Serverless SSR

操作指南

腾讯云

【联系我们】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100或95716。

文档目录

操作指南
快速部署 Nextjs 框架

最近更新时间：2023-08-28 20:59:32

操作场景

本文将为您指导如何通过 Web Function，将您的本地 Next.js SSR 项目快速部署到云端。

> **说明**
> 本文档主要介绍控制台部署方案，您也可以通过命令行完成部署，详情请参见 通过命令行完成框架部署 。

前提条件

在使用腾讯云云函数服务之前，您需要 注册腾讯云账号 并完成 实名认证 。

操作步骤

模板部署：一键部署 Next.js 项目

1. 登录 Serverless 控制台，单击左侧导航栏的**函数服务**。
2. 在主界面上方选择期望创建函数的地域，并单击**新建**，进入函数创建流程。
3. 选择使用**模板创建**来新建函数，在搜索框里输入 webfunc 筛选函数模板，选择**Next.js 框架模板**并单击**下一步**。如下图所示：

4. 在**新建**页面，您可以查看模板项目的具体配置信息并进行修改。
5. 单击**完成**即可创建函数。函数创建完成后，您可在**函数管理**页面查看 Web 函数的基本信息。
6. 您可以通过 API 网关生成的访问路径 URL，访问您部署的 Next.js 项目。单击左侧菜单栏中的**触发管理**，查看访问路径。如下图所示：

7. 单击访问路径 URL，即可访问服务 Next.js 项目。如下图所示：

> **说明**
> 由于 Nextjs 框架每次部署前需要重新构建，请确保本地更新代码并且重新 `build` 之后再进行部署。

自定义部署：快速迁移本地项目上云

前提条件

本地已安装 Node.js 运行环境。

本地开发

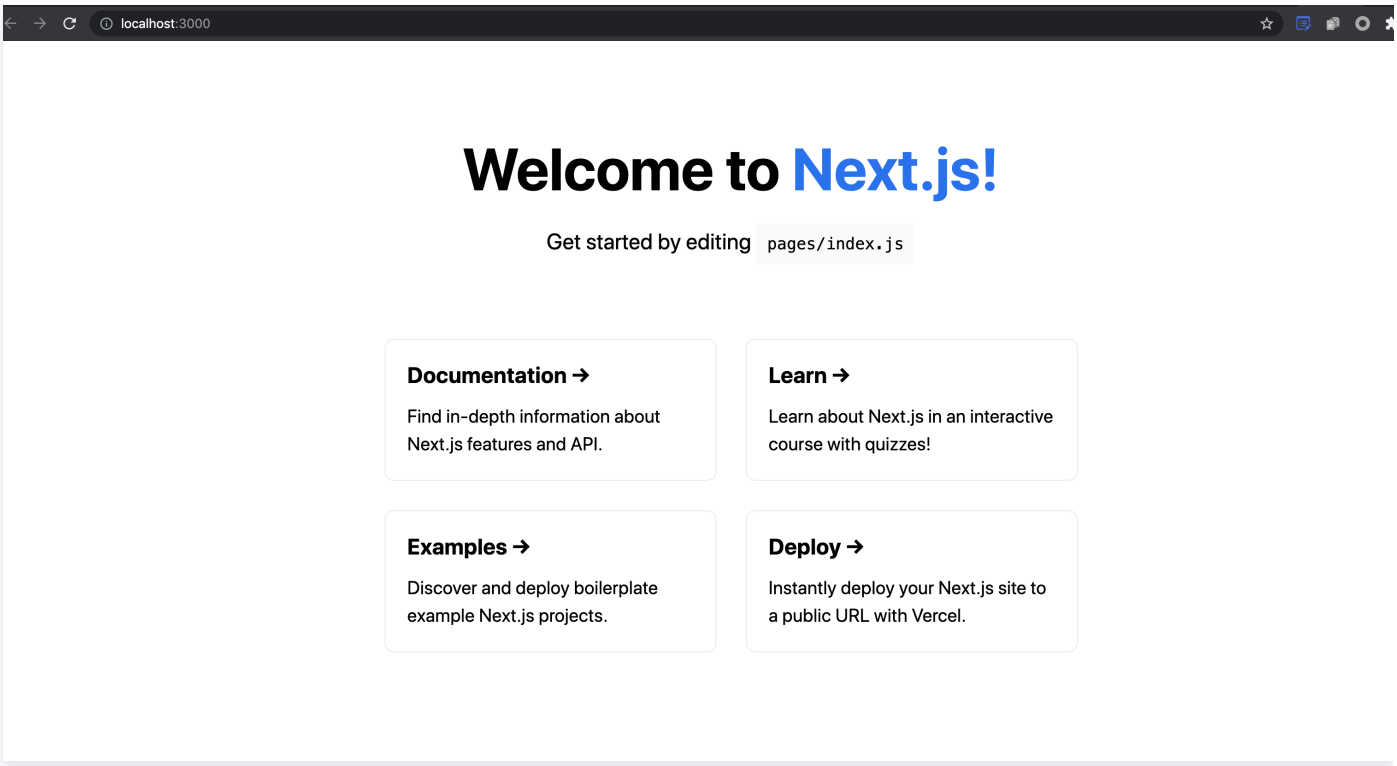
1. 参考 Next.js 官方文档，安装并初始化您的 Next.js 项目：

```
npx create-next-app
```

2. 在根目录下，执行以下命令在本地直接启动服务。

```
cd my-app && npm run dev
```

3. 打开浏览器访问 `http://localhost:3000` ，即可在本地完成 Next.js 示例项目的访问。如下图所示：



部署上云

接下来执行以下步骤，对已初始化的项目进行简单修改，使其可以通过 Web Function 快速部署，此处项目改造通常分为以下两步：

- 修改监听地址与端口为 `0.0.0.0:9000` 。
- 新增 `scf_bootstrap` 启动文件。

具体步骤如下：

1. 在项目根目录下新建 `scf_bootstrap` 启动文件，在该文件添加如下内容（用于启动服务并指定启动端口）：

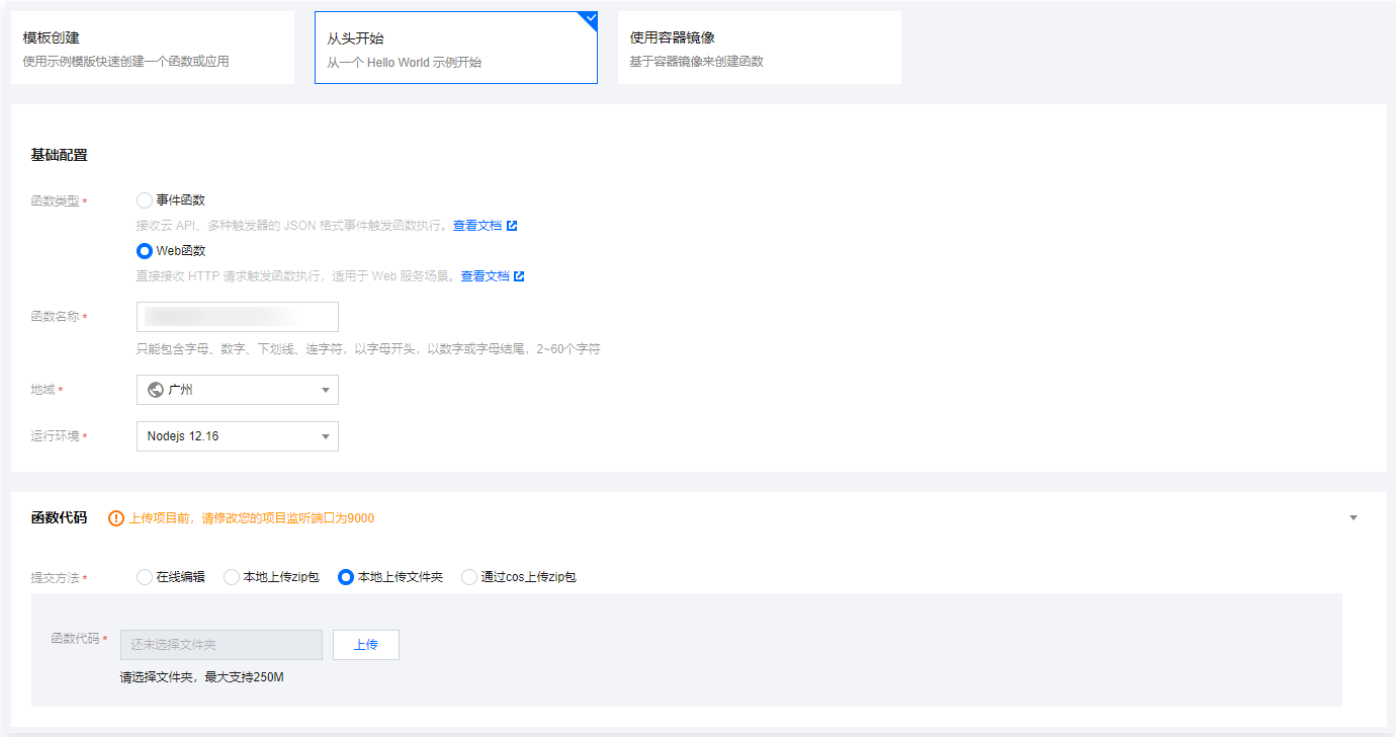
```
#!/var/lang/node12/bin/node
const { nextStart } = require('next/dist/cli/next-start');
nextStart([ '--port', '9000', '--hostname', '0.0.0.0' ])
```

> **注意**
> - 此处仅为示例启动文件，具体请根据您的业务场景进行调整。
> - 示例使用的是云函数标准 Node 环境路径，本地调试时，需修改成您的本地路径。

2. 新建完成后，还需执行以下命令修改文件可执行权限，默认需要 `777` 或 `755` 权限才可正常启动。示例如下：

```
chmod 777 scf_bootstrap
```

3. 登录 Serverless 控制台，单击左侧导航栏的**函数服务**。
4. 在主界面上方选择期望创建函数的地域，并单击**新建**，进入函数创建流程。
5. 选择**从头开始**新建函数，根据页面提示配置相关选项。如下图所示：

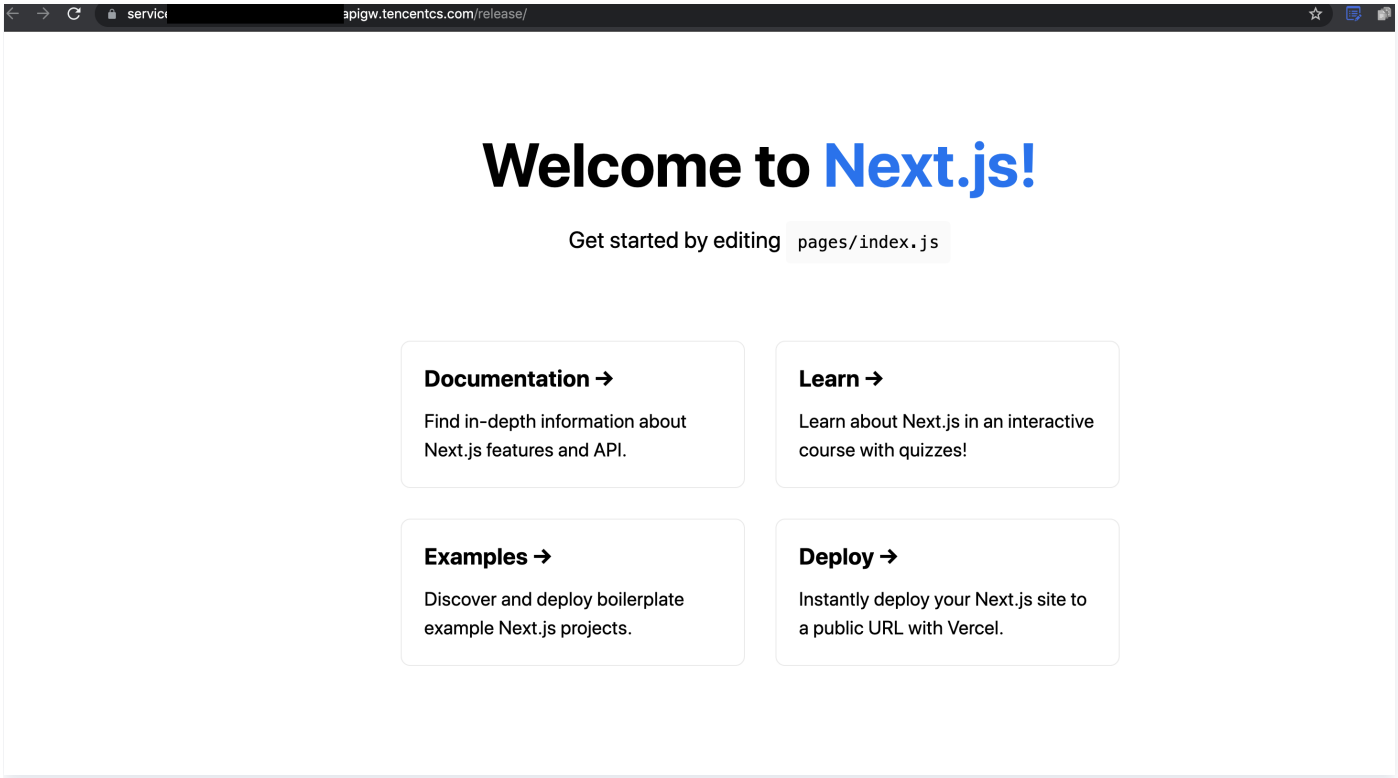


- **函数类型**：选择 “Web 函数”。
- **函数名称**：填写您自己的函数名称。
- **地域**：填写您的函数部署地域，默认为广州。
- **运行环境**：选择 “Nodejs 12.16”。
- **部署方式**：选择“代码部署”，上传您的本地项目。
- **提交方法**：选择“本地上传文件夹”。
- **函数代码**：选择函数代码在本地的具体文件夹。

6. 单击**完成**完成 Next.js 项目的部署。

开发管理

部署完成后，即可在 SCF 控制台快速访问并测试您的 Web 服务，并且体验云函数多项特色功能，例如层绑定、日志管理等，享受 Serverless 架构带来的低成本、弹性扩缩容等优势。



快速部署 Nuxtjs 框架

最近更新时间：2024-03-18 14:38:21

操作场景

本文将为您指导如何通过 Web Function，将您的本地 Nuxt.js SSR 项目快速部署到云端。

> **说明**
> 本文档主要介绍控制台部署方案，您也可以通过命令行完成部署，详情请参见 通过命令行完成框架部署 。

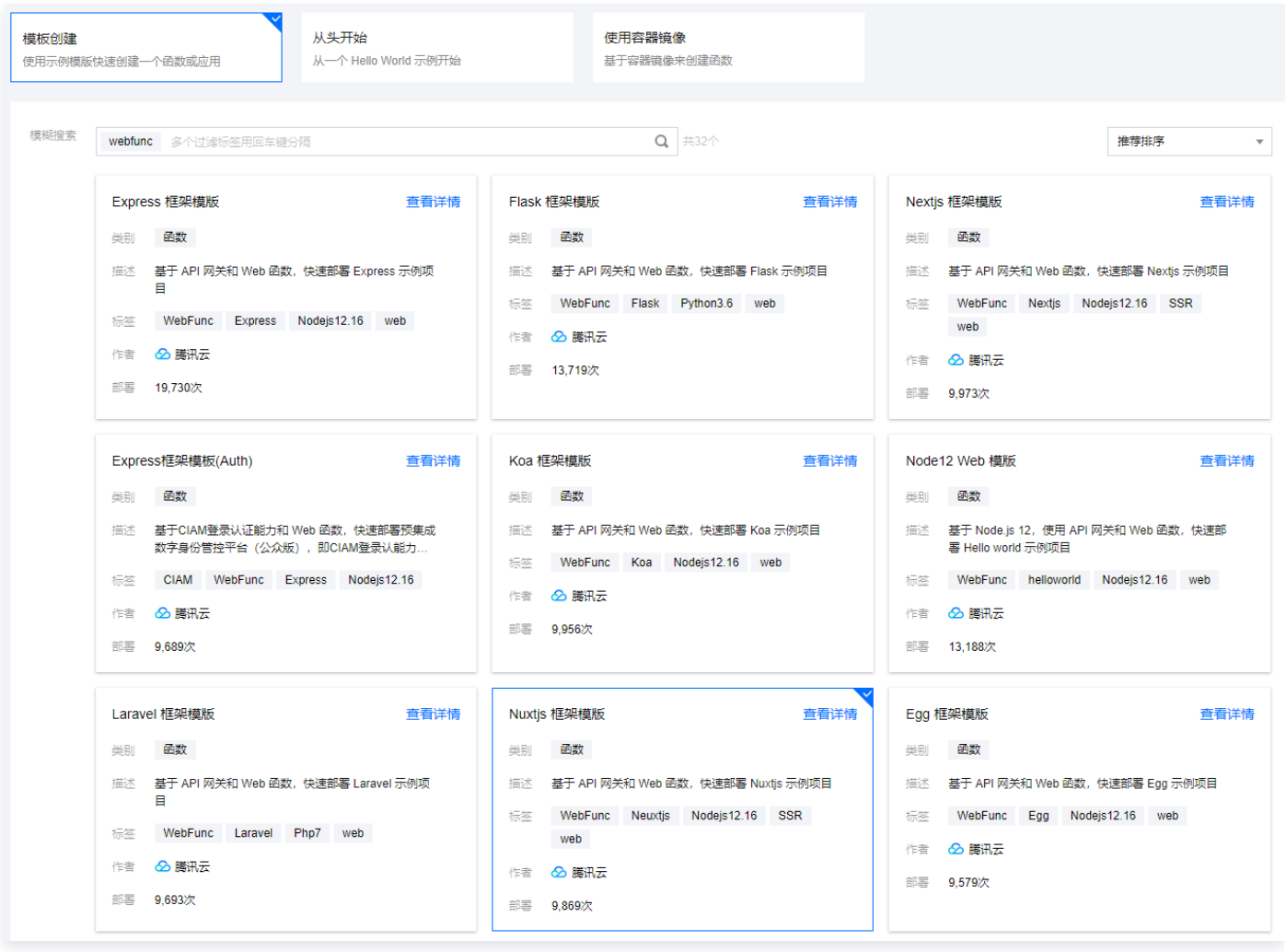
前提条件

在使用腾讯云云函数服务之前，您需要 注册腾讯云账号 并完成 实名认证 。

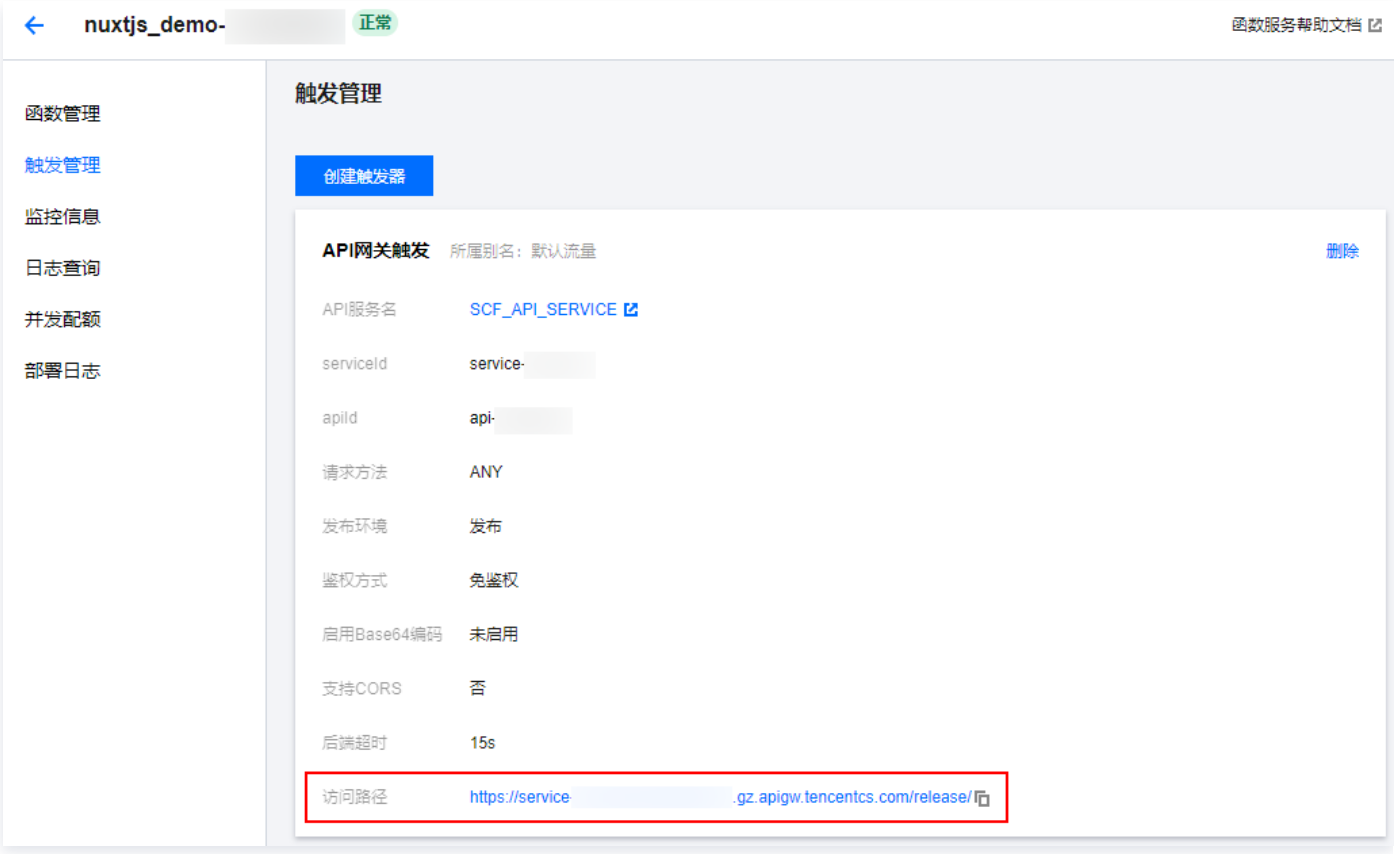
操作步骤

模板部署：一键部署 Nuxt.js 项目

1. 登录 Serverless 控制台，单击左侧导航栏的**函数服务**。
2. 在主界面上方选择期望创建函数的地域，并单击**新建**，进入函数创建流程。
3. 选择使用**模板创建**来新建函数，在搜索框里输入 `webfunc` 筛选函数模板，选择**Nuxt.js 框架模板**并单击**下一步**。如下图所示：



4. 在**新建**页面，您可以查看模板项目的具体配置信息并进行修改。
5. 单击**完成**即可创建函数。函数创建完成后，您可在**函数管理**页面查看 Web 函数的基本信息。
6. 您可以通过 API 网关生成的访问路径 URL，访问您部署的 Nuxt.js 项目。单击左侧菜单栏中的**触发管理**，查看访问路径。如下图所示：



7. 单击访问路径 URL，即可访问服务 Nuxt.js 项目。如下图所示：

> **说明**
> 由于 Nuxtjs 框架每次部署前需要重新构建，请确保本地更新代码并且重新 `build` 之后再进行部署。

自定义部署：快速迁移本地项目上云

前提条件

本地已安装 Node.js 运行环境。

本地开发

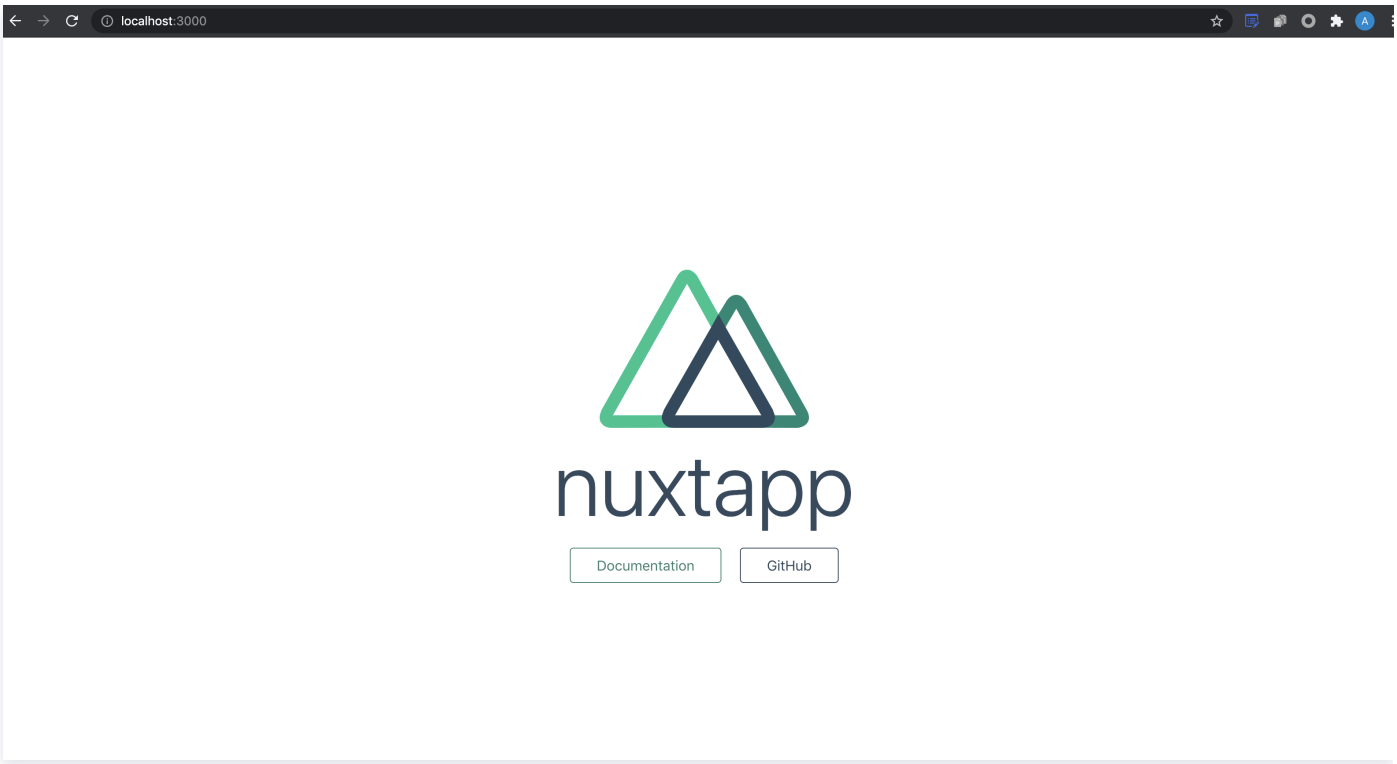
1. 参考 Nuxt.js 官方文档，安装并初始化您的 Nuxt.js 项目：

```
npx create-nuxt-app nuxt-app
```

2. 在根目录下，执行以下命令在本地直接启动服务。

```
cd nuxt-app && npm run dev
```

3. 打开浏览器访问 `http://localhost:3000` ，即可在本地完成 Nuxt.js 示例项目的访问。如下图所示：



部署上云

接下来执行以下步骤，对已初始化的项目进行简单修改，使其可以通过 Web Function 快速部署，此处项目改造通常分为以下两步：

- 新增 `scf_bootstrap` 启动文件。
- 修改监听地址与端口为 `0.0.0.0:9000` 。

具体步骤如下：

1. 在项目根目录下新建 `scf_bootstrap` 启动文件，在该文件添加如下内容（用于启动服务并指定启动端口）：

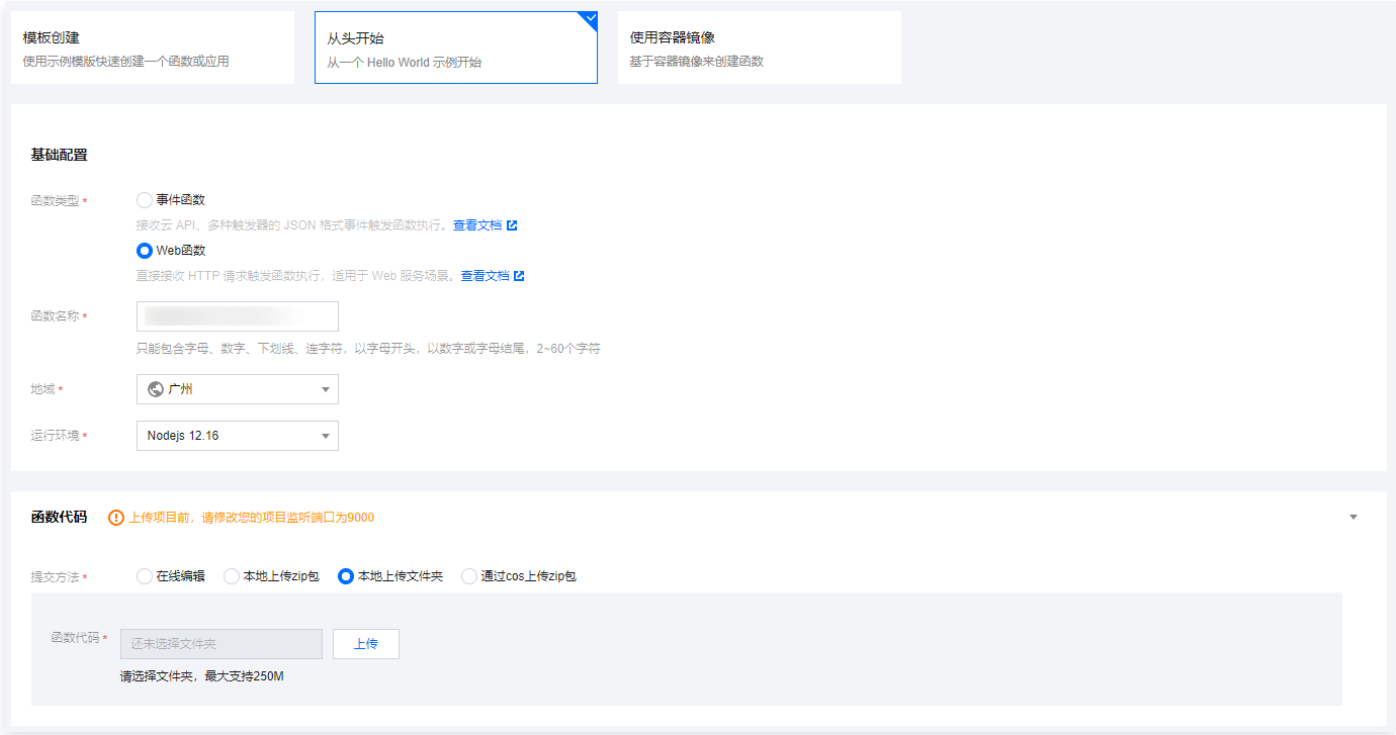
```
#!/var/lang/node12/bin/node
require("@nuxt/cli")
  .run(["start", "--port", "9000", "--hostname", "0.0.0.0"])
  .catch(error => {
    require("consola").fatal(error);
    require("exit")(2);
  });
```

> **注意**
> - 此处仅为示例启动文件，具体请根据您的业务场景进行调整。
> - 示例使用的是云函数标准 node 环境路径，本地调试时，需修改成您的本地路径。

2. 新建完成后，还需执行以下命令修改文件可执行权限，默认需要 `777` 或 `755` 权限才可正常启动。示例如下：

```
chmod 777 scf_bootstrap
```

3. 登录 Serverless 控制台，单击左侧导航栏的**函数服务**。
4. 在主界面上方选择期望创建函数的地域，并单击**新建**，进入函数创建流程。
5. 选择**从头开始**新建函数，根据页面提示配置相关选项。如下图所示：



   - **函数类型**：选择 “Web 函数”。
   - **函数名称**：填写您自己的函数名称。
   - **地域**：填写您的函数部署地域，默认为广州。
   - **运行环境**：选择 “Nodejs 12.16”。
   - **部署方式**：选择“代码部署”，上传您的本地项目。
   - **提交方法**：选择“本地上传文件夹”。
   - **函数代码**：选择函数代码在本地的具体文件夹。
6. 单击**完成**完成 Nuxt.js 项目的部署。

> **注意**
> 访问 URL 时，可能由于前端路由导致访问失败，访问时需去掉 `/release` 路径。

开发管理

部署完成后，即可在 SCF 控制台快速访问并测试您的 Web 服务，并且体验云函数多项特色功能，例如层绑定、日志管理等，享受 Serverless 架构带来的低成本、弹性扩缩容等优势。

高级能力配置
层部署

最近更新时间：2023-08-28 20:59:32

操作场景

目前 Serverless SSR 只支持上传小于 50MB 的代码，当您的项目过大时，您可以将依赖放在层中而不是部署包中，可确保部署包保持较小的体积。层的具体使用请参见 层管理相关操作 。

操作步骤

创建层

新建层并上传依赖，您可以通过以下两种方式操作：

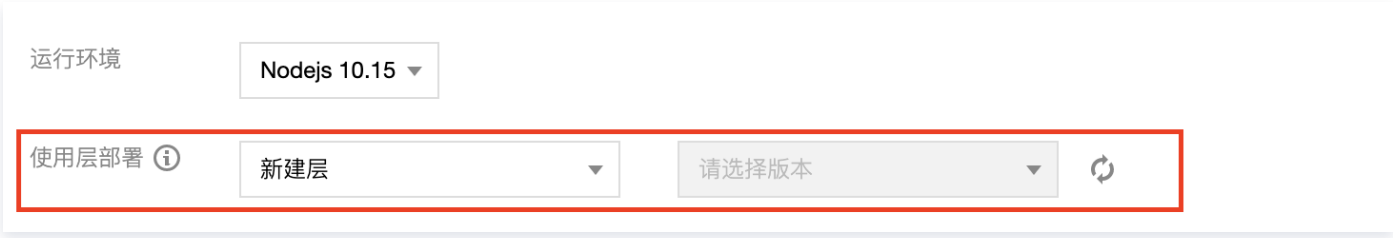
- 通过 Serverless SSR 控制台 直接创建
- 使用 Serverless Framework 的 Layer 组件（参考 Layer 组件 ）

使用层

您可以通过控制台配置和本地配置两种方法，在项目配置中使用层部署，具体如下：

控制台配置

- 使用模板创建时，Serverless SSR 会自动为您创建名为 `${appName}-layer` 的层，并自动帮您把应用的依赖项 node_modules 上传到该层中。
- 导入已有项目时，您可以选择使用新建层或已有层完成部署，选择新建层时，Serverless SSR 会自动帮您把应用的依赖项 node_modules 上传到该层中。



本地配置

在本地项目目录下，创建 **serverless.yml** 文件，完成层的名称与版本配置，再通过 `sls deploy` 进行上传，模板如下：

```
component: layer
name: layerDemo
org: orgDemo
app: appDemo
stage: dev

inputs:
  name: test
  region: ap-guangzhou
  src: ./node_modules #需要上传的目标文件路径
  runtimes:
    - Nodejs10.14
```

查看详细配置，请参见 layer 组件全量配置文档 。

项目部署

完成配置部署后，您 SSR 项目中的依赖文件将自动通过层来管理，再次开发部署时，直接引用该层信息即可，缩短部署时间，提高开发效率。

高级配置

最近更新时间：2023-08-28 20:59:33

在 Serverless SSR 控制台中，新建应用时可以选择是否启用高级配置，以实现对应用访问的优化。 本文主要对默认优化和建议优化的配置项进行说明

默认优化

node_modules 拆分为层

默认会把工程中的 node_modules 拆分为 层（layer）部署。简单来说，层就是把项目中不经常变更的部分独立出来，云函数运行时再把层包含的文件挂载到云函数的执行环境中。

- Next.js 和 Nuxt.js 项目的 node_modules 会比较大，项目稳定后，部署不需要每次都更新 node_modules。
- 把 node_modules 部署为层，每次部署只需要上传业务代码，部署会快很多。
- 更新 node_modules 的层时，建议使用 `npm install --production` 安装依赖包，减少代码包体积，加快部署和启动速度。

点击查看 层部署指引 >>

启用静态资源托管

- Next.js 应用默认会把 `.next/static` 目录和 public 目录下的所有文件启用静态资源托管。
- 把静态内容（ HTML、JavaScript、CSS、图像、视频等文件）托管到 COS，不占用云函数的访问资源，成本更低，访问速度更快。

点击查看 静态资源托管改造 >>

建议优化

CDN 加速

前提条件

- 启用 CDN 加速必须使用您已经在**腾讯云备案**的自有域名，详情请参见 备案指引 。
- 设置自定义域名，需要在域名管理设置一条 CNAME 记录，指向配置的对象存储的二级域名。
- 更多 COS 设置 CDN 加速域名的说明请参见 COS 域名管理 。

配置指引

- 控制台配置
  创建应用时，您可以直接在高级配置里开启 CDN 加速，填入您**已备案且配置好解析**的域名，完成 CDN 域名加速的配置。

  选择"自动配置 CNAME"时，Serverless SSR 会尝试自动为您配置域名的 CNAME 解析，一般需要5 – 10分钟生效，您可以在 域名解析控制台 查看解析结果，或手动配置解析。
- 本地配置

在 serverless.yml 文件中，inputs 字段里增加以下配置内容，完成 CDN 加速域名的配置。

```
org: orgDemo
app: appDemo
stage: dev
component: nextjs
name: nextjsDemo

inputs:
# 此处省略....

# 静态资源相关配置
staticConf:
 cosConf:
   # 这里是创建的 COS 桶名称
   bucket: serverless-nextjs
 cdnConf: # CDN 加速配置
   domain: static.test.com
   https:
     certId: abcdefg
```

修改配置文件 next.config.js(nuxt.js 项目为 nuxt.config.js)，将 STATIC_URL 参数改为您的自定义域名，以 next.js 为例：

```
const isProd = process.env.NODE_ENV === 'production';
const STATIC_URL ="static.test.com";
 module.exports = {
  env: {
    STATIC_URL: isProd ? STATIC_URL : '',
    },
  assetPrefix: isProd ? STATIC_URL : '',
};
```

完成修改后，重新部署，即可完成 CDN 加速域名的配置。

```
sls deploy
```

自定义域名

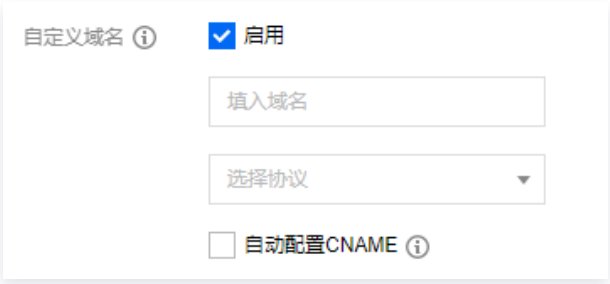
前提条件

- 系统默认会给应用分配一个二级域名，您可以通过绑定自有域名，用自有域名访问应用。
- 自定义域名必须是已经在**腾讯云备案**的域名，目前，腾讯云已推出 Serverless 备案资源包，帮您快速完成 ICP 备案，详情请参考 ICP 备案 。
- 设置自定义域名需要在域名管理配置一条 CNAME 记录，指向系统分配的二级域名。 详情请参考 配置自定义域名 。

配置指引

- 控制台配置

创建应用时，您可以直接在高级配置里开启自定义域名，填入您**已备案且配置好解析**的域名，完成自定义域名的配置。



建议您 手动配置域名解析，您也可选择自动配置，勾选"自动配置CNAME"时，Serverless SSR 会尝试自动为您配置域名的 CNAME 解析，一般需要5 – 10分钟生效，您可以在 域名解析控制台 查看解析结果，或手动配置解析。

- 本地配置

在 serverless.yml 文件中，inputs 字段里增加以下配置内容，完成自定义域名的配置。

```
org: orgDemo
app: appDemo
stage: dev
component: nextjs
name: nextjsDemo
  inputs:
# 此处省略....
# 自定义域名相关配置
 customDomains:
   - domain: test.com
     certificateId: abcdefg # 证书 ID
     # 这里将 API 网关的 release 环境映射到根路径
     pathMappingSet:
       - path: /
         environment: release
     protocols:
       - https
```

完成修改后，重新部署，即可完成自定义域名的配置。

持续构建

最近更新时间：2023-08-28 20:59:33

功能介绍

Serverless SSR 支持用户通过代码托管实现应用的持续构建，您可以按需选择您的代码仓库，并配置构建计划运行的触发规则。目前，Serverless SSR 支持 **GitHub**、**GitLab**、**Gitee** 代码源，您可以授权后直接拉取代码仓库信息，也可以填入公开的自定义仓库完成部署。

触发规则介绍

Serverless SSR 支持您选择需要的触发规则，包括以下两种触发方式：

- 自动触发构建
- 手动触发构建

自动触发构建

> **说明**
> 自定义仓库不支持自动触发。自定义仓库即指用户自有的外部代码仓库，如已托管在 GitHub、GitLab、Gitee 上的各类 repo。

选择自动触发构建，会自动监听创建应用时所选择的代码仓库分支，根据其变化来自动触发构建计划。您只需要更新代码仓库指定分支里的项目代码即可，Serverless SSR 会自动完成应用的持续部署。

手动触发构建

选择手动触发，每次应用变更，您需要在 SSR 控制台手动完成触发构建。

1. 在 SSR 控制台 首页，单击目标应用进入应用详情页，在页面顶部单击**开发部署**。
2. 在部署方式中选择**代码托管**。
3. 单击页面底部的**更新应用**，即可完成应用的重新构建。

您可以在应用详情页的部署日志页面查看每次部署信息。

静态资源托管改造

最近更新时间：2023-08-28 20:59:33

操作场景

当项目中的静态资源过多时，直接部署会导致每次请求页面时，所有的静态资源也要进行重新请求与加载，使得应用的单位时间并发数会根据页面静态资源请求数而增加，从而造成冷启动问题。 Serverless SSR 支持使用静态资源托管来存储您的静态资源，缩短冷启动时间。

配置流程

控制台配置

模板部署

通过模板创建时，Serverless SSR 已提前在模板中完成了静态资源的配置，并默认开启，您无需进行任何改造操作，即可创建一个使用静态资源托管的 SSR 应用。
创建流程：创建过程中，Serverless SSR 会自动为您创建一个新的 COS 存储桶，并将项目中自动将编译生成的 .next（或 .nuxt）和 public 文件夹静态资源上传到该 COS，使得静态资源均通过访问 COS 获取，无需重复请求云函数获取静态资源。

导入已有项目

导入已有项目时，除了在控制台开启“静态资源托管”功能外，您还需对您已有项目进行如下改造：
1. 在项目目录下，创建 `next.config.js` 配置文件（nuxt 项目配置文件名为 `nuxt.config.js`）。
2. 在配置文件中加入如下内容：

- next.js

```
const isProd = process.env.NODE_ENV === 'production';

module.exports = {
  env: {
    STATIC_URL: isProd ? process.env.STATIC_URL : '',
  },
  assetPrefix: isProd ? process.env.STATIC_URL : '',
};
```

- nuxt.js

```
export default {
  // ...
  env: {
    STATIC_URL: process.env.STATIC_URL || '',
  },
  build: {
    extend(config, { isDev, isClient }) {
      if (!isDev && process.env.STATIC_URL) {
        config.output.publicPath = process.env.STATIC_URL
      }
    },
  },
  // ...
```

```
}
```

3. 改造完成后，将已有项目导入并完成部署，Serverless SSR 会自动帮您注入生成的静态文件托管 URL，完成静态资源托管的配置。

命令行部署配置

如果您使用 Serevrless Framework 命令行工具完成项目开发，静态资源托管配置步骤如下：

1. 选择静态资源存储桶，获取存储路径。

   您可以通过 COS 控制台 或 Serverless COS 组件 快速创建您的存储桶，也可以选择已有存储桶，创建完成后通过 COS 控制台 的存储桶概览页获取存储桶路径。

2. 创建配置文件 next.config.js （nuxt 项目配置文件名为 nuxt.config.js ），填入以下内容，STATIC_URL 改为您存储桶的 URL 路径：

   - next.js

```
const isProd = process.env.NODE_ENV === 'production';
const STATIC_URL =
  "https://xxxxx.cos.ap-guangzhou.myqcloud.com";
module.exports = {
  env: {
    STATIC_URL: isProd ? STATIC_URL : '',
  },
  assetPrefix: isProd ? STATIC_URL : '',
};
```

   - nuxt.js

```
const STATIC_URL =
  "https://xxxxx.cos.ap-guangzhou.myqcloud.com";
export default {
  // ...
  env: {
    STATIC_URL: STATIC_URL || '',
  },
  build: {
```

```
    extend(config, { isDev, isClient }) {
      if (!isDev && STATIC_URL) {
        config.output.publicPath = STATIC_URL
      }
    },
  },
  // ...
}
```

3. 在 serverless.yml 中`，新增静态资源相关配置 staticConf，如下：

```
org: orgDemo
app: appDemo
stage: dev
component: nextjs
name: nextjsDemo

inputs:
  src:
    dist: ./
    hook: npm run build
    exclude:
      - .env
  region: ap-guangzhou
  runtime: Nodejs10.15
  apigatewayConf:
    # 此处省略....
  # 静态资源相关配置
  staticConf:
    cosConf:
      # 这里是创建的 COS 桶名称
      bucket: serverless-nextjs
```

4. 修改好配置后，在根目录下执行 serverless deploy ，完成部署。

```
sls deploy
```

自定义路由项目改造

最近更新时间：2024-03-04 16:19:11

操作场景

如果没有用 Express 等 Web 框架替代 Next.js 或 Nuxt.js 默认的 Web Server，上传整个工程即可。如果有用到 Express 等 Web 框架，需要做简单的改造，本文以 Express 为例进行说明。

操作步骤

步骤1：修改启动文件名

您可以通过命令行开发部署修改启动文件名：
把启动 js 文件重命名为 `sls.js` ，并把它放在项目的根目录下。

步骤2：修改监听端口

将本地监听端口修改为**导出 app 应用**，以 next.js 为例：

```
const express = require('express')
const next = require('next')
async function createServer() {
  const app = next({ dev: false })
  const handle = app.getRequestHandler()

  await app.prepare()
  const server = express()
  server.all('*', (req, res) => {
    return handle(req, res)
  })
  // define binary type for response
  // if includes, will return base64 encoded, very useful for images
  server.binaryTypes = ['*/*']
  return server
}
// comment out `listen`
// exports app server
// createServer().litsen(3000);
module.exports = createServer
```

> **说明**
> Nuxt.js 项目修改可参见 Nuxt.js 产品文档 。

ICP 备案

最近更新时间：2023-08-28 20:59:33

为什么要备案?

根据国务院令第292号《互联网信息服务管理办法》和工信部令第33号《非经营性互联网信息服务备案管理办法》规定，国家对经营性互联网信息服务实行许可制度，对非经营性互联网信息服务实行备案制度。未获取许可或者未履行备案手续的，不得从事互联网信息服务，否则属于违法行为。
因此，使用中国内地（大陆）的 Serverless 服务开办网站并绑定域名服务时必须先办理网站备案，备案成功并获取通信管理局下发的 ICP 备案号后才能开通域名访问。

- ICP 备案号以工信部网站公共查询为准：查询入口
- 更多相关法律法规请参见：法律法规

备案场景

如果您的网站托管在腾讯云中国内地（大陆）的 Serverless 服务中，且网站的主办者和域名从未办理过备案，则在开通 Serverless 服务并且使用云函数 SCF 进行自定义域名的 HTTP 访问服务前，需在腾讯云备案系统进行首次备案的操作。

备案准备

- 为了节约备案时间和顺利通过备案，建议您提前了解备案流程。
- 因各地管局要求不同，需准备的材料也有所不同。建议您提前了解各省、自治区、直辖市管局的 备案要求，以及相关 备案限制。
- Serverless 备案要求：备案本身不收取任何费用，但通过 Serverless 方式备案需购买云函数**5000万次**调用次数包与**40万GBs**资源用量包。请前往 资源包购买页面 完成购买。

> **注意**
> Serverless 备案方式已向全部用户开放。

备案流程

请参考 首次备案，通过小程序完成备案操作。

> **注意**
> 在执行 填写网站信息 步骤时，需先开启云函数备案，即在备案类型中选择**Serverless**。

常见问题

使用 Serverless 的访问域名必须要备案吗?

需根据实际情况进行判断。若默认采用 API 网关提供的三级域名访问自身是无需进行备案的，只有需要自定义域名且该域名指向中国内地（大陆）的 Serverless 服务才需要备案。例如，访问博客页面等场景。您可根据以下场景判断是否需要备案：

- 不需要备案：
  域名解析指向托管于非中国内地（大陆）的 Serverless 服务时，例如中国香港服务器，则不需要备案。
- 需要备案：
  域名指向中国内地（大陆）的 Serverless 服务时，需要完成备案。

Serverless 备案流程与云服务器 CVM 备案流程是否一致?

Serverless 备案与 CVM 备案流程上并无实质性差别，所有体验完全一致。唯一的差异点在于，CVM 备案过程中填写的 IP 将直接暴露给用户，Serverless 备案过程中 IP 将由系统自动拉取并提供。

为什么备案时 IP 地址不可访问?

备案在腾讯云，解析也需要在腾讯云。用户在使用腾讯云 Serverless 方式备案过程中，域名解析 IP 需指向腾讯云站点的 IP 地址 。

Serverless 备案有限制吗?

由于 Serverless 较为轻量，可能会实时删除或新增。为了迎合用户使用 Serverless 习惯，Serverless 备案将以账号作为维度，每个账号支持购买1个云函数资源包，支持备案2个网站。

已经在 CVM 备案的域名是否需要重新备案?

若同账号下已在 CVM 进行 ICP 备案的域名，可直接通过 API 网关绑定，无需重复备案。

注册的域名当前不使用，还需要备案吗?

域名本身无需备案的，但需实名认证。仅当该域名开通 Web 服务时，才需要备案。

Serverless 站点未完成搭建，需要办理备案吗?

不需要备案。在您的 Serverless 站点正式提供对外访问前，才需要备案。由于备案需要一定的办理时间，建议您提前在腾讯云办理备案，以便您的站点做好之后可以马上投入使用。

已备案域名接入腾讯云 Serverless 服务，是否需要重新备案?

不需要重新备案，但要办理接入备案。详情请参见 接入备案 。

什么情况下需要新增网站备案?

- 如果您存在多个域名，都需要进行备案。
- 您已经有域名进行过备案，现在需要备案新的域名。

接入备案是否可以接入多个 Serverless 服务?

同一主体备案信息可以同时接入多个网站信息，最多同时可以接入10个云函数或 CVM 备案信息。

Serverless 备案资源包购买后是否支持退款?

资源包购买之后立即生效，暂不支持退款。五天无理由退款及其他特殊退款请通过 提交工单 进行咨询。