONException e) {
// JSON 解析错误
e.printStackTrace();
} catch (IOException e) {
// 网络 IO 错误
e.printStackTrace();
}
```

- **使用连接池**

  多个线程可以共用一个连接池发送 API 请求，多线程并发单发短信示例如下：

```
import com.github.qcloudsms.SmsSingleSender;
import com.github.qcloudsms.SmsSingleSenderResult;
import com.github.qcloudsms.httpclient.HTTPException;
import com.github.qcloudsms.httpclient.PoolingHTTPClient;
import org.json.JSONException;
import java.io.IOException;
```

```
class SmsThread extends Thread {

private final SmsSingleSender sender;
private final String nationCode;
private final String phoneNumber;
private final String msg;

public SmsThread(SmsSingleSender sender, String nationCode, String phoneNumber, String msg)
{
this.sender = sender;
this.nationCode = nationCode;
this.phoneNumber = phoneNumber;
this.msg = msg;
}

@Override
public void run() {
try {
SmsSingleSenderResult result = sender.send(0, nationCode, phoneNumber, msg, "", "");
System.out.println(result);
} catch (HTTPException e) {
// HTTP 响应码错误
e.printStackTrace();
} catch (JSONException e) {
// JSON 解析错误
e.printStackTrace();
} catch (IOException e) {
// 网络 IO 错误
e.printStackTrace();
}
}
}
public class SmsTest {

public static void main(String[] args) {

int appid = 122333333;
String appkey = "9ff91d87c2cd7cd0ea762f141975d1df37481d48700d70ac37470aefc60f9bad";
String[] phoneNumbers = {
"21212313123", "12345678902", "12345678903",
"21212313124", "12345678903", "12345678904",
"21212313125", "12345678904", "12345678905",
"21212313126", "12345678905", "12345678906",
"21212313127", "12345678906", "12345678907",
};
```

```
// 创建一个连接池 httpclient, 并设置最大连接量为10
PoolingHTTPClient httpclient = new PoolingHTTPClient(10);

// 创建 SmsSingleSender 时传入连接池 http client
SmsSingleSender ssender = new SmsSingleSender(appid, appkey, httpclient);

// 创建线程
SmsThread[] threads = new SmsThread[phoneNumbers.length];
for (int i = 0; i < phoneNumbers.length; i++) {
threads[i] = new SmsThread(ssender, "86", phoneNumbers[i], "您验证码是：5678");
}

// 运行线程
for (int i = 0; i < threads.length; i++) {
threads[i].start();
}

// join 线程
for (int i = 0; i < threads.length; i++) {
threads[i].join();
}

// 关闭连接池 httpclient
httpclient.close();
}
}
```

- **使用自定义 HTTP client 实现**
  如果需要使用自定义的 HTTP client 实现，只需实现 `com.github.qcloudsms.httpclient.HTTPClient` 接口，并在构造 API 对象时传入自定义 HTTP client 即可，参考示例如下：

```
import com.github.qcloudsms.httpclient.HTTPClient;
import com.github.qcloudsms.httpclient.HTTPRequest;
import com.github.qcloudsms.httpclient.HTTPResponse;
import java.io.IOException;
import java.net.URISyntaxException;
// import com.example.httpclient.MyHTTPClient
// import com.exmaple.httpclient.MyHTTPRequest
// import com.example.httpclient.MyHTTPresponse
public class CustomHTTPClient implements HTTPClient {

public HTTPResponse fetch(HTTPRequest request) throws IOException, URISyntaxException {
// 1. 创建自定义 HTTP request
```

```
// MyHTTPrequest req = MyHTTPRequest.build(request)

// 2. 创建自定义 HTTP cleint
// MyHTTPClient client = new MyHTTPClient();

// 3. 使用自定义 HTTP client 获取 HTTP 响应
// MyHTTPResponse response = client.fetch(req);

// 4. 转换 HTTP 响应到 HTTPResponse
// HTTPResponse res = transformToHTTPResponse(response);

// 5. 返回 HTTPResponse 实例
// return res;
}

public void close() {
// 如果需要关闭必要资源
}
}
// 创建自定义 HTTP client
CustomHTTPClient httpclient = new CustomHTTPClient();
// 构造 API 对象时传入自定义 HTTP client
SmsSingleSender ssender = new SmsSingleSender(appid, appkey, httpclient);
```

PHP SDK

最近更新时间：2019-09-23 11:55:24

SDK 功能简介

目前腾讯云短信为客户提供**国内短信**和**国际/港澳台短信**服务，腾讯云短信 SDK 支持以下操作：

国内短信	国际/港澳台短信
• 指定模板单发短信 • 指定模板群发短信 • 拉取短信回执和短信回复状态	• 指定模板单发短信 • 指定模板群发短信 • 拉取短信回执

说明：

- 群发短信
  一次群发请求最多支持200个号码，如对号码数量有特殊需求请联系腾讯云短信技术支持（QQ：3012203387）。
- 拉取短信回执
  该功能默认关闭。您可以根据实际需求联系腾讯云短信技术支持（QQ：3012203387）开通，实现批量拉取短信回执。

SDK 使用指南

相关资料

各个接口及其参数的详情介绍请参见 API 指南 、SDK 文档 和 错误码。

前提条件

在使用 SDK 前，您需要准备以下信息：

- **获取 SDKAppID 和 AppKey**
  云短信应用 **SDKAppID** 和 **AppKey** 可在 短信控制台 的应用信息里获取。如您尚未添加应用，请登录 短信控制台 添加应用。
- **申请签名并确认审核通过**
  一个完整的短信由短信**签名**和**短信正文内容**两部分组成，短信**签名**需申请和审核，**签名**可在 短信控制台 的相应服务模块【内容配置】中进行申请，详细申请操作请参见 创建签名。发送国际/港澳台短信时，允许不携带签名。

- **申请模板并确认审核通过**
  短信正文内容**模板**需申请和审核，**模板**可在 短信控制台 的相应服务模块【内容配置】中进行申请，详细申请操作请参见 创建正文模板。

配置 SDK

- **Composer 配置：**
  qcloudsms_php 采用 composer 进行安装，要使用 qcloudsms 功能，只需要在 composer.json 中添加如下依赖：

```
{
"require": {
"qcloudsms/qcloudsms_php": "0.1.*"
}
}
```

> 说明：
> Composer 的使用可以参考 demo 目录下面的示例。

- **手动配置：**

i. 手动下载或 clone 最新版本 qcloudsms_php 代码。
ii. 把 qcloudsms_php src 目录下的代码放入 Autoloading 目录。
iii. 引入 require qcloudsms_php src 目录下面的 index.php 即可使用，如把 qcloudsms 放在当前目录下，只需要执行以下命令：

```
require __DIR__ . "/qcloudsms_php/src/index.php";
```

示例代码

> 说明：
> 所有示例代码仅作参考，无法直接编译和运行，需根据实际情况进行修改。

- **准备必要参数**

```
// 短信应用 SDK AppID
$appid = 1400009099; // SDK AppID 以1400开头
```

```
// 短信应用 SDK AppKey
$appkey = "9ff91d87c2cd7cd0ea762f141975d1df37481d48700d70ac37470aefc60f9bad";
// 需要发送短信的手机号码
$phoneNumbers = ["21212313123", "12345678902", "12345678903"];
// 短信模板 ID，需要在短信控制台中申请
$templateId = 7839; // NOTE: 这里的模板 ID`7839`只是示例，真实的模板 ID 需要在短信控制台中申请
$smsSign = "腾讯云"; // NOTE: 签名参数使用的是`签名内容`，而不是`签名ID`。这里的签名"腾讯云"只是示例，真实的签名需要在短信控制台申请
```

- **指定模板 ID 单发短信**

```
use Qcloud\Sms\SmsSingleSender;
try {
$ssender = new SmsSingleSender($appid, $appkey);
$params = ["5678"];
$result = $ssender->sendWithParam("86", $phoneNumbers[0], $templateId,
$params, $smsSign, "", "");
$rsp = json_decode($result);
echo $result;
} catch(\Exception $e) {
echo var_dump($e);
}
```

- **指定模板 ID 群发短信**

```
use Qcloud\Sms\SmsMultiSender;
try {
$msender = new SmsMultiSender($appid, $appkey);
$params = ["5678"];
$result = $msender->sendWithParam("86", $phoneNumbers,
$templateId, $params, $smsSign, "", "");
$rsp = json_decode($result);
echo $result;
} catch(\Exception $e) {
echo var_dump($e);
}
```

- **拉取短信回执以及回复**

```
use Qcloud\Sms\SmsStatusPuller;
try {
$spuller = new SmsStatusPuller($appid, $appkey);
```

```
// 拉取短信回执
$callbackResult = $spuller->pullCallback(10);
$callbackRsp = json_decode($callbackResult);
echo $callbackResult;

// 拉取回复，国际/港澳台短信不支持回复功能
$replyResult = $spuller->pullReply(10);
$replyRsp = json_decode($replyResult);
echo $replyResult;
} catch (\Exception $e) {
echo var_dump($e);
}
```

- **拉取单个手机短信状态**

```
use Qcloud\Sms\SmsMobileStatusPuller;
try {
$beginTime = 1511125600; // 开始时间（UNIX timestamp）
$endTime = 1511841600; // 结束时间（UNIX timestamp）
$maxNum = 10; // 单次拉取最大量
$mspuller = new SmsMobileStatusPuller($appid, $appkey);

// 拉取短信回执
$callbackResult = $mspuller->pullCallback("86", $phoneNumbers[0],
$beginTime, $endTime, $maxNum);
$callbackRsp = json_decode($callbackResult);
echo $callbackResult;

// 拉取回复，国际/港澳台短信不支持回复功能
$replyResult = $mspuller->pullReply("86", $phoneNumbers[0],
$beginTime, $endTime, $maxNum);
$replyRsp = json_decode($replyResult);
echo $replyResult;
} catch (\Exception $e) {
echo var_dump($e);
}
```

- **发送国际/港澳台短信**

  发送国际/港澳台短信与发送国内短信类似，只需替换相应的国家码或地区码。详细示例请参考：
  - 指定模板单发短信
  - 指定模板群发短信

- 拉取短信回执

Python SDK

最近更新时间：2019-09-23 11:55:35

SDK 功能简介

目前腾讯云短信为客户提供**国内短信**和**国际/港澳台短信**服务，腾讯云短信 SDK 支持以下操作：

国内短信	国际/港澳台短信
• 指定模板单发短信 • 指定模板群发短信 • 拉取短信回执和短信回复状态	• 指定模板单发短信 • 指定模板群发短信 • 拉取短信回执

说明：

- 群发短信
  一次群发请求最多支持200个号码，如对号码数量有特殊需求请联系腾讯云短信技术支持（QQ：3012203387）。
- 拉取短信回执
  该功能默认关闭。您可以根据实际需求联系腾讯云短信技术支持（QQ：3012203387）开通，实现批量拉取短信回执。

SDK 使用指南

相关资料

各个接口及其参数的详情介绍请参见 API 指南 、SDK 文档 和 错误码。

前提条件

在使用 SDK 前，您需要准备以下信息：

- **获取 SDKAppID 和 AppKey**
  云短信应用 **SDKAppID** 和 **AppKey** 可在 短信控制台 的应用信息里获取。如您尚未添加应用，请登录 短信控制台 添加应用。
- **申请签名并确认审核通过**
  一个完整的短信由短信**签名**和**短信正文内容**两部分组成，短信**签名**需申请和审核，**签名**可在 短信控制台 的相应服务模块【内容配置】中进行申请，详细申请操作请参见 创建签名。发送国际/港澳台短信时，允许不携带签名。

- **申请模板并确认审核通过**
  短信正文内容**模板**需申请和审核，**模板**可在 短信控制台 的相应服务模块【内容配置】中进行申请，详细申请操作请参见 创建正文模板。

配置 SDK

- **pip 配置：**
  qcloudsms_py 采用 pip 进行安装，要使用 qcloudsms 功能, 只需要执行：

```
pip install qcloudsms_py
```

- **手动配置：**
  1.手动下载或 clone 最新版本 qcloudsms_py 代码。
  2.在 qcloudsms_py 目录运行 `python setup.py install` 或直接把 qcloudsms_py 所在目录加入 `sys.path` 。

> 说明：
> Python 2/Python 3都支持。

示例代码

> 说明：
> 所有示例代码仅作参考，无法直接编译和运行，需根据实际情况进行修改。

- **准备必要参数**

```
# 短信应用 SDK AppID
appid = 1400009099 # SDK AppID 以1400开头
# 短信应用 SDK AppKey
appkey = "9ff91d87c2cd7cd0ea762f141975d1df37481d48700d70ac37470aefc60f9bad"
# 需要发送短信的手机号码
phone_numbers = ["21212313123", "12345678902", "12345678903"]
# 短信模板ID，需要在短信控制台中申请
template_id = 7839 # NOTE: 这里的模板 ID`7839`只是示例，真实的模板 ID 需要在短信控制台中申请
# 签名
sms_sign = "腾讯云" # NOTE: 签名参数使用的是`签名内容`，而不是`签名ID`。这里的签名"腾讯云"只是示例，真实的签名需要在短信控制台中申请
```

- **指定模板 ID 单发短信**

```
from qcloudsms_py import SmsSingleSender
from qcloudsms_py.httpclient import HTTPError
ssender = SmsSingleSender(appid, appkey)
params = ["5678"] # 当模板没有参数时，`params = []`
try:
result = ssender.send_with_param(86, phone_numbers[0],
template_id, params, sign=sms_sign, extend="", ext="")
except HTTPError as e:
print(e)
except Exception as e:
print(e)
print(result)
```

- **指定模板 ID 群发短信**

```
from qcloudsms_py import SmsMultiSender
from qcloudsms_py.httpclient import HTTPError
msender = SmsMultiSender(appid, appkey)
params = ["5678"]
try:
result = msender.send_with_param(86, phone_numbers,
template_id, params, sign=sms_sign, extend="", ext="")
except HTTPError as e:
print(e)
except Exception as e:
print(e)
print(result)
```

- **拉取短信回执以及回复**

```
from qcloudsms_py import SmsStatusPuller
from qcloudsms_py.httpclient import HTTPError
max_num = 10 # 单次拉取最大量
spuller = SmsStatusPuller(appid, appkey)
try:
# 拉取短信回执
callback_result = spuller.pull_callback(max_num)
# 拉取回复，国际/港澳台短信不支持回复功能
reply_result = spuller.pull_reply(max_num)
except HTTPError as e:
print(e)
except Exception as e:
print(e)
```

```
print(callback_result)
print(reply_result)
```

- **拉取单个手机短信状态**

```
from qcloudsms_py import SmsMobileStatusPuller
from qcloudsms_py.httpclient import HTTPError
begin_time = 1511125600 # 开始时间（UNIX timestamp）
end_time = 1511841600 # 结束时间（UNIX timestamp）
max_num = 10 # 单次拉取最大量
mspuller = SmsMobileStatusPuller(appid, appkey)
try:
# 拉取短信回执
callback_result = mspuller.pull_callback("86", phone_numbers[0],
begin_time, end_time, max_num)
# 拉取回复，国际/港澳台短信不支持回复功能
reply_result = mspuller.pull_reply("86", phone_numbers[0],
begin_time, end_time, max_num)
except HTTPError as e:
print(e)
except Exception as e:
print(e)
print(callback_result)
print(reply_result)
```

- **发送国际/港澳台短信**

  发送国际/港澳台短信与发送国内短信类似，只需替换相应的国家码或地区码。详细示例请参考：

  - 指定模板单发短信
  - 指定模板群发短信
  - 拉取短信回执

- **使用代理**

  有的环境需要使用代理才能上网，可以指定 HTTPSimpleClient 的 proxy 参数来实现, 示例如下:

```
from qcloudsms_py import SmsSingleSender
from qcloudsms_py.httpclient import HTTPSimpleClient, HTTPError
httpclient = HTTPSimpleClient(proxy="www.proxysever.com:8080")
ssender = SmsSingleSender(appid, appkey, httpclient=httpclient)
template_id = 7839
params = ["5678"]
try:
```

```
result = ssender.send_with_param(86, phone_numbers[0],
template_id, params, sign=sms_sign, extend="", ext="")
except HTTPError as e:
print(e)
except Exception as e:
print(e)
print(result)
```

- **统一创建对象**

  短信类的对象可以通过 qcloudsms_py.QcloudSms 统一创建，这种方式可以避免创建对象时多次传入参数 appid 和 appkey， 示例如下：

```
from qcloudsms_py import QcloudSms
# 创建 QcloudSms 对象
qcloudsms = QcloudSms(appid, appkey)
# 创建单发短信 SmsSingleSender 对象
ssender = qcloudsms.SmsSingleSender()
```

Node.js SDK

最近更新时间：2019-09-23 11:55:46

SDK 功能简介

目前腾讯云短信为客户提供**国内短信和**国际/港澳台短信**服务，腾讯云短信 SDK 支持以下操作：

国内短信	国际/港澳台短信
• 指定模板单发短信 • 指定模板群发短信 • 拉取短信回执和短信回复状态	• 指定模板单发短信 • 指定模板群发短信 • 拉取短信回执

说明：

- 群发短信
  一次群发请求最多支持200个号码，如对号码数量有特殊需求请联系腾讯云短信技术支持（QQ：3012203387）。
- 拉取短信回执
  该功能默认关闭。您可以根据实际需求联系腾讯云短信技术支持（QQ：3012203387）开通，实现批量拉取短信回执。

SDK 使用指南

相关资料

各个接口及其参数的详情介绍请参见 API 指南 、SDK 文档 和 错误码。

前提条件

在使用 SDK 前，您需要准备以下信息：

- **获取 SDKAppID 和 AppKey**
  云短信应用 **SDKAppID** 和 **AppKey** 可在 短信控制台 的应用信息里获取。如您尚未添加应用，请登录 短信控制台 添加应用。
- **申请签名并确认审核通过**
  一个完整的短信由短信**签名**和**短信正文内容**两部分组成，短信**签名**需申请和审核，**签名**可在 短信控制台 的相应服务模块【内容配置】中进行申请，详细申请操作请参见 创建签名。发送国际/港澳台短信时，允许不携带签名。

- **申请模板并确认审核通过**
  短信正文内容**模板**需申请和审核，**模板**可在 短信控制台 的相应服务模块【内容配置】中进行申请，详细申请操作请参见 创建正文模板。

配置 SDK

- **npm 配置：**
  qcloudsms_js 采用 npm 进行安装，要使用 qcloudsms 功能，只需要执行：

```
npm install qcloudsms_js
```

- **手动配置：**
  1.手动下载或 clone 最新版本 qcloudsms_js 代码。
  2.把 qcloudsms_js 把代码放入项目目录。
  3.在项目里 require qcloudsms_js， 如： `var moduleName = require("path/to/qcloudsms_js")` 。

示例代码

> 说明：
> 所有示例代码仅作参考，无法直接编译和运行，需根据实际情况进行修改。

- 准备必要参数和实例化 QcloudSms

```
var QcloudSms = require("qcloudsms_js");
// 短信应用 SDK AppID
var appid = 1400009099; // SDK AppID 以1400开头
// 短信应用 SDK AppKey
var appkey = "9ff91d87c2cd7cd0ea762f141975d1df37481d48700d70ac37470aefc60f9bad";
// 需要发送短信的手机号码
var phoneNumbers = ["21212313123", "12345678902", "12345678903"];
// 短信模板 ID，需要在短信控制台中申请
var templateId = 7839; // NOTE: 这里的模板ID`7839`只是示例，真实的模板 ID 需要在短信控制台中申请
// 签名
var smsSign = "腾讯云"; // NOTE: 签名参数使用的是`签名内容`，而不是`签名ID`。这里的签名"腾讯云"只是示例，真实的签名需要在短信控制台申请
// 实例化 QcloudSms
var qcloudsms = QcloudSms(appid, appkey);
// 设置请求回调处理, 这里只是演示，用户需要自定义相应处理回调
function callback(err, res, resData) {
if (err) {
```

```
console.log("err: ", err);
} else {
console.log("request data: ", res.req);
console.log("response data: ", resData);
}
}
```

- **指定模板 ID 单发短信**

```
var ssender = qcloudsms.SmsSingleSender();
var params = ["5678"];
ssender.sendWithParam("86", phoneNumbers[0], templateId,
params, smsSign, "", "", callback);
```

- **指定模板 ID 群发短信**

```
var msender = qcloudsms.SmsMultiSender();
var params = ["5678"];
msender.sendWithParam("86", phoneNumbers, templateId,
params, smsSign, "", "", callback);
```

- **拉取短信回执以及回复**

```
var maxNum = 10; // 单次拉取最大量
var spuller = qcloudsms.SmsStatusPuller();
// 拉取短信回执
spuller.pullCallback(maxNum, callback);
// 拉取回复，国际/港澳台短信不支持回复功能
spuller.pullReply(maxNum, callback);
```

- **拉取单个手机短信状态**

```
var beginTime = 1511125600; // 开始时间（UNIX timestamp）
var endTime = 1511841600; // 结束时间（UNIX timestamp）
var maxNum = 10; // 单次拉取最大量
var mspuller = qcloudsms.SmsMobileStatusPuller();
// 拉取短信回执
mspuller.pullCallback("86", phoneNumbers[0], beginTime, endTime, maxNum, callback);
// 拉取回复，国际/港澳台短信不支持回复功能
mspuller.pullReply("86", phoneNumbers[0], beginTime, endTime, maxNum, callback);
```

- **发送国际/港澳台短信**

  发送国际/港澳台短信与发送国内短信类似，只需替换相应的国家码或地区码。详细示例请参考：

  - 指定模板单发短信
  - 指定模板群发短信
  - 拉取短信回执

C# SDK

最近更新时间：2019-09-23 11:55:56

SDK 功能简介

目前腾讯云短信为客户提供**国内短信**和**国际/港澳台短信**服务，腾讯云短信 SDK 支持以下操作：

国内短信	国际/港澳台短信
• 指定模板单发短信 • 指定模板群发短信 • 拉取短信回执和短信回复状态	• 指定模板单发短信 • 指定模板群发短信 • 拉取短信回执

说明：

- 群发短信
  一次群发请求最多支持200个号码，如对号码数量有特殊需求请联系腾讯云短信技术支持（QQ：3012203387）。
- 拉取短信回执
  该功能默认关闭。您可以根据实际需求联系腾讯云短信技术支持（QQ：3012203387）开通，实现批量拉取短信回执。

SDK 使用指南

相关资料

各个接口及其参数的详情介绍请参见 API 指南 、SDK 文档 和 错误码。

前提条件

在使用 SDK 前，您需要准备以下信息：

- **获取 SDKAppID 和 AppKey**
  云短信应用 **SDKAppID** 和 **AppKey** 可在 短信控制台 的应用信息里获取。如您尚未添加应用，请登录 短信控制台 添加应用。
- **申请签名并确认审核通过**
  一个完整的短信由短信**签名**和**短信正文内容**两部分组成，短信**签名**需申请和审核，**签名**可在 短信控制台 的相应服务模块【内容配置】中进行申请，详细申请操作请参见 创建签名。发送国际/港澳台短信时，允许不携带签名。

- **申请模板并确认审核通过**
  短信正文内容**模板**需申请和审核，**模板**可在 短信控制台 的相应服务模块【内容配置】中进行申请，详细申请操作请参见 创建正文模板。

配置 SDK

- **nuget：**
  要使用 qcloudsms_csharp 功能，只需要在 .nuspec 文件中添加如下依赖：

```
<dependencies>
<dependency id="qcloud.qcloudsms_csharp" version="0.1.5" />
</dependencies>
```

  或参考 nuget 官方网站 进行安装。

- **命令行**
  - Package Manager

```
Install-Package qcloud.qcloudsms_csharp -Version 0.1.5
```

  - .NET CLI

```
dotnet add package qcloud.qcloudsms_csharp --version 0.1.5
```

  - Paket CLI

```
paket add qcloud.qcloudsms_csharp --version 0.1.5
```

示例代码

> 说明：
> 所有示例代码仅作参考，无法直接编译和运行，需根据实际情况进行修改。

- **准备必要参数**

```
// 短信应用 SDK AppID
int appid = 122333333;
// 短信应用 SDK AppKey
string appkey = "9ff91d87c2cd7cd0ea762f141975d1df37481d48700d70ac37470aefc60f9bad";
```

```
// 需要发送短信的手机号码
string[] phoneNumbers = {"21212313123", "12345678902", "12345678903"};
// 短信模板 ID，需要在短信控制台中申请
int templateId = 7839; // NOTE: 这里的模板 ID`7839`只是示例，真实的模板 ID 需要在短信控制台中申请
// 签名
string smsSign = "腾讯云"; // NOTE: 签名参数使用的是`签名内容`，而不是`签名ID`。这里的签名"腾讯云"只是示例，真实的签名需要在短信控制台申请
```

- **指定模板 ID 单发短信**

```
using qcloudsms_csharp;
using qcloudsms_csharp.json;
using qcloudsms_csharp.httpclient;
using System;
try
{
SmsSingleSender ssender = new SmsSingleSender(appid, appkey);
var result = ssender.sendWithParam("86", phoneNumbers[0],
templateId, new[]{ "5678" }, smsSign, "", "");
Console.WriteLine(result);
}
catch (JSONException e)
{
Console.WriteLine(e);
}
catch (HTTPException e)
{
Console.WriteLine(e);
}
catch (Exception e)
{
Console.WriteLine(e);
}
```

- **指定模板 ID 群发短信**

```
using qcloudsms_csharp;
using qcloudsms_csharp.json;
using qcloudsms_csharp.httpclient;
using System;
try
{
SmsMultiSender msender = new SmsMultiSender(appid, appkey);
var sresult = msender.sendWithParam("86", phoneNumbers, templateId,
```

```
new[]{"5678"}, smsSign, "", "");
Console.WriteLine(sresult);
}
catch (JSONException e)
{
Console.WriteLine(e);
}
catch (HTTPException e)
{
Console.WriteLine(e);
}
catch (Exception e)
{
Console.WriteLine(e);
}
```

- **拉取短信回执以及回复**

```
using qcloudsms_csharp;
using qcloudsms_csharp.json;
using qcloudsms_csharp.httpclient;
using System;
try
{
// Note: 短信拉取功能需要联系腾讯云短信技术支持（QQ：3012203387）开通权限
int maxNum = 10; // 单次拉取最大量
SmsStatusPuller spuller = new SmsStatusPuller(appid, appkey);

// 拉取短信回执
var callbackResult = spuller.pullCallback(maxNum);
Console.WriteLine(callbackResult);

// 拉取回复，仅国内短信支持拉取回复状态
var replyResult = spuller.pullReply(maxNum);
Console.WriteLine(replyResult);
}
catch (JSONException e)
{
Console.WriteLine(e);
}
catch (HTTPException e)
{
Console.WriteLine(e);
}
```

```
catch (Exception e)
{
Console.WriteLine(e);
}
```

- **拉取单个手机短信状态**

```
using qcloudsms_csharp;
using qcloudsms_csharp.json;
using qcloudsms_csharp.httpclient;
using System;
try
{
int beginTime = 1511125600; // 开始时间（UNIX timestamp）
int endTime = 1511841600; // 结束时间（UNIX timestamp）
int maxNum = 10; // 单次拉取最大量
SmsMobileStatusPuller mspuller = new SmsMobileStatusPuller(appid, appkey);

// 拉取短信回执
var callbackResult = mspuller.pullCallback("86",
phoneNumbers[0], beginTime, endTime, maxNum);
Console.WriteLine(callbackResult);

// 拉取回复，国际/港澳台短信不支持回复功能
var replyResult = mspuller.pullReply("86",
phoneNumbers[0], beginTime, endTime, maxNum);
Console.WriteLine(replyResult);
}
catch (JSONException e)
{
Console.WriteLine(e);
}
catch (HTTPException e)
{
Console.WriteLine(e);
}
catch (Exception e)
{
Console.WriteLine(e);
}
```

- **发送国际/港澳台短信**

  发送国际/港澳台短信与发送国内短信类似，只需替换相应的国家码或地区码。详细示例请参考：

- 指定模板单发短信
- 指定模板群发短信
- 拉取短信回执

- **使用连接池**
  多个线程可以共用一个连接池发送 API 请求，多线程并发单发短信示例如下：

```
using qcloudsms_csharp;
using qcloudsms_csharp.httpclient;
using qcloudsms_csharp.json;
using System;
using System.Threading;
public class SmsTest
{
public class SmsArg
{
public SmsSingleSender sender;
public string nationCode;
public string phoneNumber;
public string msg;

public SmsArg(SmsSingleSender sender, string nationCode, string phoneNumber, string msg)
{
this.sender = sender;
this.nationCode = nationCode;
this.phoneNumber = phoneNumber;
this.msg = msg;
}
}

public static void SendSms(object data)
{
SmsArg arg = (SmsArg)data;
try
{
var result = arg.sender.send(0, arg.nationCode, arg.phoneNumber, arg.msg, "", "");
Console.WriteLine("{0}, {1}", result, arg.phoneNumber);
}
catch (JSONException e)
{
Console.WriteLine(e);
}
catch (HTTPException e)
{
```

```
Console.WriteLine(e);
}
catch (Exception e)
{
Console.WriteLine(e);
}
}

static void Main(string[] args)
{
int appid = 122333333;
string appkey = "9ff91d87c2cd7cd0ea762f141975d1df37481d48700d70ac37470aefc60f9bad";
string[] phoneNumbers = {
"21212313123", "12345678902", "12345678903",
"21212313124", "12345678903", "12345678904",
"21212313125", "12345678904", "12345678905",
"21212313126", "12345678905", "12345678906",
"21212313127", "12345678906", "12345678907",
};

// 创建一个连接池httpclient
PoolingHTTPClient httpclient = new PoolingHTTPClient();

// 创建SmsSingleSender时传入连接池http client
SmsSingleSender ssender = new SmsSingleSender(appid, appkey, httpclient);

// 创建线程
Thread[] threads = new Thread[phoneNumbers.Length];
for (int i = 0; i < phoneNumbers.Length; i++)
{
threads[i] = new Thread(SmsTest.SendSms);
}

// 运行线程
for (int i = 0; i < threads.Length; i++)
{
threads[i].Start(new SmsArg(ssender, "86", phoneNumbers[i], "您验证码是："));
}

// join线程
for (int i = 0; i < threads.Length; i++)
{
threads[i].Join();
}
```

```
// 关闭连接池httpclient
httpclient.close();
}
}
```

- **使用自定义 HTTP client 实现**
  如果需要使用自定义的 HTTP client 实现，只需实现 `qcloudsms_csharp.httpclient.IHTTPClient` 接口，并在构造 API 对象时传入自定义 HTTP client 即可，参考示例如下：

```
using qcloudsms_csharp;
using qcloudsms_csharp.httpclient;
// using myhttp_namespace;
public class CustomHTTPClient : IHTTPClient
{
public HTTPResponse fetch(HTTPRequest request)
{
// 1. 创建自定义 HTTP request
// MyHTTPrequest req = MyHTTPRequest.build(request)

// 2. 创建自定义 httpclient
// MyHTTPClient client = new MyHTTPClient();

// 3. 使用自定义 httpclient 获取 HTTP 响应
// MyHTTPResponse response = client.fetch(req);

// 4. 转换 HTTP 响应到 HTTPResponse
// HTTPResponse res = transformToHTTPResponse(response);

// 5. 返回 HTTPResponse 实例
// return res;
}

public void close()
{
}
}
// 创建自定义 httpclient
CustomHTTPClient httpclient = new CustomHTTPClient();
// 构造 API 对象时传入自定义 httpclient
SmsSingleSender ssender = new SmsSingleSender(appid, appkey, httpclient);
```

小程序

最近更新时间：2019-09-23 11:56:05

前提条件

在使用 SDK 前，您需要准备以下信息：

- **获取 SDKAppID 和 AppKey**
  云短信应用 **SDKAppID** 和 **AppKey** 可在 短信控制台 的应用信息里获取。如您尚未添加应用，请登录 短信控制台 添加应用。
- **申请签名并确认审核通过**
  一个完整的短信由短信**签名**和**短信正文内容**两部分组成，短信**签名**需申请和审核，**签名**可在 短信控制台 的相应服务模块【内容配置】中进行申请，详细申请操作请参见 创建签名。发送国际/港澳台短信时，允许不携带签名。
- **申请模板并确认审核通过**
  短信正文内容**模板**需申请和审核，**模板**可在 短信控制台 的相应服务模块【内容配置】中进行申请，详细申请操作请参见 创建正文模板。

参考文档

各个接口及其参数的详情介绍请参考 API 文档、SDK 文档、错误码 和 微信小程序·云开发。

使用指南

1. 安装微信开发者工具 IDE

根据 微信小程序云开发·起步 注册并获取小程序 AppID，同时安装微信开发者工具。

2. 新建小程序

1. 选择【小程序项目】>【小程序】，单击  。
2. 根据实际需求，填写以下参数：
   - 项目名称：请填写项目的名称。
   - 目录：请选择项目文件夹。
   - AppID：请输入注册时获取的正确 AppID。

> 注意：
> 小程序.云开发场景不支持测试号和无 AppID。

   - 开发模式：请选择【小程序】
   - 后端服务：支持【不使用云服务】、【小程序.云开发】和【腾讯云】三种选择，本文以选择【小程序.云开发】为例。

> 说明：
> 当使用【不使用云服务】或【腾讯云】时需自建后端 Server，由后端 Server 再调用云短信 API，相关 API 以及 SDK 请参考 腾讯云短信 API 以及 腾讯云短信 SDK。

3. 新建云函数

1. 右键单击【cloudfunctions】文件夹，选择【新建 Node.js 云函数】。
2. 输入 `sendsms` ，按 `Enter` 确认。

4. 添加云函数依赖库

打开 `/cloudfunctions/sendsms/package.json` 文件，在 `dependencies` 中添加 `qcloudsms_js` 库，如下所示：

```
{
"name": "sendsms",
"version": "1.0.0",
"description": "",
"main": "index.js",
"scripts": {
"test": "echo \"Error: no test specified\" && exit 1"
```

```
},
"author": "",
"license": "ISC",
"dependencies": {
"wx-server-sdk": "latest",
"qcloudsms_js": "^0.1.1"
}
}
```

5. 编写云函数处理逻辑

在 /cloudfunctions/sendsms/index.js 文件中添加处理发送短信实现逻辑，本部分主要调用 qcloudsms_js 库相关接口。

```
// 云函数入口文件
const cloud = require('wx-server-sdk')
const QcloudSms = require("qcloudsms_js")
const appid = 140000001 // 替换成您申请的云短信 AppID 以及 AppKey
const appkey = "abcdefghijkl123445"
const templateId = 1234 // 替换成您所申请模板 ID
const smsSign = "腾讯云" // 替换成您所申请的签名

cloud.init()

// 云函数入口函数
exports.main = async (event, context) => new Promise((resolve, reject) => {
/*单发短信示例为完整示例，更多功能请直接替换以下代码*/
var qcloudsms = QcloudSms(appid, appkey);
var ssender = qcloudsms.SmsSingleSender();
var params = ["5678"];
// 获取发送短信的手机号码
var mobile = event.mobile
// 获取手机号国家/地区码
var nationcode = event.nationcode
ssender.sendWithParam(nationcode, mobile, templateId, params, smsSign, "", "", (err, res, resData) =>
{
/*设置请求回调处理, 这里只是演示，您需要自定义相应处理逻辑*/
if (err) {
console.log("err: ", err);
reject({ err })
} else {
resolve({ res: res.req, resData })
}
}
```

```
);

})
```

6. 部署云函数

右键单击 /cloudfunctions/sendsms 文件夹，选择【创建并部署：云端安装依赖】。

7. 调用云函数

小程序提供 callfunction 进行云函数调用，您可以根据实际需求在任意JS逻辑处理函数中添加如下代码进行云函数调用。
例如，需要在页面小程序首页加载完成后发送短信，则可以在 miniprogram/index/index.js 的 onLoad 函数中添加如下代码。

```
if(!wx.cloud) {
wx.redirectTo({
url: '../chooseLib/chooseLib',
})
return
}
wx.cloud.callFunction({
name: 'sendsms',
data: {
mobile: '19012345678',
nationcode: '86'
},
success: res => {
console.log('[云函数] [sendsms] 调用成功')
console.log(res)
},
fail: err => {
console.error('[云函数] [sendsms] 调用失败', err)
}
})
```

最后编译您的小程序，进行预览，至此您的小程序已可进行短信下发。

说明：

- 如果程序中调用失败可以参考接口 错误码 说明，更多常见问题可参阅 FAQ。
- 如需使用更多功能，例如指定内容单发短信，指定模板群发短信以及拉取短信回执状态等，请参考 qcloudsms_js 库或自行封装 HTTP 接口。

- 无论单发/群发短信还是指定模板 ID 单发/群发短信都需要**从控制台中申请模板并且模板已经审核通过**，才可能下发成功，否则返回失败。