Serverless Framework

最佳实践

产品文档

腾讯云

文档目录

最佳实践
部署 Express 框架

最近更新时间：2019-11-25 13:03:50

操作场景

Express 组件通过使用 serverless-tencent 的基础组件（如 API 网关组件、SCF 组件等），快速且方便地在腾讯云创建、配置和管理一个 Express 框架。

Serverless Framework + 腾讯云 >>

操作步骤

通过 Express 组件，对一个 Express 应用进行完整的创建、配置、部署和删除等操作，支持的命令如下：

安装

通过 npm 安装 Serverless：

```
$ npm install -g serverless
```

创建

本地创建 `serverless.yml` 文件：

```
$ touch serverless.yml
```

初始化一个新的 npm 包，并安装 Express：

```
npm init # 创建后持续回车
npm i --save express # 安装express
```

创建一个 `app.js` 文件，并在其中创建您的 Express App：

```
const express = require('express')
const app = express()

app.get('/', function(req, res) {
res.send('Hello Express')
```

```
})

// don't forget to export!
module.exports = app
```

配置

在 serverless.yml 中进行如下配置：

```
# serverless.yml

express:
component: '@serverless/tencent-express'
inputs:
region: ap-shanghai
```

查看详细配置文档>>

部署

如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

通过 `sls` 命令进行部署，并可以添加 `--debug` 参数查看部署过程中的信息：

```
$ sls --debug

DEBUG ─ Resolving the template's static variables.
DEBUG ─ Collecting components from the template.
DEBUG ─ Downloading any NPM components found in the template.
DEBUG ─ Analyzing the template's components dependencies.
DEBUG ─ Creating the template's components graph.
DEBUG ─ Syncing template state.
DEBUG ─ Executing the template's components graph.
DEBUG ─ Compressing function ExpressComponent_7xRrrd file to /Users/dfounderliu/Desktop/temp/code/.serverless/ExpressComponent_7xRrrd.zip.
DEBUG ─ Compressed function ExpressComponent_7xRrrd file successful
DEBUG ─ Uploading service package to cos[sls-cloudfunction-ap-shanghai-code]. sls-cloudfunction-default-ExpressComponent_7xRrrd-1572512568.zip
DEBUG ─ Uploaded package successful /Users/dfounderliu/Desktop/temp/code/.serverless/ExpressComponent_7xRrrd.zip
DEBUG ─ Creating function ExpressComponent_7xRrrd
DEBUG ─ Created function ExpressComponent_7xRrrd successful
DEBUG ─ Starting API-Gateway deployment with name express.TencentApiGateway in the ap-shanghai region
DEBUG ─ Using last time deploy service id service-n0vs2ohb
DEBUG ─ Updating service with serviceId service-n0vs2ohb.
```

```
DEBUG – Endpoint ANY / already exists with id api-9z60urs4.
DEBUG – Updating api with api id api-9z60urs4.
DEBUG – Service with id api-9z60urs4 updated.
DEBUG – Deploying service with id service-n0vs2ohb.
DEBUG – Deployment successful for the api named express.TencentApiGateway in the ap-shanghai r
egion.

express:
region: ap-shanghai
functionName: ExpressComponent_7xRrrd
apiGatewayServiceId: service-n0vs2ohb
url: http://service-n0vs2ohb-1300415943.ap-shanghai.apigateway.myqcloud.com/release/

36s › express › done
```

部署完毕后，可以在浏览器中访问返回的链接中看到对应的 Express 返回值。

移除

通过以下命令移除已部署的存储桶：

```
$ sls remove --debug

DEBUG – Flushing template state and removing all components.
DEBUG – Removed function ExpressComponent_MHrAzr successful
DEBUG – Removing any previously deployed API. api-kf2hxrhc
DEBUG – Removing any previously deployed service. service-4ndfl6pz

13s › express › done
```

账号配置（可选）

当前默认支持 CLI 扫描二维码登录，如您希望配置持久的环境变量/密钥信息，也可以本地创建 `.env` 文件：

```
$ touch .env # 腾讯云的配置信息
```

在 `.env` 文件中配置腾讯云的 SecretId 和 SecretKey 信息并保存：

```
# .env
TENCENT_SECRET_ID=123
TENCENT_SECRET_KEY=123
```

说明：

- 如果没有腾讯云账号，请先 注册新账号。
- 如果已有腾讯云账号，可以在 API 密钥管理 中获取 SecretId 和 SecretKey。

部署静态网站

最近更新时间：2019-12-25 20:24:13

操作场景

静态网站 Component 通过腾讯云对象存储 COS 组件，快速部署静态页面到 COS 中，并生成外网访问的域名。

参考本文 操作步骤，您可以将自定义的 Web 页面托管到对象存储 COS 中，并生成域名供外网访问。基于该静态网站 Component ，您可以更方便的搭建博客系统（如 Hexo ）；您也可以稍加扩展该静态网站 Component，以实现前端框架的支持（如 Vue、React ）。

操作步骤

安装

通过 npm 安装 Serverless：

```
$ npm install -g serverless
```

创建

本地创建 my-website 文件夹：

```
$ mkdir my-website
$ cd my-website
```

在文件夹中创建对应的 `serverless.yml` 文件，并将静态页面放在 `code` 目录下，文件目录结构如下：

```
$ touch serverless.yml
```

```
|- code
|- index.html
|- serverless.yml
```

`code` 目录下应该对应 HTML/CSS/JS 资源的文件，或者一个完整的 React 应用。
下载 示例 HTML，将以下代码放在 index.html 文件中：

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
<meta charset="UTF-8">
<title>Hello, Tencent Cloud</title>
</head>
<body>
Hello, Tencent Cloud
</body>
</html>
```

配置

在 `serverless.yml` 文件中进行如下配置：

```
# serverless.yml

myWebsite:
component: "@serverless/tencent-website"
inputs:
code:
src: ./code
index: index.html
error: index.html
region: ap-guangzhou
bucketName: my-bucket
```

查看详细配置文档>>

部署

如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

通过 `sls` 命令进行部署，并可以添加 `--debug` 参数查看部署过程中的信息：

```
$ sls --debug

DEBUG ─ Resolving the template's static variables.
DEBUG ─ Collecting components from the template.
DEBUG ─ Downloading any NPM components found in the template.
DEBUG ─ Analyzing the template's components dependencies.
DEBUG ─ Creating the template's components graph.
DEBUG ─ Syncing template state.
DEBUG ─ Executing the template's components graph.
DEBUG ─ Starting Website Component.
DEBUG ─ Preparing website Tencent COS bucket my-bucket-1300415943.
DEBUG ─ Deploying "my-bucket-1300415943" bucket in the "ap-guangzhou" region.
DEBUG ─ "my-bucket-1300415943" bucket was successfully deployed to the "ap-guangzhou" region.
```

```
DEBUG ─ Setting ACL for "my-bucket-1300415943" bucket in the "ap-guangzhou" region.
DEBUG ─ Ensuring no CORS are set for "my-bucket-1300415943" bucket in the "ap-guangzhou" regi
on.
DEBUG ─ Ensuring no Tags are set for "my-bucket-1300415943" bucket in the "ap-guangzhou" regio
n.
DEBUG ─ Configuring bucket my-bucket-1300415943 for website hosting.
DEBUG ─ Uploading website files from /Users/dfounderliu/Desktop/temp/code/src to bucket my-buc
ket-1300415943.
DEBUG ─ Starting upload to bucket my-bucket-1300415943 in region ap-guangzhou
DEBUG ─ Uploading directory /Users/dfounderliu/Desktop/temp/code/src to bucket my-bucket-1300
415943
DEBUG ─ Website deployed successfully to URL: https://my-bucket-1300415943.cos-website.ap-guan
gzhou.myqcloud.com.

myWebsite:
url: https://my-bucket-1300415943.cos-website.ap-guangzhou.myqcloud.com
env:

2s › myWebsite › done
```

移除

通过以下命令移除项目：

```
sls remove --debug

DEBUG ─ Flushing template state and removing all components.
DEBUG ─ Starting Website Removal.
DEBUG ─ Removing Website bucket.
DEBUG ─ Removing files from the "my-bucket-1300415943" bucket.
DEBUG ─ Removing "my-bucket-1300415943" bucket from the "ap-guangzhou" region.
DEBUG ─ "my-bucket-1300415943" bucket was successfully removed from the "ap-guangzhou" regio
n.
DEBUG ─ Finished Website Removal.

3s › myWebsite › done
```

账号配置（可选）

当前默认支持 CLI 扫描二维码登录，如您希望配置持久的环境变量/密钥信息，也可以本地创建 `.env` 文件：

```
$ touch .env # 腾讯云的配置信息
```

在 `.env` 文件中配置腾讯云的 SecretId 和 SecretKey 信息并保存：

```
# .env
TENCENT_SECRET_ID=123
TENCENT_SECRET_KEY=123
```

说明：

- 如果没有腾讯云账号，请先 注册新账号。
- 如果已有腾讯云账号，可以在 API 密钥管理 中获取 SecretId 和 SecretKey。

快速构建 REST API

最近更新时间：2019-12-31 11:25:22

操作场景

REST API 模板使用 Tencent SCF 组件及其触发器能力，方便的在腾讯云创建，配置和管理一个 REST API 应用。您可以通过 Serverless SCF 组件快速构建一个 REST API 应用，实现 GET/PUT 操作。

操作步骤

安装

通过 npm 安装 Serverless Framework：

```
$ npm install -g serverless
```

配置

通过如下命令直接下载示例：

```
$ serverless create --template-url https://github.com/serverless/components/tree/master/templates/tencent-python-rest-api
```

目录结构如下：

```
.
├── code
| └── index.py
└── serverless.yml
```

查看 code/index.py 代码，可以看到接口的传参和返回逻辑：

```
# -*- coding: utf8 -*-

def teacher_go():
# todo: teacher_go action
return {
"result": "it is student_get action"
}
```

```
def student_go():
# todo: student_go action
return {
"result": "it is teacher_put action"
}

def student_come():
# todo: student_come action
return {
"result": "it is teacher_put action"
}

def main_handler(event, context):
print(str(event))
if event["pathParameters"]["user_type"] == "teacher":
if event["pathParameters"]["action"] == "go":
return teacher_go()
if event["pathParameters"]["user_type"] == "student":
if event["pathParameters"]["action"] == "go":
return student_go()
if event["pathParameters"]["action"] == "come":
return student_come()
```

部署

通过 sls 命令进行部署，并可以添加 --debug 参数查看部署过程中的信息。
如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

```
$ serverless --debug

DEBUG ─ Resolving the template's static variables.
DEBUG ─ Collecting components from the template.
DEBUG ─ Downloading any NPM components found in the template.
DEBUG ─ Analyzing the template's components dependencies.
DEBUG ─ Creating the template's components graph.
DEBUG ─ Syncing template state.
DEBUG ─ Executing the template's components graph.
DEBUG ─ Compressing function myRestAPI file to /Users/dfounderliu/Desktop/restAPI/component/.s
erverless/myRestAPI.zip.
DEBUG ─ Compressed function myRestAPI file successful
DEBUG ─ Uploading service package to cos[sls-cloudfunction-ap-singapore-code]. sls-cloudfunction
-default-myRestAPI-1574856533.zip
DEBUG ─ Uploaded package successful /Users/dfounderliu/Desktop/restAPI/component/.serverless/
myRestAPI.zip
DEBUG ─ Creating function myRestAPI
```

```
DEBUG – Updating code...
DEBUG – Updating configure...
DEBUG – Created function myRestAPI successful
DEBUG – Setting tags for function myRestAPI
DEBUG – Creating trigger for function myRestAPI
DEBUG – Starting API-Gateway deployment with name myRestAPI.serverless in the ap-singapore regi
on
DEBUG – Service with ID service-ibmk6o22 created.
DEBUG – API with id api-pjs3q3qi created.
DEBUG – Deploying service with id service-ibmk6o22.
DEBUG – Deployment successful for the api named myRestAPI.serverless in the ap-singapore region.
DEBUG – Deployed function myRestAPI successful

myRestAPI:
Name: myRestAPI
Runtime: Python3.6
Handler: index.main_handler
MemorySize: 128
Timeout: 20
Region: ap-singapore
Role: QCS_SCFExcuteRole
Description: My Serverless Function
APIGateway:
- serverless - http://service-ibmk6o22-1250000000.sg.apigw.tencentcs.com/release

10s › myRestAPI › done
```

测试

通过如下命令测试 REST API 的返回情况：

> 说明：
> 如果 Windows 系统中未安装 `curl` ，也可以直接通过浏览器打开对应链接查看返回情况。

```
$ curl -XGET http://service-9t28e0tg-1250000000.sg.apigw.tencentcs.com/release/users/teacher/go

{"result": "it is student_get action"}
```

```
$ curl -PUT http://service-9t28e0tg-1250000000.sg.apigw.tencentcs.com/release/users/student/go

{"result": "it is teacher_put action"}
```

移除

通过以下命令移除 REST API 应用：

```
$ sls remove --debug

DEBUG ─ Flushing template state and removing all components.
DEBUG ─ Removing any previously deployed API. api-37gk3l8q
DEBUG ─ Removing any previously deployed service. service-9t28e0tg
DEBUG ─ Removing function
DEBUG ─ Request id
DEBUG ─ Removed function myRestAPI successful

7s » myRestAPI » done
```

账号配置（可选）

当前默认支持 CLI 扫描二维码登录，如您希望配置持久的环境变量/密钥信息，也可以本地创建 `.env` 文件：

```
$ touch .env # 腾讯云的配置信息
```

在 `.env` 文件中配置腾讯云的 SecretId 和 SecretKey 信息并保存。

```
# .env
TENCENT_SECRET_ID=123
TENCENT_SECRET_KEY=123
```

说明：

- 如果没有腾讯云账号，请先 注册新账号。
- 如果已有腾讯云账号，可以在 API 密钥管理 中获取 SecretId 和 SecretKey。

快速部署 Hexo 博客

最近更新时间：2020-03-03 11:42:22

操作场景

该任务指导您通过 Serverless Website 组件，快速构建一个 Serverless Hexo 站点。

前提条件

- 已安装 Node.js（Node.js 版本需不低于8.6，建议使用 Node.js10.0 及以上版本）
- 已安装 Git

如您未安装上述应用程序，可以参考 Hexo 安装说明。

操作步骤

安装

通过 npm 安装 Serverless Framework：

```
$ npm install -g serverless
```

通过 npm 安装 Hexo：

```
$ npm install -g hexo-cli
```

安装 Hexo 完成后，请执行下列命令，Hexo 将会在指定文件夹中新建所需要的文件。

```
$ hexo init hexo # 生成 Hexo 目录
$ cd hexo
$ npm install
```

新建完成后，指定文件夹的目录如下：

```
.
├── _config.yml
├── package.json
├── scaffolds
```

```
├── source
| ├── _drafts
| └── _posts
└── themes
```

安装完成后，可以通过 hexo g 命令生成静态页面：

```
$ hexo g # generate
```

> 说明：
> 如果希望在本地查看效果，也可以运行下列命令，通过浏览器访问 localhost:4000 查看页面效果。

```
$ hexo s # server
```

配置

在 hexo 目录下，创建 serverless.yml 文件：

```
$ touch serverless.yml
```

在 serverless.yml 文件中进行如下配置：

```
# serverless.yml

myWebsite:
component: '@serverless/tencent-website'
inputs:
code:
src: ./public # Upload static files generated by HEXO
index: index.html
error: index.html
region: ap-guangzhou
bucketName: my-bucket
```

配置完成后，文件目录如下：

```
.
├── .serverless
├── hexo
| ├── public
| ├── ...
```

```
| ├── _config.yml
| ├── ...
| └── source
└── serverless.yml
```

部署

通过 sls 命令进行部署，并可以添加 --debug 参数查看部署过程中的信息。
如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

```
$ serverless --debug

DEBUG ─ Resolving the template's static variables.
DEBUG ─ Collecting components from the template.
DEBUG ─ Downloading any NPM components found in the template.
DEBUG ─ Analyzing the template's components dependencies.
DEBUG ─ Creating the template's components graph.
DEBUG ─ Syncing template state.
DEBUG ─ Executing the template's components graph.
DEBUG ─ Starting Website Component.

Please scan QR code login from wechat
Wait login...
Login successful for TencentCloud
DEBUG ─ Preparing website Tencent COS bucket my-bucket-1250000000.
DEBUG ─ Deploying "my-bucket-1250000000" bucket in the "ap-guangzhou" region.
DEBUG ─ "my-bucket-1250000000" bucket was successfully deployed to the "ap-guangzhou" region.
DEBUG ─ Setting ACL for "my-bucket-1250000000" bucket in the "ap-guangzhou" region.
DEBUG ─ Ensuring no CORS are set for "my-bucket-1250000000" bucket in the "ap-guangzhou" regi
on.
DEBUG ─ Ensuring no Tags are set for "my-bucket-1250000000" bucket in the "ap-guangzhou" regio
n.
DEBUG ─ Configuring bucket my-bucket-1250000000 for website hosting.
DEBUG ─ Uploading website files from D:\hexotina\localhexo\public to bucket my-bucket-12500000
00.
DEBUG ─ Starting upload to bucket my-bucket-1250000000 in region ap-guangzhou
DEBUG ─ Uploading directory D:\hexotina\localhexo\public to bucket my-bucket-1250000000
DEBUG ─ Website deployed successfully to URL: https://my-bucket-1250000000.cos-website.ap-guan
gzhou.myqcloud.com.

myWebsite:
url: https://my-bucket-1250000000.cos-website.ap-guangzhou.myqcloud.com
env:

13s » myWebsite » done
```

访问命令行输出的 website url，即可查看您的 Serverless Hexo 站点

> 注意：
> 如果希望更新 Hexo 站点中的文章，需要在本地重新运行 `hexo g` 进行生成静态页面，再运行 `serverless` 更新到页面。

移除

通过以下命令移除 Hexo 网站：

```
$ sls remove --debug

DEBUG ─ Flushing template state and removing all components.
DEBUG ─ Starting Website Removal.
DEBUG ─ Removing Website bucket.
DEBUG ─ Removing files from the "my-bucket-1250000000" bucket.
DEBUG ─ Removing "my-bucket-1250000000" bucket from the "ap-guangzhou" region.
DEBUG ─ "my-bucket-1250000000" bucket was successfully removed from the "ap-guangzhou" regio
n.
DEBUG ─ Finished Website Removal.

6s » myWebsite » done
```

账号配置（可选）

当前默认支持 CLI 扫描二维码登录，如您希望配置持久的环境变量/密钥信息，也可以本地创建 `.env` 文件：

```
$ touch .env # 腾讯云的配置信息
```

在 `.env` 文件中配置腾讯云的 SecretId 和 SecretKey 信息并保存：

```
# .env
TENCENT_SECRET_ID=123
TENCENT_SECRET_KEY=123
```

> 说明：
>
> - 如果没有腾讯云账号，请先 注册新账号。
> - 如果已有腾讯云账号，可以在 API 密钥管理 中获取 SecretId 和 SecretKey。

快速部署一个全栈应用（Vue.js+Express.js）

最近更新时间：2019-12-31 14:37:01

操作场景

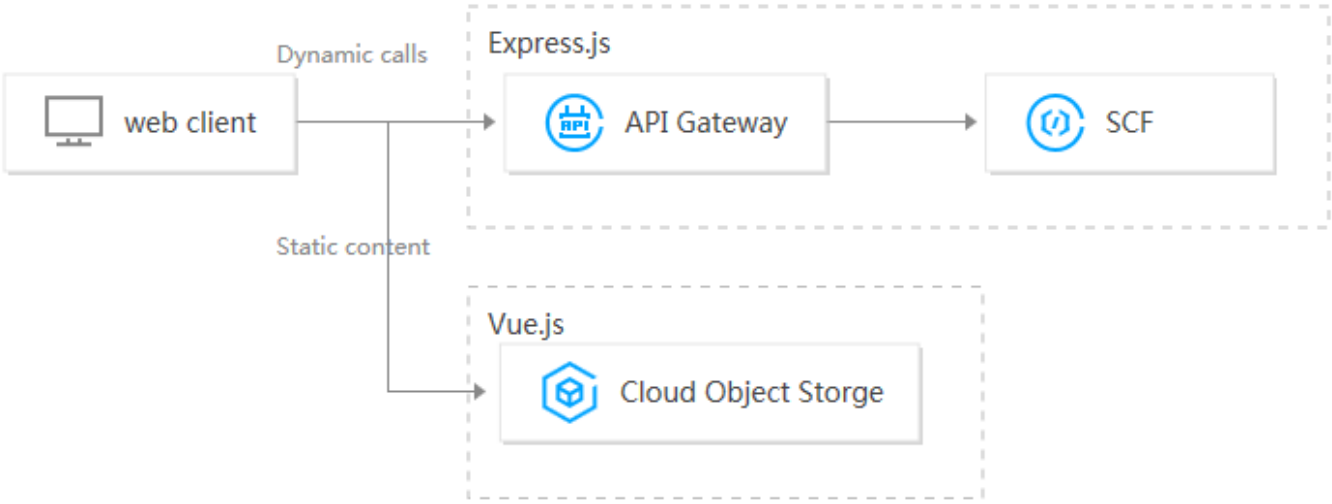
全栈组件（Vue.js+Express.js）用于通过多个 Serverless Components 部署 Serverless 全栈应用程序。可以帮助开发者更方便快捷的部署 Serverless 应用，例如利用后端 API 与前端 Vue.js 结合等场景。

此项目完全基于腾讯云 Serverless 服务器，可很大程度的缩减使用成本。 如果您正在寻找一个低开销的便捷轻量的 Serverless 服务管理框架，全栈组件（Vue.js+Express.js）将是很好的选择。

该 Template 包括：

- **serverless Express.js 后端**：由腾讯云 Servelress Cloud Function（云函数 SCF） 和腾讯云 API 网关提供相关能力，支持 express.js 框架，帮助开发者架构自己的项目和路由。
- **serverless Vue.js 前端**：由腾讯云 Cloud Object Storage（对象存储 COS）提供相关存储能力，通过后端 API 传递到前端，并使用 Vue.js 做相关渲染。

该全栈 Web 应用架构图如下：



操作步骤

安装

1. 通过如下命令安装 Serverless Framework：

```
$ npm i -g serverless
```

2. 新建一个本地文件夹，使用 `create --template-url` ，安装相关 template。您也可以将文件直接下载到本地：

```
serverless create --template-url https://github.com/serverless/components/tree/master/templates/tencent-fullstack-vue-application
```

使用 `cd` 命令，进入 `templates\tencent-fullstack-application` 文件夹，可以查看到如下目录结构：

```
|- api
|- dashboard
|- serverless.yml # 使用项目中的 yml 文件
```

分别在 `dashboard` 和 `api` 两个文件目录执行 NPM 依赖的安装，如下命令所示：

```
$ cd dashboard
$ npm i
```

```
$ cd api
$ npm i
```

部署

回到 `tencent-fullstack-application` 目录下，直接通过 `serverless` 命令来部署应用:

```
$ serverless
```

如果希望查看部署详情，可以通过调试模式的命令 `serverless --debug` 进行部署。

如您的账号未 登录 或 注册 腾讯云，您可以直接通过微信扫描命令行中的二维码进行授权登录和注册。

部署成功后，可以直接在浏览器中访问日志中返回的 dashboard url 地址，查看该全栈 Web App 的效果：

```
dashboard:
url: https://jcwm1l-myappid.cos-website.ap-guangzhou.myqcloud.com
env:
apiUrl: https://service-id-myappid.gz.apigw.tencentcs.com/release/
api:
region: undefined
functionName: tencent-fullstack-api
apiGatewayServiceId: service-id
url: https://service-id-myappid.gz.apigw.tencentcs.com/release/

15s » dashboard » done
```

注意事项

1. 首次部署成功后，也可以通过以下命令，在本地运行服务，并与后端腾讯云服务进行通讯：

```
$ cd dashboard && npm run start
```

2. 目前暂不支持淘宝等第三方 npm 源，如报错 `Component "@serverless/tencent-express" was not found on NPM nor could it be resolved locally.` 请设置并使用 npm 官方源体验：

```
$ npm config rm registry
$ npm set registry https://registry.npmjs.org/
```

3. 腾讯云 Component 已支持二维码一键登录，如您希望使用配置密钥的方式登录，也可以参考如下步骤：
   在 `tencent-fullstack-application` 文件夹根目录创建 `.env` 文件：

```
$ touch .env # 腾讯云的配置信息
```

   在 `.env` 文件中配置腾讯云的 SecretId 和 SecretKey 信息并保存：

```
# .env
TENCENT_SECRET_ID=123
TENCENT_SECRET_KEY=123
```

说明：

- 如果没有腾讯云账号，请先 注册新账号。
- 如果已有腾讯云账号，可以在 API 密钥管理 中获取 SecretId 和 SecretKey。

快速部署一个全栈应用（React.js+Express.js）

最近更新时间：2019-12-13 15:20:23

操作场景

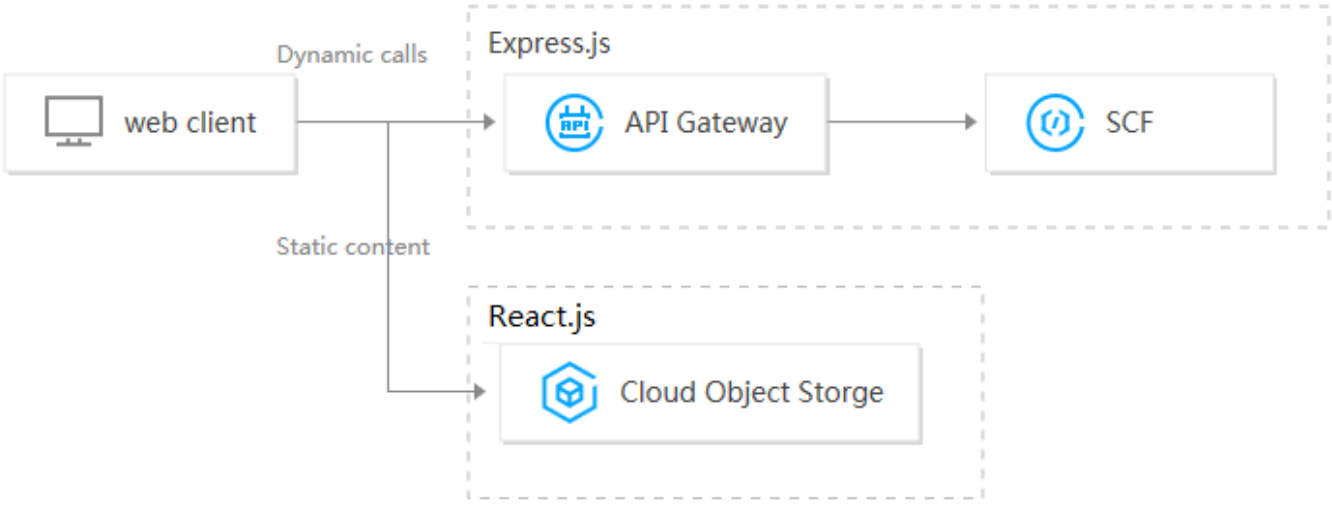
全栈组件（React.js+Express.js）用于通过多个 Serverless Components 部署 Serverless 全栈应用程序。可以帮助开发者更方便快捷的部署 Serverless 应用，例如利用后端 API 与前端 React.js 结合等场景。

此项目完全基于腾讯云 Serverless 服务器，可很大程度的缩减使用成本。 如果您正在寻找一个低开销的便捷轻量的 Serverless 服务管理框架，这里将是很好的选择。

该示例包括：

- **serverless REST API**：由腾讯云 Servelress Cloud Function（无服务云函数 SCF） 和腾讯云 API Gateway 提供相关能力，帮助开发者架构自己的项目和路由。
- **serverless React.js 站点**：由腾讯云 Cloud Object Storage（对象存储 COS）提供相关存储能力. 通过后端 API 传递到前端，并使用 React.js 做相关渲染。

该全栈 Web 应用架构图如下：



操作步骤

安装

1. 通过如下命令安装 Serverless Framework：

```
$ npm i -g serverless
```

2. 新建一个空的文件夹，使用 create --template-url 安装相关 template。

```
$ serverless create --template-url https://github.com/serverless/components/tree/master/templates/tencent-fullstack-react-application
```

3. 使用 cd 命令，进入 templates\tencent-fullstack-react-application 文件夹，可以查看到如下目录结构：

```
|- api
|- dashboard
|- serverless.yml # 使用项目中的 yml 文件
```

4. 分别在 dashboard 和 api 两个文件目录执行 NPM 依赖的安装，如下命令所示：

```
$ cd dashboard
$ npm i
```

```
$ cd api
$ npm i
```

部署

回到 tencent-fullstack-react-application 目录下，直接通过 serverless 命令部署应用：

```
$ serverless
```

如果希望查看部署详情，可以通过调试模式的命令 serverless --debug 进行部署。

如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

部署成功后，可以直接在浏览器中访问日志中返回的 Dashboard URL 地址，查看该全栈 Web App 的效果：

```
dashboard:
url: https://jcwm1l-myappid.cos-website.ap-guangzhou.myqcloud.com
env:
apiUrl: https://service-id-myappid.gz.apigw.tencentcs.com/release/
api:
region: undefined
functionName: tencent-fullstack-api
apiGatewayServiceId: service-id
url: https://service-id-myappid.gz.apigw.tencentcs.com/release/
```

```
15s » dashboard » done
```

注意事项

1. 首次部署成功后，也可以通过以下命令，在本地运行服务，并与后端腾讯云服务进行通讯：

```
$ cd dashboard && npm run start
```

2. 目前暂不支持淘宝等第三方 npm 源，如报错 `Component "@serverless/tencent-express" was not found on NPM nor could it be resolved locally.` 请设置并使用 npm 官方源体验：

```
$ npm config rm registry
$ npm set registry https://registry.npmjs.org/
```

3. 腾讯云 Component 已支持二维码一键登录，如您希望使用配置密钥的方式登录，也可以参考如下步骤：
   在 `tencent-fullstack-react-application` 文件夹根目录创建 `.env` 文件：

```
$ touch .env # 腾讯云的配置信息
```

   在 `.env` 文件中配置腾讯云的 SecretId 和 SecretKey 信息并保存。

```
# .env
TENCENT_SECRET_ID=123
TENCENT_SECRET_KEY=123
```

> 说明：
>
> - 如果没有腾讯云账号，请先 注册新账号。
> - 如果已有腾讯云账号，可以在 API 密钥管理 中获取 SecretId 和 SecretKey。

快速部署 Egg.js 框架

最近更新时间：2020-03-05 19:38:03

操作场景

腾讯云 Egg.js Serverless Component，支持 Restful API 服务的部署。

前提条件

初始化 Egg 项目

```
$ mkdir egg-example && cd egg-example
$ npm init egg --type=simple
$ npm i
```

新增初始化文件

在项目根目录下新建 sls.js 文件，内容如下：

```
const { Application } = require('egg')

const app = new Application({
env: 'prod'
})

module.exports = app
```

修改 Egg 配置

由于云函数在执行时，只有 /tmp 可读写的，所以我们需要将 egg.js 框架运行尝试的日志写到该目录下，为此需要修改 config/config.default.js 中的配置如下：

```
const config = exports = {
env: 'prod', // 推荐云函数的 egg 运行环境变量修改为 prod
rundir: '/tmp',
logger: {
dir: '/tmp',
},
};
```

注意事项

由于 egg 的 egg-static 静态资源插件是默认开启的，所以在启动应用时，会尝试创建 app/public 目录，但是云函数执行环境只有 /tmp 可读写，所以需要本地创建，并添加 .gitkeep 文件（内容为空）。

如果您不需要使用静态资源，可以通过修改 config/plugin.js 禁用静态资源功能：

```
module.exports = {
static: {
enable: false
}
}
```

如果您需要开启静态资源功能，并且 public 已经存在，且里面包含静态资源。此时需要配置 binaryTypes ，修改 sls.js 文件如下：

```
const { Application } = require('egg')

const app = new Application({
env: 'prod'
})

// 这里可以根据实际情况来配置
// 如果您的站点开启 gzip，那么所有返回类型都应该是二进制类型，所以应该是 app.binaryTypes = ['*/*']
app.binaryTypes = ['image/*']

module.exports = app
```

操作步骤

安装

通过 npm 全局安装 Serverless CLI：

```
$ npm install -g serverless
```

配置

在项目根目录创建 serverless.yml 文件：

```
$ touch serverless.yml
```

在 serverless.yml 文件中进行如下配置：

```
# serverless.yml

MyComponent:
component: "@serverless/tencent-egg"
inputs:
region: ap-beijing
functionName: egg-function
code: ./
functionConf:
timeout: 10
memorySize: 128
environment:
variables:
TEST: vale
vpcConfig:
subnetId: ''
vpcId: ''
apigatewayConf:
protocols:
- https
environment: release
```

查看详细配置文档 >>

部署

如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

通过 `sls` 命令进行部署，并可以添加 `--debug` 参数查看部署过程中的信息：

> 说明：
> `sls` 是 `serverless` 命令的简写。

```
$ sls --debug

DEBUG ─ Resolving the template's static variables.
DEBUG ─ Collecting components from the template.
DEBUG ─ Downloading any NPM components found in the template.
DEBUG ─ Analyzing the template's components dependencies.
DEBUG ─ Creating the template's components graph.
DEBUG ─ Syncing template state.
DEBUG ─ Executing the template's components graph.
DEBUG ─ Compressing function egg-function file to /Users/tina/Desktop/live/egg-proj/.serverless/eg
```

```
g-function.zip.
DEBUG — Compressed function egg-function file successful
DEBUG — Uploading service package to cos[sls-cloudfunction-ap-beijing-code]. sls-cloudfunction-default-egg-function-1581335565.zip
DEBUG — Uploaded package successful /Users/tina/Desktop/live/egg-proj/.serverless/egg-function.zip
DEBUG — Creating function egg-function
DEBUG — Updating code...
DEBUG — Updating configure...
DEBUG — Created function egg-function successful
DEBUG — Setting tags for function egg-function
DEBUG — Creating trigger for function egg-function
DEBUG — Deployed function egg-function successful
DEBUG — Starting API-Gateway deployment with name MyComponent.TencentApiGateway in the ap-beijing region
DEBUG — Service with ID service-n5m5e8x3 created.
DEBUG — API with id api-cmkhknda created.
DEBUG — Deploying service with id service-n5m5e8x3.
DEBUG — Deployment successful for the api named MyComponent.TencentApiGateway in the ap-beijing region.

MyComponent:
region: ap-beijing
functionName: egg-function
apiGatewayServiceId: service-n5m5e8x3
url: https://service-n5m5e8x3-1251971143.bj.apigw.tencentcs.com/release/

32s › MyComponent › done
```

移除

通过以下命令移除部署的 API 网关和云函数：

```
$ sls remove --debug

DEBUG — Flushing template state and removing all components.
DEBUG — Removing function
DEBUG — Request id
DEBUG — Removed function egg-function successful
DEBUG — Removing any previously deployed API. api-cmkhknda
DEBUG — Removing any previously deployed service. service-n5m5e8x3

8s › MyComponent › done
```

账号配置（可选）

当前默认支持 CLI 扫描二维码登录，如您希望配置持久的环境变量/密钥信息，也可以本地创建 .env 文件：

```
$ touch .env # 腾讯云的配置信息
```

在 .env 文件中配置腾讯云的 SecretId 和 SecretKey 信息并保存：

```
# .env
TENCENT_SECRET_ID=123
TENCENT_SECRET_KEY=123
```

说明：

- 如果没有腾讯云账号，请先 注册新账号。
- 如果已有腾讯云账号，可以在 API 密钥管理 中获取 SecretId 和 SecretKey。

快速部署 Next.js 框架

最近更新时间：2020-03-05 19:38:43

操作场景

腾讯云 Next.js Serverless Component，支持 Restful API 服务的部署。

前提条件

初始化 Next.js 项目

```
$ npm init next-app
```

新增初始化文件

在项目根目录下新建 sls.js 文件，内容如下：

```
const express = require('express')
const next = require('next')

const app = next({ dev: false })
const handle = app.getRequestHandler()

async function creatServer() {
await app.prepare()
const server = express()

server.all('*', (req, res) => {
return handle(req, res)
})

// 定义是否返回 base64 编码的文件 mime 类型。默认是所有文件，因为 next.js 默认 build 开启 gzip.
// 如果需要修改，请先理解 gzip 的文件编码方式。
server.binaryTypes = ['*/*']

return server
}

module.exports = creatServer
```

添加 express 依赖：

```
$ npm i express --save
```

> 说明：
> 这里通过 express 服务来代理 next.js 的服务。

操作步骤

安装

通过 npm 全局安装 Serverless CLI：

```
$ npm install -g serverless
```

配置

1. 在项目根目录创建 serverless.yml 文件：

   ```
   $ touch serverless.yml
   ```

2. 在 serverless.yml 中进行如下配置：

   ```
   # serverless.yml
   NextjsFunc:
   component: '@serverless/tencent-nextjs'
   inputs:
   functionName: nextjs-function
   region: ap-guangzhou
   code: ./
   functionConf:
   timeout: 30
   memorySize: 128
   environment:
   variables:
   RUN_ENV: test
   apigatewayConf:
   protocols:
   - http
   - https
   environment: release
   ```

查看详细配置文档 >>

部署

您可以通过下面步骤进行部署：

1. 构建静态资源

```
$ npm run build
```

2. 部署到云端

如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

通过 `sls` 命令进行部署，并可以添加 `--debug` 参数查看部署过程中的信息：

> 说明：
> `sls` 是 `serverless` 命令的简写。

```
$ sls --debug

DEBUG ─ Resolving the template's static variables.
DEBUG ─ Collecting components from the template.
DEBUG ─ Downloading any NPM components found in the template.
DEBUG ─ Analyzing the template's components dependencies.
DEBUG ─ Creating the template's components graph.
DEBUG ─ Syncing template state.
DEBUG ─ Executing the template's components graph.
DEBUG ─ Compressing function nextjs-function file to /Users/yugasun/Desktop/Develop/serverless/tencent-nextjs/example/.serverless/nextjs-function.zip.
DEBUG ─ Compressed function nextjs-function file successful
DEBUG ─ Uploading service package to cos[sls-cloudfunction-ap-guangzhou-code]. sls-cloudfunction-default-nextjs-function-1582430808.zip
DEBUG ─ Uploaded package successful /Users/yugasun/Desktop/Develop/serverless/tencent-nextjs/example/.serverless/nextjs-function.zip
DEBUG ─ Creating function nextjs-function
DEBUG ─ Updating code...
DEBUG ─ Updating configure...
DEBUG ─ Created function nextjs-function successful
DEBUG ─ Setting tags for function nextjs-function
DEBUG ─ Creating trigger for function nextjs-function
DEBUG ─ Deployed function nextjs-function successful
```

```
DEBUG – Starting API-Gateway deployment with name NextjsFunc.TencentApiGateway in the ap-gua
ngzhou region
DEBUG – Using last time deploy service id service-32okcrfq
DEBUG – Updating service with serviceId service-32okcrfq.
DEBUG – Endpoint ANY / already exists with id api-5242vfgi.
DEBUG – Updating api with api id api-5242vfgi.
DEBUG – Service with id api-5242vfgi updated.
DEBUG – Deploying service with id service-32okcrfq.
DEBUG – Deployment successful for the api named NextjsFunc.TencentApiGateway in the ap-guang
zhou region.

NextjsFunc:
region: ap-guangzhou
functionName: nextjs-function
apiGatewayServiceId: service-32okcrfq
url: https://service-32okcrfq-1251556596.gz.apigw.tencentcs.com/release/

34s › NextjsFunc › done
```

移除

通过以下命令移除部署的服务：

```
$ sls remove --debug

DEBUG – Flushing template state and removing all components.
DEBUG – Removing function
DEBUG – Request id
DEBUG – Removed function nextjs-function successful
DEBUG – Removing any previously deployed API. api-5242vfgi
DEBUG – Removing any previously deployed service. service-32okcrfq

11s › NextjsFunc › done
```

账号配置（可选）

当前默认支持 CLI 扫描二维码登录，如您希望配置持久的环境变量/密钥信息，也可以本地创建 `.env` 文件：

```
$ touch .env # 腾讯云的配置信息
```

在 `.env` 文件中配置腾讯云的 SecretId 和 SecretKey 信息并保存：

```
# .env
TENCENT_SECRET_ID=123
```

```
TENCENT_SECRET_KEY=123
```

说明：

- 如果没有腾讯云账号，请先 注册新账号。
- 如果已有腾讯云账号，可以在 API 密钥管理 中获取 SecretId 和 SecretKey。

快速部署 Koa 框架

最近更新时间：2019-12-27 18:42:03

操作场景

Koa 组件通过使用 serverless-tencent 的基础组件（如 API 网关组件、SCF 组件等），可以帮助我们快速、方便的在腾讯云创建、配置和管理一个 Koa 框架。

操作步骤

通过 Koa 组件，对一个 Koa 应用进行完整的创建、配置、部署和删除等操作。支持命令如下：

安装

通过 npm 安装 Serverless：

```
$ npm install -g serverless
```

创建

1. 本地创建 `serverless.yml` 文件和 `app.js` 文件：

   ```
   $ touch serverless.yml
   ```

2. 初始化一个新的 npm 包，并安装 Koa：

   ```
   npm init # 创建后持续回车
   npm i --save koa # 安装 koa
   ```

3. 创建一个 `app.js` 文件，

   ```
   $ touch app.js
   ```

4. 在 `app.js` 文件中创建您的 Koa App：

   ```
   const koa = require('koa');
   const app = new koa();
   app.use(async (ctx, next) => {
   if (ctx.path !== '/') return next();
   ctx.body = 'Hello from Koa';
   });
   ```

```
// don't forget to export!
module.exports = app;
```

配置

在 serverless.yml 中进行如下配置：

```
# serverless.yml

koa:
component: '@serverless/tencent-koa'
inputs:
region: ap-shanghai
```

部署

如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

通过 `sls` 命令进行部署，并可以添加 `--debug` 参数查看部署过程中的信息。

> 说明：
> `sls` 命令是 `serverless` 命令的缩写

```
$ sls --debug

DEBUG ─ Resolving the template's static variables.
DEBUG ─ Collecting components from the template.
DEBUG ─ Downloading any NPM components found in the template.
DEBUG ─ Analyzing the template's components dependencies.
DEBUG ─ Creating the template's components graph.
DEBUG ─ Syncing template state.
DEBUG ─ Executing the template's components graph.
DEBUG ─ Compressing function KoaComponent_7xRrrd file to /Users/dfounderliu/Desktop/temp/code/.serverless/KoaComponent_7xRrrd.zip.
DEBUG ─ Compressed function KoaComponent_7xRrrd file successful
DEBUG ─ Uploading service package to cos[sls-cloudfunction-ap-shanghai-code]. sls-cloudfunction-default-KoaComponent_7xRrrd-1572512568.zip
DEBUG ─ Uploaded package successful /Users/dfounderliu/Desktop/temp/code/.serverless/KoaComponent_7xRrrd.zip
DEBUG ─ Creating function KoaComponent_7xRrrd
DEBUG ─ Created function KoaComponent_7xRrrd successful
DEBUG ─ Starting API-Gateway deployment with name koa.TencentApiGateway in the ap-shanghai r
```

```
egion
DEBUG - Using last time deploy service id service-n0vs2ohb
DEBUG - Updating service with serviceId service-n0vs2ohb.
DEBUG - Endpoint ANY / already exists with id api-9z60urs4.
DEBUG - Updating api with api id api-9z60urs4.
DEBUG - Service with id api-9z60urs4 updated.
DEBUG - Deploying service with id service-n0vs2ohb.
DEBUG - Deployment successful for the api named koa.TencentApiGateway in the ap-shanghai regi
on.

koa:
region: ap-shanghai
functionName: KoaComponent_7xRrrd
apiGatewayServiceId: service-n0vs2ohb
url: http://service-n0vs2ohb-1300415943.ap-shanghai.apigateway.myqcloud.com/release/

36s › koa › done
```

部署完毕后，可以在浏览器中访问返回的链接，看到对应的 Koa 返回值。

移除

通过以下命令移除部署的存储桶：

```
$ sls remove --debug

DEBUG - Flushing template state and removing all components.
DEBUG - Removed function KoaComponent_MHrAzr successful
DEBUG - Removing any previously deployed API. api-kf2hxrhc
DEBUG - Removing any previously deployed service. service-n0vs2ohb

13s › koa › done
```

账号配置（可选）

当前默认支持 CLI 扫描二维码登录，如您希望配置持久的环境变量/密钥信息，也可以本地创建 `.env` 文件：

```
$ touch .env # 腾讯云的配置信息
```

在 `.env` 文件中配置腾讯云的 SecretId 和 SecretKey 信息并保存：

```
# .env
TENCENT_SECRET_ID=123
TENCENT_SECRET_KEY=123
```

说明：

- 如果没有腾讯云账号，请先 注册新账号。
- 如果已有腾讯云账号，可以在 API 密钥管理中获取 SecretId 和 SecretKey。

部署 PHP Laravel

最近更新时间：2020-03-04 16:28:22

操作场景

腾讯云 Laravel Serverless Component，支持 Restful API 服务的部署。

前提条件

初始化 Laravel 项目

在使用此组件之前，您需要先初始化一个 `laravel` 项目：

```
php composer create-project --prefer-dist laravel/laravel serverless-laravel
```

> 注意：
> Laravel 使用 Composer 管理依赖，所以您需要先自行安装 Composer，请参考 官方安装文档。

修改 Laravel 项目

由于云函数在执行时，只有 `/tmp` 可读写的，所以我们需要将 `laravel` 框架运行时的 `storage` 目录写到该目录下，为此需要修改 `bootstrap/app.php` 文件，在 `$app = new Illuminate\Foundation\Application` 后添加：

```
$app->useStoragePath($_ENV['APP_STORAGE'] ?? $app->storagePath());
```

然后在根目录下的 `.env` 文件中新增如下配置：

```
# 视图文件编译路径
VIEW_COMPILED_PATH=/tmp/storage/framework/views

# 由于是无服务函数，所以没法存储 session 在硬盘上，如果不需要 sessions，可以使用 array
# 如果需要您可以将 session 存储到 cookie 或者数据库中
SESSION_DRIVER=array

# 建议将错误日志输出到控制台，方便云端去查看
LOG_CHANNEL=stderr
```

```
# 应用的 storage 目录必须为 /tmp
APP_STORAGE=/tmp
```

操作步骤

安装

通过 npm 全局安装 serverless cli：

```
$ npm install -g serverless
```

配置

在项目根目录，创建 `serverless.yml` 文件：

```
$ touch serverless.yml
```

在 `serverless.yml` 中进行如下配置：

```
# serverless.yml

MyComponent:
component: "@serverless/tencent-laravel"
inputs:
region: ap-guangzhou
functionName: laravel-function
code: ./
functionConf:
timeout: 10
memorySize: 128
environment:
variables:
TEST: vale
vpcConfig:
subnetId: ''
vpcId: ''
apigatewayConf:
protocol: https
environment: release
```

查看详细配置文档 >>

部署

如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

通过 `sls` 命令进行部署，并添加 `--debug` 参数查看部署过程中的信息：

> 说明：
> `sls` 是 `serverless` 命令的简写。

```
$ sls --debug

DEBUG ─ Resolving the template's static variables.
DEBUG ─ Collecting components from the template.
DEBUG ─ Downloading any NPM components found in the template.
DEBUG ─ Analyzing the template's components dependencies.
DEBUG ─ Creating the template's components graph.
DEBUG ─ Syncing template state.
DEBUG ─ Executing the template's components graph.
DEBUG ─ Compressing function laravel-function file to /Users/Downloads/serverless-laravel/.serverless/laravel-function.zip.
DEBUG ─ Compressed function laravel-function file successful
DEBUG ─ Uploading service package to cos[sls-cloudfunction-ap-guangzhou-code]. sls-cloudfunction-default-laravel-function-1581888194.zip
DEBUG ─ Uploaded package successful /Users/Downloads/serverless-laravel/.serverless/laravel-function.zip
DEBUG ─ Creating function laravel-function
DEBUG ─ Created function laravel-function successful
DEBUG ─ Setting tags for function laravel-function
DEBUG ─ Creating trigger for function laravel-function
DEBUG ─ Deployed function laravel-function successful
DEBUG ─ Starting API-Gateway deployment with name MyComponent.TencentApiGateway in the ap-guangzhou region
DEBUG ─ Service with ID service-ok334ism created.
DEBUG ─ API with id api-l7cppn6s created.
DEBUG ─ Deploying service with id service-ok334ism.
DEBUG ─ Deployment successful for the api named MyComponent.TencentApiGateway in the ap-guangzhou region.

MyComponent:
region: ap-guangzhou
functionName: laravel-function
apiGatewayServiceId: service-ok334ism
url: http://service-ok334ism-1258834142.gz.apigw.tencentcs.com/release/
```

```
192s › MyComponent › done
```

移除

通过以下命令移除部署的服务：

```
$ sls remove --debug

DEBUG ─ Flushing template state and removing all components.
DEBUG ─ Removing function
DEBUG ─ Request id
DEBUG ─ Removed function laravel-function successful
DEBUG ─ Removing any previously deployed API. api-l7cppn6s
DEBUG ─ Removing any previously deployed service. service-ok334ism

18s › MyComponent › done
```

账号配置（可选）

当前默认支持 CLI 扫描二维码登录，如您希望配置持久的环境变量/密钥信息，也可以本地创建 `.env` 文件：

在 `.env` 文件中配置腾讯云的 SecretId 和 SecretKey 信息并保存：

```
# .env
TENCENT_SECRET_ID=123
TENCENT_SECRET_KEY=123
```

说明：

- 如果没有腾讯云账号，可以在此 注册新账号。
- 如果已有腾讯云账号，可以在 API 密钥管理 中获取 SecretId 和 SecretKey。

部署 Python Flask

最近更新时间：2020-03-04 16:25:39

操作场景

腾讯云 Flask Serverless Component，支持 Restful API 服务的部署，不支持 Flask Command。

> 说明：
> 任何支持 WSGI（Web Server Gateway Interface，即 Web 服务器网关接口）的 Python 服务端框架都可以通过该组件进行部署，例如 Falcon 框架等。

前提条件

在使用此组件之前，需要先初始化一个 Flask 项目，具体步骤如下：

Flask 安装与配置

您可以在全局或虚拟环境中完成 Flask 安装。

1. 创建项目目录：

```
$ mkdir myapp
$ cd myapp
```

2. 在项目目录中生成依赖文件 `requiements.txt` ：

```
$ pip freeze > requirements.txt
```

3. 将 `Flask` 和 `werkzeug` 添加到依赖文件 `requiements.txt` 中：

```
Flask==1.0.2
werkzeug==0.16.0
```

4. 完成 Flask 安装：

```
$ pip install -r requirements.txt
```

创建应用服务

新增 API 服务 `app.py` ，以下代码仅供参考：

```
from flask import Flask, jsonify
app = Flask(__name__)

@app.route("/")
def index():
return "Hello Flask"

@app.route("/users")
def users():
users = [{'name': 'test1'}, {'name': 'test2'}]
return jsonify(data=users)

@app.route("/users/<id>")
def user(id):
return jsonify(data={'name': 'test1'})
```

操作步骤

安装

通过 npm 全局安装 Serverless CLI：

```
$ npm install -g serverless
```

配置

本地创建 `serverless.yml` 文件：

```
$ touch serverless.yml
```

在 `serverless.yml` 文件中进行如下配置：

```
# serverless.yml

MyComponent:
component: "@serverless/tencent-flask"
inputs:
region: ap-guangzhou
functionName: flask-function
code: ./
```

```
functionConf:
timeout: 10
memorySize: 128
environment:
variables:
TEST: vale
vpcConfig:
subnetId: ''
vpcId: ''
apigatewayConf:
protocol: https
environment: release
```

查看详细配置文档 >>

部署

如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

通过 `sls` 命令进行部署，并添加 `--debug` 参数查看部署过程中的信息。

> 说明：
> `sls` 是 `serverless` 命令的简写。

```
$ sls --debug

DEBUG ─ Resolving the template's static variables.
DEBUG ─ Collecting components from the template.
DEBUG ─ Downloading any NPM components found in the template.
DEBUG ─ Analyzing the template's components dependencies.
DEBUG ─ Creating the template's components graph.
DEBUG ─ Syncing template state.
DEBUG ─ Executing the template's components graph.
DEBUG ─ Generated requirements from /Users/Downloads/myapp/requirements.txt in /Users/Downloads/myapp/.serverless/requirements.txt...
DEBUG ─ Installing requirements from /Users/Library/Caches/serverless-python-requirements/78f1b71bd84112bad004679c938bd8c63c80ddf8c468c5484e3de8214a52a8bc_slspyc/requirements.txt ...
DEBUG ─ Using download cache directory /Users/Library/Caches/serverless-python-requirements/downloadCacheslspyc
DEBUG ─ Running ...
DEBUG ─ Compressing function flask-function file to /Users/Downloads/myapp/.serverless/flask-function.zip.
DEBUG ─ Compressed function flask-function file successful
```

```
DEBUG - Uploading service package to cos[sls-cloudfunction-ap-guangzhou-code]. sls-cloudfunctio
n-default-flask-function-1581890739.zip
DEBUG - Uploaded package successful /Users/Downloads/myapp/.serverless/flask-function.zip
DEBUG - Creating function flask-function
DEBUG - Created function flask-function successful
DEBUG - Setting tags for function flask-function
DEBUG - Creating trigger for function flask-function
DEBUG - Deployed function flask-function successful
DEBUG - Starting API-Gateway deployment with name MyComponent.TencentApiGateway in the ap-
guangzhou region
DEBUG - Service with ID service-rf5pzbfy created.
DEBUG - API with id api-6i3uv432 created.
DEBUG - Deploying service with id service-rf5pzbfy.
DEBUG - Deployment successful for the api named MyComponent.TencentApiGateway in the ap-gu
angzhou region.

MyComponent:
region: ap-guangzhou
functionName: flask-function
apiGatewayServiceId: service-rf5pzbfy
url: http://service-rf5pzbfy-1258834142.gz.apigw.tencentcs.com/release/

75s › MyComponent › done
```

移除

通过以下命令移除部署的服务：

```
$ sls remove --debug

DEBUG - Flushing template state and removing all components.
DEBUG - Removing function
DEBUG - Request id
DEBUG - Removed function flask-function successful
DEBUG - Removing any previously deployed API. api-6i3uv432
DEBUG - Removing any previously deployed service. service-rf5pzbfy

17s › MyComponent › done
```

账号配置（可选）

当前默认支持 CLI 扫描二维码登录，如您希望配置持久的环境变量/密钥信息，也可以本地创建 `.env` 文件：

```
$ touch .env # 腾讯云的配置信息
```

在 .env 文件中配置腾讯云的 SecretId 和 SecretKey 信息并保存：

```
# .env
TENCENT_SECRET_ID=123
TENCENT_SECRET_KEY=123
```

说明：

- 如果没有腾讯云账号，请先 注册新账号。
- 如果已有腾讯云账号，可以在 API 密钥管理 中获取 SecretId 和 SecretKey。

部署 Python Bottle

最近更新时间：2020-03-05 19:39:11

操作场景

腾讯云 Bottle Serverless Component，支持 Restful API 服务的部署。

前提条件

1. 安装 Bottle，新建 Python 文件，例如 app.py ：

```
from bottle import route, run, template

@route('/hello/<name>')
def index(name):
return template('<b>Hello {{name}}</b>!', name=name)
```

2. 将 Python 所需要的依赖安装到项目目录，例如本实例需要 bottle ，所以可以通过 pip 进行安装：

```
pip install bottle -t ./
```

如果因为网络问题，可以考虑使用国内源，例如：

```
pip install bottle -t ./ -i https://pypi.tuna.tsinghua.edu.cn/simple
```

操作步骤

安装

通过 npm 全局安装 Serverless CLI：

```
$ npm install -g serverless
```

配置

在本地创建 serverless.yml 文件：

```
$ touch serverless.yml
```

在 serverless.yml 中进行如下配置：

```
BottleTest:
component: '@serverless/tencent-bottle'
inputs:
region: ap-guangzhou
functionName: BottleFunctionTest
code: ./
functionConf:
timeout: 10
memorySize: 256
environment:
variables:
TEST: vale
vpcConfig:
subnetId: ''
vpcId: ''
apigatewayConf:
protocols:
- http
environment: release
```

查看详细配置文档 >>

部署

如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

通过 sls 命令进行部署，并可以添加 --debug 参数查看部署过程中的信息：

> 说明：
> sls 是 serverless 命令的简写。

```
$ sls --debug
```

移除

通过以下命令移除部署的服务：

```
$ sls remove --debug
```

账号配置（可选）

当前默认支持 CLI 扫描二维码登录，如您希望配置持久的环境变量/密钥信息，也可以本地创建 .env 文件：

```
$ touch .env # 腾讯云的配置信息
```

在 .env 文件中配置腾讯云的 SecretId 和 SecretKey 信息并保存：

```
# .env
TENCENT_SECRET_ID=123
TENCENT_SECRET_KEY=123
```

说明：

- 如果没有腾讯云账号，请先 注册新账号。
- 如果已有腾讯云账号，可以在 API 密钥管理 中获取 SecretId 和 SecretKey。

部署 Python Django

最近更新时间：2020-03-05 19:39:23

操作场景

腾讯云 Django Serverless Component，支持 Restful API 服务的部署。

前提条件

1. 新建一个 Django 服务，并通过 Django 创建一个 app（本实践中创建了名为 mytest 的 app）。创建方法请参考 Django 官方文档。
创建后可以查看 view.py 内的信息：

```
from django.shortcuts import render
from django.http import HttpResponse

# Create your views here.

def hello(request):
return HttpResponse("Hello world ! ")
```

2. 增加路由信息：

```
"""mydjango URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
https://docs.djangoproject.com/en/3.0/topics/http/urls/
Examples:
Function views
1. Add an import: from my_app import views
2. Add a URL to urlpatterns: path('', views.home, name='home')
Class-based views
1. Add an import: from other_app.views import Home
2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
Including another URLconf
1. Import the include() function: from django.urls import include, path
2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
"""
# from django.contrib import admin
from django.urls import path
```

```
from mytest.views import hello

urlpatterns = [
path('hello/', hello),
# path('admin/', admin.site.urls),
]
```

3. 对 settings.py 进行修改：

- 注释掉数据库部分，如果有需要可以考虑使用 MySQL 等。
- ALLOWED_HOSTS 部分增加 * ： ALLOWED_HOSTS = ['*']

操作步骤

安装

通过 npm 全局安装 Serverless CLI：

```
$ npm install -g serverless
```

配置

在本地创建 serverless.yml 文件：

```
$ touch serverless.yml
```

在 serverless.yml 中进行如下配置：

```
DjangoTest:
component: '@serverless/tencent-django'
inputs:
region: ap-guangzhou
functionName: DjangoFunctionTest
djangoProjectName: mydjango
code: ./
functionConf:
timeout: 10
memorySize: 256
environment:
variables:
TEST: vale
vpcConfig:
subnetId: ''
vpcId: ''
```

```
apigatewayConf:
protocols:
- http
environment: release
```

> 注意：
> 这里的 djangoProjectName 必须要和项目名称一致。

并将 Python 所需要的依赖安装到项目目录，例如本实例需要 `Django` ，所以可以通过 `pip` 进行安装：

```
pip install Django -t ./
```

如果因为网络问题，可以考虑使用国内源，例如：

```
pip install Django -t ./ -i https://pypi.tuna.tsinghua.edu.cn/simple
```

查看详细配置文档 >>

部署

如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

通过 `sls` 命令进行部署，并可以添加 `--debug` 参数查看部署过程中的信息

```
$ sls --debug
```

移除

通过以下命令移除部署的服务：

```
$ sls remove --debug
```

账号配置（可选）

当前默认支持 CLI 扫描二维码登录，如您希望配置持久的环境变量/密钥信息，也可以本地创建 `.env` 文件：

```
$ touch .env # 腾讯云的配置信息
```

在 `.env` 文件中配置腾讯云的 SecretId 和 SecretKey 信息并保存：

```
# .env
TENCENT_SECRET_ID=123
TENCENT_SECRET_KEY=123
```

说明：

- 如果没有腾讯云账号，请先 注册新账号。
- 如果已有腾讯云账号，可以在 API 密钥管理 中获取 SecretId 和 SecretKey。

部署 Python Pyramid

最近更新时间：2020-03-05 19:39:34

操作场景

腾讯云 Pyramid Serverless Component，支持 Restful API 服务的部署。

前提条件

1. 安装 Pyramid，新建 Python 文件，例如 app.py ：

```
from wsgiref.simple_server import make_server
from pyramid.config import Configurator
from pyramid.response import Response


def hello_world(request):
return Response('Hello World!')


with Configurator() as config:
config.add_route('hello', '/')
config.add_view(hello_world, route_name='hello')
app = config.make_wsgi_app()

if __name__ == "__main__":
server = make_server('0.0.0.0', 6543, app)
server.serve_forever()
```

2. 将 Python 所需要的依赖安装到项目目录，例如本实例需要 Pyramid ，所以可以通过 pip 进行安装：

```
pip install Pyramid -t ./
```

如果因为网络问题，可以考虑使用国内源，例如：

```
pip install Pyramid -t ./ -i https://pypi.tuna.tsinghua.edu.cn/simple
```

操作步骤

安装

通过 npm 全局安装 serverless cli：

```
$ npm install -g serverless
```

配置

本地创建 serverless.yml 文件：

```
$ touch serverless.yml
```

在 serverless.yml 中进行如下配置：

```
PyramidTest:
component: '@serverless/tencent-pyramid'
inputs:
region: ap-guangzhou
functionName: PyramidFunctionTest
pyramidProjectName: myPyramid
code: ./
functionConf:
timeout: 10
memorySize: 256
environment:
variables:
TEST: vale
vpcConfig:
subnetId: ''
vpcId: ''
apigatewayConf:
protocols:
- http
environment: release
```

注意：

- 这里的 pyramidProjectName 必须要和您的项目名称一致。
- pyramidProjectName 实际上是包括您的 app 文件即 app = config.make_wsgi_app() ，并且要确保可以通过 文件名.app 从外引用该文件。

查看详细配置文档 >>

部署

如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

通过 sls 命令进行部署，并可以添加 --debug 参数查看部署过程中的信息：

> 说明：
> sls 是 serverless 命令的简写。

```
$ sls --debug
```

移除

通过以下命令移除部署的服务：

```
$ sls remove --debug
```

账号配置（可选）

当前默认支持 CLI 扫描二维码登录，如您希望配置持久的环境变量/密钥信息，也可以本地创建 .env 文件：

```
$ touch .env # 腾讯云的配置信息
```

在 .env 文件中配置腾讯云的 SecretId 和 SecretKey 信息并保存：

```
# .env
TENCENT_SECRET_ID=123
TENCENT_SECRET_KEY=123
```

> 说明：
>
> - 如果没有腾讯云账号，请先 注册新账号。
> - 如果已有腾讯云账号，可以在 API 密钥管理 中获取 SecretId 和 SecretKey。

部署 Python Tornado

最近更新时间：2020-03-05 19:39:47

操作场景

腾讯云 Tornado Serverless Component，支持 Restful API 服务的部署。目前只支持 Tornado 3.* 及以前版本，以及支持同步操作，不支持异步操作。

前提条件

1. 安装 Tornado，新建 Python文件，例如 app.py ：

```
from tornado.web import RequestHandler

class IndexHandler(RequestHandler):
def get(self):
self.write('hello word tornado')


url = [
(r'/', IndexHandler),
]
```

2. 将 Python 所需要的依赖安装到项目目录，例如本实例需要 Tornado ，所以可以通过 pip 进行安装：

```
pip install Tornado -t ./
```

如果因为网络问题，可以考虑使用国内源，例如：

```
pip install Tornado -t ./ -i https://pypi.tuna.tsinghua.edu.cn/simple
```

操作步骤

安装

通过 npm 全局安装 Serverless CLI：

```
$ npm install -g serverless
```

配置

本地创建 serverless.yml 文件：

```
$ touch serverless.yml
```

在 serverless.yml 中进行如下配置：

```
TornadoTest:
component: '@serverless/tencent-tornado'
inputs:
region: ap-guangzhou
functionName: tornadoFunctionTest
tornadoProjectName: myTornado
code: ./
functionConf:
timeout: 10
memorySize: 256
environment:
variables:
TEST: vale
vpcConfig:
subnetId: ''
vpcId: ''
apigatewayConf:
protocols:
- http
environment: release
```

注意：

- 这里的 tornadoProjectName 必须要和项目名称一致。
- tornadoProjectName 实际上是包括您的 app 文件即路由列表，并且要确保可以通过 文件名.url 从外引用该文件。

查看详细配置文档 >>

部署

如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

通过 sls 命令进行部署，并可以添加 --debug 参数查看部署过程中的信息：

```
$ sls --debug
```

移除

通过以下命令移除部署的服务：

```
$ sls remove --debug
```

账号配置（可选）

当前默认支持 CLI 扫描二维码登录，如您希望配置持久的环境变量/密钥信息，也可以本地创建 .env 文件：

```
$ touch .env # 腾讯云的配置信息
```

在 .env 件中配置腾讯云的 SecretId 和 SecretKey 信息并保存：

```
# .env
TENCENT_SECRET_ID=123
TENCENT_SECRET_KEY=123
```

说明：

- 如果没有腾讯云账号，请先 注册新账号。
- 如果已有腾讯云账号，可以在 API 密钥管理 中获取 SecretId 和 SecretKey。

环境变量管理最佳实践

最近更新时间：2020-03-05 19:38:56

使用 serverless-global 组件进行全局变量管理

在使用 Serverless Components 时，会遇到配置一些全局变量。例如对于多个函数，需要配置相同的数据库等信息时，可以通过如下 serverless-global 组件进行变量的复用和管理。

使用方式如下，首先在 Yaml 中增加全局配置的字段，然后在对应的 Components 中直接通过 `${Conf.mysql_host}` 的方式引用即可。

```
Conf:
component: "serverless-global"
inputs:
mysql_host: gz-cdb-mytest.sql.tencentcdb.com
mysql_user: mytest
mysql_password: mytest
mysql_port: 62580
mysql_db: mytest
mini_program_app_id: mytest
mini_program_app_secret: mytest


Album_Login:
component: "@serverless/tencent-scf"
inputs:
name: Album_Login
codeUri: ./album/login
handler: index.main_handler
runtime: Python3.6
region: ap-shanghai
environment:
variables:
mysql_host: ${Conf.mysql_host}
mysql_port: ${Conf.mysql_port}
mysql_user: ${Conf.mysql_user}
mysql_password: ${Conf.mysql_password}
mysql_db: ${Conf.mysql_db}
```

自定义域名及 HTTPS 访问配置

最近更新时间：2020-03-12 17:16:39

操作场景

通过 Serverless Component 快速构建一个 Serverless Web 网站服务后，如果您希望配置自定义域名及支持 HTTPS 的访问，则可以按照本文提供的两种方案快速配置。

前提条件

- 已经部署了网站服务，获取了 COS/API 网关的网站托管地址。具体部署方法参考 部署 Vue.js+Express.js 全栈应用 或 快速部署 Hexo 博客。
- 已拥有自定义域名（例如 www.example.com），并确保输入的域名已 备案。
- 如果需要 HTTPS 访问，可以申请证书并且 获得证书 ID （例如：certificateId : axE1bo3），个人站点可以直接申请 域名型（DV）免费SSL证书。

方案一：通过 CDN 加速配置支持自定义域名的 HTTPS 访问

配置前，需要确保账号实名并已经 开通 CDN 服务。

增加配置

在 `serverless.yml` 中，增加 CDN 自定义域名配置：

```
myWebsite:
myWebsite:
component: '@serverless/tencent-website'
inputs:
code:
src: ./public # Upload static files generated by HEXO
index: index.html
error: index.html
region: ap-guangzhou
bucketName: my-hexo-bucket
protocol: https
# 新增的 CDN 自定义域名配置
hosts:
- host: www.example.com # 希望配置的自定义域名
```

```
https:
certId: axE1bo3 # SSL 证书 ID
http2: off
httpsType: 4
forceSwitch: -2
```

查看完整配置项说明 >>

部署服务

再次通过 sls 命令进行部署，并可以添加 --debug 参数查看部署过程中的信息。

如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

> 说明：
> sls 是 serverless 命令的简写。

```
$ sls --debug
DEBUG ─ Resolving the template's static variables.
DEBUG ─ Collecting components from the template.
DEBUG ─ Downloading any NPM components found in the template.
DEBUG ─ Analyzing the template's components dependencies.
DEBUG ─ Creating the template's components graph.
DEBUG ─ Syncing template state.
DEBUG ─ Executing the template's components graph.
DEBUG ─ Preparing website Tencent COS bucket my-hexo-bucket-1250000000.
DEBUG ─ Bucket "my-hexo-bucket-1250000000" in the "ap-guangzhou" region already exist.
DEBUG ─ Setting ACL for "my-hexo-bucket-1250000000" bucket in the "ap-guangzhou" region.
DEBUG ─ Ensuring no CORS are set for "my-hexo-bucket-1250000000" bucket in the "ap-guangzhou" region.
DEBUG ─ Ensuring no Tags are set for "my-hexo-bucket-1250000000" bucket in the "ap-guangzhou" region.
DEBUG ─ Configuring bucket my-hexo-bucket-1250000000 for website hosting.
DEBUG ─ Uploading website files from /Users/tina/Documents/hexoblog/hexo/public to bucket my-hexo-bucket-1250000000.
DEBUG ─ Starting upload to bucket my-hexo-bucket-1250000000 in region ap-guangzhou
DEBUG ─ Uploading directory /Users/tina/Documents/hexoblog/hexo/public to bucket my-hexo-bucket-1250000000
DEBUG ─ The CDN domain www.example.com has existed.
DEBUG ─ Updating...
DEBUG ─ Waiting for CDN deploy success..
DEBUG ─ CDN deploy success to host: www.example.com
DEBUG ─ Setup https for www.example.com...
```

```
DEBUG ─ Website deployed successfully to URL: https://my-hexo-bucket-1250000000.cos-website.ap-guangzhou.myqcloud.com.
myWebsite:
url: https://my-hexo-bucket-1250000000.cos-website.ap-guangzhou.myqcloud.com
env:
host:
- https://www.example.com (CNAME: www.example.com.cdn.dnsv1.com )
17s › myWebsite › done
```

添加 CNAME

部署完成后，在命令行的输出中可以查看到一个以 `.cdn.dnsv1.com` 为后缀的 CNAME 域名。参考 CNAME 配置文档，在 DNS 服务商处设置好对应的 CNAME 并生效后，即可访问自定义 HTTPS 域名。

方案二：对 API 网关域名进行自定义域名配置

增加配置

在 `serverless.yml` 中，增加 API 网关自定义域名配置。本文以 egg.js 框架为例，配置如下：

```
# serverless.yml
restApi:
component: "@serverless/tencent-apigateway"
inputs:
region: ap-shanghai
protocols:
- http
- https
serviceName: serverless
environment: release
endpoints:
- path: /users
method: POST
function:
functionName: myFunction
# 增加 API 网关自定义域名配置
customDomain:
- domain: www.example.com
certificateId: axE1bo3
protocols:
- https
```

查看完整配置项说明>>

部署服务

再次通过 sls 命令进行部署，并可以添加 --debug 参数查看部署过程中的信息。
如您的账号未 登录 或 注册 腾讯云，您可以直接通过**微信**扫描命令行中的二维码进行授权登录和注册。

> 说明：
> sls 是 serverless 命令的简写。

```
$ sls --debug
DEBUG ─ Resolving the template's static variables.
DEBUG ─ Collecting components from the template.
DEBUG ─ Downloading any NPM components found in the template.
DEBUG ─ Analyzing the template's components dependencies.
DEBUG ─ Creating the template's components graph.
DEBUG ─ Syncing template state.
DEBUG ─ Executing the template's components graph.
DEBUG ─ Starting API-Gateway deployment with name restApi in the ap-shanghai region
DEBUG ─ Using last time deploy service id service-lqhc88sr
DEBUG ─ Updating service with serviceId service-lqhc88sr.
DEBUG ─ Endpoint POST /users already exists with id api-e902tx1q.
DEBUG ─ Updating api with api id api-e902tx1q.
DEBUG ─ Service with id api-e902tx1q updated.
DEBUG ─ Deploying service with id service-lqhc88sr.
DEBUG ─ Deployment successful for the api named restApi in the ap-shanghai region.
DEBUG ─ Start unbind all exist domain for service service-lqhc88sr
DEBUG ─ Start bind custom domain for service service-lqhc88sr
DEBUG ─ Custom domain for service service-lqhc88sr created successfullly.
DEBUG ─ Please add CNAME record service-lqhc88sr-1250000000.sh.apigw.tencentcs.com for www.example.com.
restApi:
protocols:
- http
- https
subDomain: service-lqhc88sr-1250000000.sh.apigw.tencentcs.com
environment: release
region: ap-shanghai
serviceId: service-lqhc88sr
apis:
-
path: /users
method: POST
apiId: api-e902tx1q
```

```
customDomains:
- www.example.com (CNAME: service-lqhc88sr-1250000000.sh.apigw.tencentcs.com)
8s › restApi › done
```

添加 CNAME 记录

部署完成后，在命令行的输出中可以查看到一个以 `.apigw.tencentcs.com` 为后缀的 CNAME 域名。参考 添加 CNAME 记录，在 DNS 服务商处设置好对应的 CNAME 并生效后，即可访问自定义 HTTPS 域名。