ALIBABA CLOUD

阿里云

函数计算

迁移传统框架

文档版本：20201019

阿里云

法律声明

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

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

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

目录

1.迁移Beego到函数计算

本文介绍如何将Beego应用部署到函数计算。与传统的部署方法相比，您可以跳过购买机器等步骤，将传统的Beego应用一键部署至远端直接用于生产，并且拥有了弹性伸缩、按量付费和免运维等特性。

背景信息

Beego是一个快速开发Go应用的HTTP框架，可以用来快速开发API、Web及后端服务等各种应用，是一个RESTful的框架，主要设计灵感来源于Tornado、Sinatra和Flask这三个框架，但是结合了Go本身的一些特性Interface、Struct嵌入等。

步骤一：准备环境

您无需安装Docker，仅安装Funcraft即可，最简单的方式即直接下载可执行的二进制文件。

1. 安装Funcraft到本机。详情请参见安装Funcraft。
2. 执行 `fun --version` 检查安装是否成功。

步骤二：迁移应用

1. 执行以下命令安装Beego，详情请参见官方示例。

   ```
   go get github.com/astaxie/beego
   ```

2. 执行以下命令创建 `hello.go` 项目。

   ```
   package main
   import "github.com/astaxie/beego"
   func main(){
      beego.Run()
   }
   ```

3. 执行以下命令运行本地项目。

   ```
   go run hello.go
   ```

4. 执行 `fun deploy -y` 命令将项目部署至函数计算。

   ```
   fun deploy -y
   ```

   返回结果如下。

```
current folder is not a fun project.
Fun detected your application doesn't listen on '0.0.0.0:9000' in hello.go
Fun will replace your addr to '0.0.0.0:9000', and also backup your origin file hello.go to hello.go.bak
? Are your sure? Yes
Could not find any bin files from current folder.
Before using 'fun deploy', you must use 'GOARCH=amd64 GOOS=linux go build -ldflags "-s -w"' to c
ompile your project.
? Let Fun exec this command for you? Yes
Executing command 'GOARCH=amd64 GOOS=linux go build -ldflags "-s -w"'...
Tips:
You must use 'GOARCH=amd64 GOOS=linux go build -ldflags "-s -w"' to recompile your project ever
y time before using fun deploy.
Generating template.yml...
Generate Fun project successfully!
========= Fun will use 'fun deploy' to deploy your application to Function Compute! =========
... ..... .....

            trigger httpTrigger deploy success
      function express deploy success
service express deploy success
Detect 'DomainName:Auto' of custom domain 'Domain'
Fun will reuse the temporary domain 15014775-XXX.test.functioncompute.com, expired at 2020-04-
03 09:52:55, limited by 1000 per day.
Waiting for custom domain Domain to be deployed...
custom domain Domain deploy success
```

函数计算要求启动服务必须监听0.0.0.0:9000端口，详情请参见简介。您可以在部署日志中看到，Funcraft会尝试去检测应用的启动端口。如果端口不匹配，按下回车后Funcraft会帮您修改，然后自动检测构建生成的可执行程序。如果检测不到可执行程序，则会提示您使用指定命令进行编译，您按下回车后Funcraft会帮您编译，编译完成后，会自动生成Funcraft所需要的*bootstrap*文件以及*template.yml*文件，最后进行自动部署。

部署成功后，您可以在日志中看到函数计算为您生成的临时域名，通过这个临时域名您可直接访问刚部署的应用。

> **说明** 临时域名仅用作演示以及开发，具有时效性。如需用作生产，请绑定已经在阿里云备案的域名，详情请参见绑定自定义域名。

2.迁移Express到函数计算

本文介绍如何将Express应用部署到函数计算。与传统的部署方法相比，您可将传统的Express应用一键部署至远端直接用于生产。跳过购买机器等步骤的同时，还拥有了弹性伸缩、按量付费和免运维等特性。

背景信息

Express是一个基于Node.js平台的极简、灵活的Web应用开发框架，提供一系列强大的特征，帮助您创建各种Web和移动设备应用。

步骤一：准备环境

您无需安装Docker，仅安装Funcraft即可，最简单的方式即直接下载可执行的二进制文件。

1. 安装Funcraft到本机。详情请参见安装Funcraft。
2. 执行 `fun --version` 检查安装是否成功。

步骤二：迁移应用

1. 执行以下命令创建一个Express项目。

   ```
   npx express-generator
   ```

   详情请参见Express application generator，或者您可以按照官方描述的步骤创建简单的Hello world example，若有Express项目则跳过该步骤。

2. 执行以下命令进入刚创建的示例项目或您已有的项目。

   ```
   cd <project-name>
   ```

3. 执行以下命令安装依赖。

   ```
   npm install
   ```

4. 执行以下命令运行项目。
   - MacOS、Linux平台

     ```
     DEBUG=myapp:* npm start
     ```

   - Windows平台

     ```
     set DEBUG=myapp:* & npm start
     ```

5. 执行 `fun deploy -y` 命令将项目部署至函数计算。

   Funcraft会自动进入部署流程。

   ```
   fun deploy -y
   ```

   返回结果如下。

```
current folder is not a fun project.
Generating /Users/txd123/Desktop/Express/bootstrap...
Generating template.yml...
Generate Fun project successfully!
========= Fun will use 'fun deploy' to deploy your application to Function Compute! =========
using region: cn-qingdao
using accountId: **********3743
using accessKeyId: **********Ptgk
using timeout: 60

Collecting your services information, in order to caculate devlopment changes...

Resources Changes(Beta version! Only FC resources changes will be displayed):
...  ...  ...  ...
            trigger httpTrigger deploy success
      function Express deploy success
service Express deploy success

Detect 'DomainName:Auto' of custom domain 'Domain'
Request a new temporary domain ...
The assigned temporary domain is 15795585-XXX.test.functioncompute.com,  expired at 2020-04-1
2 10:46:25, limited by 1000 per day.
Waiting for custom domain Domain to be deployed...
custom domain Domain deploy success
```

部署成功后，您可以在日志中看到函数计算为您生成的临时域名，通过这个临时域名您可直接访问刚部署的应用。

> **说明** 临时域名仅用作演示以及开发，具有时效性。如需用作生产，请绑定已经在阿里云备案的域名，详情请参见绑定自定义域名。

3.迁移Gin到函数计算

本文介绍如何将Gin应用部署到函数计算。与传统的部署方法相比，您可以跳过购买机器等步骤，将传统的Gin应用一键部署至远端直接用于生产，并且拥有弹性伸缩、按量付费和免运维等特性。

背景信息

Gin是一个Golang的微框架，封装优雅，API友好，源码注释明确。具有快速灵活，容错方便等特点。

步骤一：准备环境

您无需安装Docker，仅安装Funcraft即可，最简单的方式即直接下载可执行的二进制文件。

1. 安装Funcraft到本机。详情请参见安装Funcraft。
2. 执行 `fun --version` 检查安装是否成功。

步骤二：迁移应用

示例一

1. 在已安装1.11及以上版本的Golang的环境中安装Gin，详情请参见官方示例。

   ```
   go get -u github.com/gin-gonic/gin
   ```

2. 创建一个example.go项目，并写入以下代码。

   ```
   package main
   import "github.com/gin-gonic/gin"
   func main() {
   r := gin.Default()
   r.GET("/ping", func(c *gin.Context) {
      c.JSON(200, gin.H{
         "message": "pong",
      })
   })
   r.Run() // listen and serve on 0.0.0.0:8080 (for windows "localhost:8080")
   }
   ```

3. 执行以下命令运行本地项目。

   ```
   go run example.go
   ```

4. 执行 `fun deploy -y` 命令将项目部署至函数计算。

   ```
   fun deploy -y
   ```

   返回结果如下。

```
current folder is not a fun project.
Fun detected your application doesn't listen on '0.0.0.0:9000' in example.go
Fun will replace your addr to '0.0.0.0:9000', and also backup your origin file example.go to example.
go.bak
? Are your sure? Yes
Could not find any bin files from current folder.
Before using 'fun deploy', you must use 'GOARCH=amd64 GOOS=linux go build -ldflags "-s -w"' to c
omile your project.
? Let Fun exec this command for you? Yes
Executing command 'GOARCH=amd64 GOOS=linux go build -ldflags "-s -w"'...
Tips:
You must use 'GOARCH=amd64 GOOS=linux go build -ldflags "-s -w"' to recompile your project ever
y time before using fun deploy.
Generating template.yml...
Generate Fun project successfully!
========= Fun will use 'fun deploy' to deploy your application to Function Compute! =========
... ... ....
            trigger httpTrigger deploy success
      function express deploy success
service express deploy success
Detect 'DomainName:Auto' of custom domain 'Domain'
Request a new temporary domain ...
The assigned temporary domain is 15014775-1986***.test.functioncompute.com, expired at 2020-0
4-03 09:52:55, limited by 1000 per day.
Waiting for custom domain Domain to be deployed...
custom domain Domain deploy success
```

函数计算要求启动服务必须监听0.0.0.0:9000端口，详情请参见简介。您可以在部署日志中看到，Funcraft会尝试去检测应用的启动端口。如果端口不匹配，按下回车后Funcraft会帮您修改，然后自动检测构建生成的可执行程序。如果检测不到可执行程序，则会提示您使用指定命令进行编译，您按下回车后Funcraft会帮您编译，编译完成后，会自动生成Funcraft所需要的*bootstrap*文件以及*template.yml*文件，最后进行自动部署。

部署成功后，您可以在日志中看到函数计算为您生成的临时域名，通过这个临时域名您可直接访问刚部署的应用。

> **说明** 临时域名仅用作演示以及开发，具有时效性。如需用作生产，请绑定已经在阿里云备案的域名，详情请参见绑定自定义域名。

示例二

1. 执行以下命令将示例克隆到本地。详情请参见官方示例。

```
git clone https://github.com/tanhe123/mdblog.git
```

2. 修改配置文件。
   i. 在*config*目录下，将*config.example.toml*文件名称修改为*config.toml*。
   ii. 打开*config.toml*文件，修改配置。
   - 将 `port = 8091` 修改为 `port = 9000` ，表示应用在9000端口启动。
   - 将 `debug = true` 修改为 `debug = false` ，表示使用生产版本。
   - 将 `dir = "logs"` 修改为 `dir = "/tmp"` ，表示日志写到*/tmp*目录，函数计算在不挂载NAS的情况下，只有该目录有读写权限。
3. 执行以下命令编译项目。

```
go build
```

4. 执行以下命令运行本地项目。

```
./mdblog
```

5. 执行 `fun deploy -y` 命令将项目部署至函数计算。

```
fun deploy -y
```

部署成功后，您可以在日志中看到函数计算为您生成的临时域名，通过这个临时域名您可直接访问刚部署的应用。

> **说明** 临时域名仅用作演示以及开发，具有时效性。如需用作生产，请绑定已经在阿里云备案的域名，详情请参见绑定自定义域名。

4.迁移Hacker News到函数计算

本文介绍如何将Hacker News应用部署到函数计算。与传统的部署方法相比，您可以跳过购买机器等步骤，将传统的Hacker News应用一键部署至远端直接用于生产，并且拥有弹性伸缩、按量付费和免运维等特性。

步骤一：准备环境

您无需安装Docker，仅安装Funcraft即可，最简单的方式即直接下载可执行的二进制文件。

1. 安装Funcraft到本机。详情请参见安装Funcraft。
2. 执行 `fun --version` 检查安装是否成功。

步骤二：迁移应用

1. 执行以下命令将Hacker News示例项目克隆到本地。若已有Hacker News项目则跳过该步骤。

   ```
   git clone https://github.com/nuxt/hackernews.git
   ```

2. 执行以下命令进入刚创建的示例项目或您已有的项目。

   ```
   cd hackernews
   ```

3. 执行以下命令安装依赖。

   ```
   npm install
   ```

4. 执行以下命令运行本地项目。

   ```
   npm run dev
   ```

5. 执行以下命令编译Hacker News项目。

   ```
   npm run build
   ```

6. 执行 `fun deploy -y` 命令将项目部署至函数计算。

   ```
   fun deploy -y
   ```

   返回结果如下。

```
current folder is not a fun project.
Generating /Users/XXX/Desktop/hackernews/bootstrap...
Generating template.yml...
Generate Fun project successfully!
========= Fun will use 'fun deploy' to deploy your application to Function Compute! =========
... ... ...
Fun detected that your function hackernews/hackernews sizes exceed 50M. It is recommended th
at using the nas service to manage your function dependencies.
? Do you want to let fun to help you automate the configuration? Yes
? We recommend using the 'NasConfig: Auto' configuration to manage your function dependencies
.
Yes
... ... ...
starting upload /Users/XXX/Desktop/hackernews/node_modules to nas://hackernews/mnt/auto/
node_modules/
start fun nas init...
... ... ...
        trigger httpTrigger deploy success
   function hackernews deploy success
service hackernews deploy success
Detect 'DomainName:Auto' of custom domain 'Domain'
Request a new temporary domain ...
The assigned temporary domain is 14942717-XXX.test.functioncompute.com, expired at 2020-04-0
2 13:51:57, limited by 1000 per day.
Waiting for custom domain Domain to be deployed...
custom domain Domain deploy success
```

函数计算平台对上传的代码包大小的有相应的限制，详情请参见资源使用限制。您可以在部署日志中看到Funcraft检查到如果您的函数代码包大小超过了函数计算平台限制后会进入大依赖向导，此时您需要输入*Y*，Funcraft会自动帮您把函数中的第三方依赖上传至NAS。

部署成功后，您可以在日志中看到函数计算为您生成的临时域名，通过这个临时域名您可直接访问刚部署的应用。

> **说明** 临时域名仅用作演示以及开发，具有时效性。如需用作生产，请绑定已经在阿里云备案的域名，详情请参见绑定自定义域名。

5.迁移Next.js到函数计算

本文介绍如何将Next.js应用部署到函数计算。与传统的部署方法相比，您可以跳过购买机器等步骤，将传统的Next.js应用一键部署至远端直接用于生产，并且拥有弹性伸缩、按量付费和免运维等特性。

背景信息

Next.js是一种React的服务端渲染框架，集成度极高，框架自身集成了webpack、babel、express等，使得开发者仅依赖Next、react、react-dom就可以非常方便地构建自己的SSR React应用，开发者甚至都不用关心路由。Next.js的高度集成性，易于实现代码分割、路由跳转、热更新以及服务端渲染和前端渲染。

步骤一：准备环境

您无需安装Docker，仅安装Funcraft即可，最简单的方式即直接下载可执行的二进制文件。

1. 安装Funcraft到本机。详情请参见安装Funcraft。
2. 执行 `fun --version` 检查安装是否成功。

步骤二：迁移应用

1. 执行以下命令创建一个Next.js项目。若已有Next.js项目则跳过该步骤。

   ```
   npm init next-app
   ```

2. 执行以下命令进入刚刚创建的示例项目或已存在的项目。

   ```
   cd nextjs
   ```

3. 执行以下命令运行本地项目。

   ```
   npm run dev
   ```

   效果如下。

   

4. 执行以下命令编译Next.js项目。

   ```
   npm run build
   ```

5. 执行 `fun deploy -y` 命令将项目部署至函数计算。

   Funcraft会自动进入部署流程。

   在部署过程中，根据提示进行相应的操作。

   i. 当Funcraft检测到这不是一个Funcraft项目，会提示协助创建。直接按回车键或者输入*Y*。

   

ii. Funcraft项目自动创建成功，提示是否进行部署。直接按回车键或者输入 *Y* 进行确认。

Funcraft会将应用部署到线上。

部署成功后，您可以在日志中看到函数计算为您生成的临时域名，通过这个临时域名您可直接访问刚部署的应用。

> **说明** 临时域名仅用作演示以及开发，具有时效性。如需用作生产，请绑定已经在阿里云备案的域名，详情请参见**绑定自定义域名**。

6.迁移Nuxt.js到函数计算

本文介绍如何将Nuxt.js应用部署到函数计算。与传统的部署方法相比，您可以跳过购买机器等步骤，将传统的Nuxt.js应用一键部署至远端直接用于生产，并且拥有弹性伸缩、按量付费和免运维等特性。

背景信息

Nuxt.js是一个基于Vue.js的通用应用框架。通过对客户端/服务端基础架构的抽象组织，主要关注的是应用的UI渲染。Nuxt.js预设了利用Vue.js开发服务端渲染的应用所需要的各种配置，为客户端/服务端这种典型的应用架构模式提供了很多有用的特性，例如异步数据加载、中间件支持、布局支持等。

步骤一：准备环境

您无需安装Docker，仅安装Funcraft即可，最简单的方式即直接下载可执行的二进制文件。

1. 安装Funcraft到本机。详情请参见安装Funcraft。
2. 执行 `fun --version` 检查安装是否成功。

步骤二：迁移应用

1. 执行以下命令创建一个Nuxt.js项目。若已有Nuxt.js项目则跳过该步骤。

   ```
   npx create-nuxt-app <project-name>
   ```

2. 执行以下命令进入刚创建的示例项目或您已有的项目。

   ```
   cd <project-name>
   ```

3. 执行以下命令安装依赖。

   ```
   yarn install
   ```

4. 执行以下命令运行本地项目。

   ```
   yarn dev
   ```

   运行效果如下。

5. 执行以下命令编译Nuxt.js项目。

   ```
   yarn build
   ```

6. 执行 `fun deploy -y` 命令将项目部署至函数计算。

   Funcraft会自动进入部署流程。

   > 说明 若出现无法添加yml文件的问题，请更新Funcraft到最新版本。

   部署成功后，您可以在日志中看到函数计算为您生成的临时域名，通过这个临时域名您可直接访问刚部

署的应用。

> **说明** 临时域名仅用作演示以及开发，具有时效性。如需用作生产，请绑定已经在阿里云备案的域名，详情请参见绑定自定义域名。

7.迁移ThinkPHP到函数计算

本文介绍如何将ThinkPHP应用部署到函数计算。与传统的部署方法相比，您可以跳过购买机器等步骤，将传统的ThinkPHP应用一键部署至远端直接用于生产，并且还拥有了弹性伸缩、按量付费和免运维等特性。

背景信息

ThinkPHP是一个免费开源的、快速简单的、面向对象的轻量级PHP开发框架，是为了敏捷Web应用开发和简化企业应用开发而诞生的。ThinkPHP从诞生以来一直秉承简洁实用的设计原则，在保持出色的性能和至简代码的同时，更注重易用性。遵循Apache2开源许可协议发布，意味着你可以免费使用ThinkPHP，甚至允许把你基于ThinkPHP开发的应用开源或商业产品发布、销售。

步骤一：准备环境

- 安装Funcraft

  > **说明** 您无需安装Docker，仅安装Funcraft即可，最简单的方式即直接下载可执行的二进制文件。安装后，您可以执行 `fun --version` 检查安装是否成功。

- 安装Composer

步骤二：迁移应用

1. 执行以下命令创建一个ThinkPHP项目。若已有ThinkPHP项目则跳过该步骤。

   ```
   composer create-project topthink/think tp
   ```

2. 执行以下命令进入刚刚创建的项目或已存在的项目。

   ```
   cd <project-name>
   ```

3. 执行以下命令运行本地项目。

   ```
   php think run
   ```

4. 执行 `fun deploy -y` 命令将项目部署至函数计算。

   Funcraft会自动进入部署流程。

   ```
   fun deploy -y
   ```

   返回代码如下。

```
current folder is not a fun project.
downloading nginx and php7.2 zip from https://gosspublic.alicdn.com/fun/frameworks/support/fun-support-custom-php-d73a6bd6.zip to /private/var/folders/wl/_2ngtj291wx1cj55xlnn290w0000gn/T/fun-support-custom-php-d73a6bd6.zip...
extract nginx and php7.2 zip to custom runtime...
Generating
... ... ...
      trigger httpTrigger deploy success
   function tp deploy success
service tp deploy success
Detect 'DomainName:Auto' of custom domain 'Domain'
Request a new temporary domain ...
The assigned temporary domain is 15631862-XXX.test.functioncompute.com，expired at 2020-04-10 13:17:42, limited by 1000 per day.
Waiting for custom domain Domain to be deployed...
custom domain Domain deploy success
```

部署成功后，您可以在日志中看到函数计算为您生成的临时域名，通过这个临时域名您可直接访问刚部署的应用。

> **说明** 临时域名仅用作演示以及开发，具有时效性。如需用作生产，请绑定已经在阿里云备案的域名，详情请参见绑定自定义域名。

8.迁移Spring Boot到函数计算

本文介绍如何将Spring Boot应用部署到函数计算。与传统的部署方法相比，您可将传统的Spring Boot应用一键部署至远端直接用于生产。跳过购买机器等步骤的同时，还拥有了弹性伸缩、按量付费和免运维等特性。

背景信息

Spring Boot是由Pivotal团队在2013年开始研发，于2014年4月发布第一个版本的全新开源轻量级框架。它基于Spring 4.0设计，不仅继承了Spring框架原有的优秀特性，还通过简化配置进一步简化Spring应用的整个搭建和开发过程。此外Spring Boot通过集成大量的框架，使得依赖包的版本冲突、引用的不稳定性等问题得到了解决。

步骤一：准备环境

您无需安装Docker，仅安装Funcraft即可，最简单的方式即直接下载可执行的二进制文件。

1. 安装Funcraft到本机。详情请参见安装Funcraft。
2. 执行 `fun --version` 检查安装是否成功。

步骤二：迁移应用

1. 创建一个Spring Boot项目，详情请参见Spring Quickstart Guide。
2. 执行以下命令进入刚创建的示例项目或您已有的项目。

   ```
   cd <project-name>
   ```

3. 执行以下命令运行本地项目。
   - MacOS、Linux平台运行项目。

     ```
     ./mvnw spring-boot:run
     ```

   - Windows平台运行项目。

     ```
     mvnw spring-boot:run
     ```

4. 在项目的根目录下执行 `mvn package` 命令打包。
   编译输出结果与以下示例类似。

   ```
   mvn package
   ```

```
[INFO] Scanning for projects...
[INFO]
[INFO] ----------------------< com.example:Spring-Boot >-----------------------
[INFO] Building Spring-Boot 0.0.1-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:3.1.0:resources (default-resources) @ Spring-Boot ---
... ... ...
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- maven-jar-plugin:3.1.2:jar (default-jar) @ Spring-Boot ---
[INFO] Building jar: /Users/txd123/Desktop/Spring-Boot/target/Spring-Boot-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- spring-boot-maven-plugin:2.2.6.RELEASE:repackage (repackage) @ Spring-Boot ---
[INFO] Replacing main artifact with repackaged archive
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  38.850 s
[INFO] Finished at: 2020-03-31T15:09:34+08:00
[INFO] ------------------------------------------------------------------------
```

5. 执行 `fun deploy -y` 命令将项目部署至函数计算。

   Funcraft会自动进入部署流程。

   ```
   fun deploy -y
   ```

```
current folder is not a fun project.
Generating template.yml...
Generate Fun project successfully!
========== Fun will use 'fun deploy' to deploy your application to Function Compute! ==========
using region: cn-qingdao
using accountId: ***********3743
using accessKeyId: ***********Ptgk
using timeout: 60

Collecting your services information, in order to caculate devlopment changes...

Resources Changes(Beta version! Only FC resources changes will be displayed):

            trigger httpTrigger deploy success
      function Spring-Boot deploy success
service Spring-Boot deploy success

Detect 'DomainName:Auto' of custom domain 'Domain'
Request a new temporary domain ...
The assigned temporary domain is 15639196-XXX.test.functioncompute.com,  expired at 2020-04-1
0 15:19:56, limited by 1000 per day.
Waiting for custom domain Domain to be deployed...
custom domain Domain deploy success
```

部署成功后，您可以在日志中看到函数计算为您生成的临时域名，通过这个临时域名您可直接访问刚部署的应用。

> **说明** 临时域名仅用作演示以及开发，具有时效性。如需用作生产，请绑定已经在阿里云备案的域名，详情请参见绑定自定义域名。

9.迁移Egg.js到函数计算

本文介绍如何将Egg.js应用部署到函数计算。与传统的部署方法相比，您可以跳过购买机器等步骤，将传统的Egg.js应用一键部署至远端直接用于生产，并且拥有弹性伸缩、按量付费和免运维等特性。

背景信息

Egg.js为企业级框架和应用而生，为开发人员提供了Web开发的核心功能和一套具有高扩展性的插件机制。开发人员按照统一的约定开发应用，从而降低开发和维护成本。更多Egg的内容，请参见Egg.js。

步骤一：准备环境

您无需安装Docker，仅安装Funcraft即可，最简单的方式即直接下载可执行的二进制文件。

1. 安装Funcraft到本机。详情请参见安装Funcraft。
2. 执行 `fun --version` 检查安装是否成功。

步骤二：迁移应用

示例一：以部署一个HelloWorld应用为例介绍具体的操作步骤。

1. 执行以下命令创建一个Egg.js项目，详情请参见快速初始化。若已有Egg.js项目则跳过该步骤。

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

2. 执行以下命令运行本地项目。

   ```
   npm run dev
   ```

3. 执行 `fun deploy -y` 命令将项目部署至函数计算。

   Funcraft会自动进入部署流程。

   ```
   fun deploy -y
   ```

   返回结果如下。

```
using template: template.yml
using region: cn-qingdao
using accountId: ***********3743
using accessKeyId: ***********Ptgk
using timeout: 60

Collecting your services information, in order to caculate devlopment changes...

Resources Changes(Beta version! Only FC resources changes will be displayed):
... ... ...
      trigger httpTrigger deploy success
  function egg-example deploy success
service egg-example deploy success

Detect 'DomainName:Auto' of custom domain 'Domain'
Fun will reuse the temporary domain 17090425-19861144305****.test.functioncompute.com, expired
at 2020-04-27 10:27:05, limited by 1000 per day.

Waiting for custom domain Domain to be deployed...
custom domain Domain deploy success
```

部署成功后，您可以在日志中看到函数计算为您生成的临时域名，通过这个临时域名您可直接访问刚部署的应用。

> **说明** 临时域名仅用作演示以及开发，具有时效性。如需用作生产，请绑定已经在阿里云备案的域名，详情请参见绑定自定义域名。

示例二：以部署一个开源的Egg.js Web应用为例介绍具体的操作步骤。

应用示例预览效果请参见预览效果。

1. 执行以下命令把示例克隆到本地。

```
git clone https://github.com/OrangeXC/mtime
```

2. 执行以下命令进入*mtime*目录，并安装依赖。

```
cd mtime
npm install
```

3. 在本地打开*config/config.default.js*文件，将MySQL的username、password配置为正确的值。在本地启动应用时，需要使用*config/config.default.js*文件中的数据库配置。

> **说明** 在生产环境中启动应用时，Egg.js默认优先使用*config/config.prod.js*中的数据库配置。因此您需要将生产环境中的数据库信息配置到该文件中。配置完成后，您可以在本地使用 `npm run start` 和 `npm run stop` 命令以生产的方式启停应用来验证配置是否正确。

4. 执行以下命令运行本地项目。

```
npm run dev
```

5. 执行以下命令修改*config/config.prod.js*文件，获取Egg.js缓存与日志目录的读取权限。

```
config.rundir = '/tmp/run',
config.logger = {
  dir: '/tmp/log',
}
```

6. 执行 `fun deploy -y` 命令将项目部署至函数计算。

```
fun deploy -y
```

返回结果如下。

```
using template: template.yml
using region: cn-qingdao
using accountId: ***********3743
using accessKeyId: ***********Ptgk
using timeout: 60

Collecting your services information, in order to caculate devlopment changes...

Resources Changes(Beta version! Only FC resources changes will be displayed):
... ... ...
      trigger httpTrigger deploy success
  function egg-example deploy success
service egg-example deploy success

Detect 'DomainName:Auto' of custom domain 'Domain'
Fun will reuse the temporary domain 17090425-19861144305****.test.functioncompute.com, expired
at 2020-04-27 10:27:05, limited by 1000 per day.

Waiting for custom domain Domain to be deployed...
custom domain Domain deploy success
```

部署成功后，您可以在日志中看到函数计算为您生成的临时域名，通过这个临时域名您可直接访问刚部署的应用。

> **说明** 临时域名仅用作演示以及开发，具有时效性。如需用作生产，请绑定已经在阿里云备案的域名，详情请参见绑定自定义域名。

10.迁移Laravel到函数计算

本文介绍如何将Laravel应用部署到函数计算。与传统的部署方法相比，您可以跳过购买机器等步骤，将传统的Laravel应用一键部署至远端直接用于生产，并且拥有弹性伸缩、按量付费和免运维等特性。

步骤一：准备环境

- 安装Funcraft

  > **说明** 您无需安装Docker，仅安装Funcraft即可，最简单的方式即直接下载可执行的二进制文件。安装后，您可以执行 `fun --version` 检查安装是否成功。

- 安装Composer

步骤二：迁移应用

1. 执行以下命令创建一个Laravel项目，详情请参见官方文档，若已有Laravel项目则跳过该步骤。

   ```
   composer create-project laravel/laravel=5.8.* --prefer-dist  mylaravel
   ```

2. 执行以下命令进入刚创建的示例项目或您已有的项目。

   ```
   cd mylaravel
   ```

3. 在本地打开*bootstrap*目录下的*app.php*文件，添加以下代码，实现将Laravel项目中的*storage*目录放到具有读写权限的*/tmp*目录下，以获取读写权限。

   ```
   $app->useStoragePath(env('STORAGE_PATH', dirname(__DIR__) . '/storage'));
   ```

4. 执行以下命令运行本地项目。

   ```
   composer install
   ```

5. 执行 `fun deploy -y` 命令将项目部署至函数计算。

   ```
   fun deploy -y
   ```

   返回结果如下。

   ```
   current folder is not a fun project.
   downloading nginx and php7.2 zip from https://gosspublic.alicdn.com/fun/frameworks/support/fun-support-custom-php-d73a6bd6.zip to /private/var/folders/wl/_2ngtj291wx1cj55xlnn290w0000gn/T/fun-support-custom-php-d73a6bd6.zip...
   extract nginx and php7.2 zip to custom runtime...
   Generating /Users/txd123/Desktop/demo123/mylaravel/.fun/root/etc/php/7.2/fpm/php-fpm.conf...
   Generating /Users/txd123/Desktop/demo123/mylaravel/.fun/root/etc/php/7.2/fpm/pool.d/www.conf...
   Generating /Users/txd123/Desktop/demo123/mylaravel/.fun/root/etc/nginx/nginx.conf
   ```

```
Generating /Users/txd123/Desktop/demo123/mylaravel/.fun/root/etc/nginx/nginx.conf...
Generating /Users/txd123/Desktop/demo123/mylaravel/.fun/root/etc/logrotate.d/nginx...
Generating /Users/txd123/Desktop/demo123/mylaravel/.fun/root/etc/logrotate.d/php7.2-fpm...
Generating /Users/txd123/Desktop/demo123/mylaravel/.fun/root/usr/lib/php/7.2/php.ini-production...
Generating /Users/txd123/Desktop/demo123/mylaravel/.fun/root/etc/nginx/sites-enabled/laravel.conf...
Generating /Users/txd123/Desktop/demo123/mylaravel/.funignore...
File /Users/txd123/Desktop/demo123/mylaravel/.funignore already exists, Fun will rename to /Users/txd123/Desktop/demo123/mylaravel/.funignore.bak
Generating /Users/txd123/Desktop/demo123/mylaravel/laravel_bootstrap...
Generating template.yml...
Generate Fun project successfully!

========== Fun will use 'fun deploy' to deploy your application to Function Compute! ==========
using region: cn-beijing
using accountId: ***********3743
using accessKeyId: ***********Ptgk
using timeout: 60

Collecting your services information, in order to caculate devlopment changes...

Resources Changes(Beta version! Only FC resources changes will be displayed):
...  ...  ...

Waiting for service mylaravel to be deployed...
    Waiting for function mylaravel to be deployed...
        Waiting for packaging function mylaravel code...
        The function mylaravel has been packaged. A total of 7987 files were compressed and the final size was 32.34 MB
        Waiting for HTTP trigger httpTrigger to be deployed...
        triggerName: httpTrigger
        methods: [ 'GET', 'POST', 'PUT' ]
        trigger httpTrigger deploy success
    function mylaravel deploy success
service mylaravel deploy success

Detect 'DomainName:Auto' of custom domain 'Domain'
Fun will reuse the temporary domain http://19247408-XXX.test.functioncompute.com, expired at 2020-05-22 09:36:48, limited by 1000 per day.
```

```
Waiting for custom domain Domain to be deployed...
custom domain Domain deploy succes
```

部署成功后，您可以在日志中看到函数计算为您生成的临时域名，通过这个临时域名您可直接访问刚部署的应用。

> **说明** 临时域名仅用作演示以及开发，具有时效性。如需用作生产，请绑定已经在阿里云备案的域名，详情请参见绑定自定义域名。