阿里云 Web应用托管服务

技术栈

文档版本: 20200330

为了无法计算的价值 | [] 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。

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

通用约定

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

目录

法律声明	. I
通用约定	. I
1 使用Procfile配置应用进程	. 1
2 Tomcat	2
2.1 项目文件夹结构设置	2
2.2 使用Web+部署Tomcat应用	3
3 Java	11
3.1 设置Java开发环境	11
3.2 使用Spring Boot开发应用	13
3.3 向Java应用的部署环境中添加RDS实例	20
4 Node.js	23
4.1 设置Node.js开发环境	23
4.2 将Express应用部署到Web+	24
4.3 向Node.js应用部署环境中添加RDS数据库实例	28
5 Go	30
5.1 设置Go开发环境	30
5.2 将Beego应用部署到Web+	.31
5.3 向Go应用的部署环境中添加RDS实例	34
6 PHP	37
6.1 设置PHP开发环境	.37
6.2 使用Laravel框架开发应用	38
6.3 使用Symfony开发应用	43
7 Python	47
7.1 设置Python开发环境	47
7.2 使用Flask开发应用	48
7.3 使用Django开发应用	50
8 ASP.NET Core	56
8.1 设置ASP.NET Core开发环境	56
8.2 部署ASP.NET Core应用至Web+	.56
9 Ruby	62
9.1 配置Ruby开发环境	62
9.2 部署Ruby on Rails应用至Web+	63
10 Native	67
10.1 部署原生应用到Web+	67

1 使用Procfile配置应用进程

通常情况下,Web+对于每种技术栈类型设置了默认的启动命令,您也可以通过配置启动命令或者 使用Procfile来指定,Web+将优先使用Procfile中的启动命令。

Procfile

如果需要使用Procfile,请在源包根目录中包含一个名为Procfile的文件,格式如下:

web: <command>

设置示例:

web: /home/admin/app/demo/startup.sh

!) 注意:

- · Procfile中的每行内容都必须符合以下正则表达式: ^[A-Za-z0-9_]+:\s*.+\$。
- ·Web+将识别开头为web:(冒号后面有一个空格)的命令,并以此作为服务的启动命令。
- ・此命令必须一直在前台运行,命令执行结束时服务即停止。
- · 启动命令的工作目录为应用部署包的根目录。
- ・启动命令将以admin用户身份执行。

2 Tomcat

2.1 项目文件夹结构设置

Tomcat服务器中使用WAR包来部署应用时,WAR包的项目结构必须符合一定的标准,一 个Web项目目录中包含Web应用程序代码和配置文件以及静态文件等。

项目文件夹结构

为了简化工程的编译和打包步骤,推荐使用下面的项目文件层次结构。



src/main/java目录下的内容包含您开发的应用程序,即未编译的java类文件,这些类将被编 译成可通过应用程序代码访问的.class文件并置于src/main/webapp/WEB-INF/classes目 录中。java类文件编译完成后,编译好的class文件将被置于webapp/WEB-INF/classes中,并 与webapp目录一并被打包部署到服务器上。

webapp根目录文件夹结构

webapp的根目录中存储了html页面,jsp页面以及静态资源等内容,这些内容将与WEB-INF一 起被打包部署到服务器上。 webapp中除了WEB-INF外的其他内容可通过客户端直接访问,如404.jsp页面和index.jsp页 面。static目录中存放css样式文件,图片文件和JavaScript文件等可由客户端访问的资源。

WEB-INF子目录文件夹结构

webapp目录下的WEB-INF的子目录,该子目录一般包含如下文件和目录,WEB-

INF是Java的WEB应用的安全目录,其中的资源客户端无法直接访问,服务端可以访问的目录中的内容,其中有如下内容:

- · classes目录中包含了您开发的应用程序源码编译后的.class文件;
- · lib目录中包含了web应用需要依赖的各种JAR文件,如数据库驱动jar文件;
- · 页面模板文件(如mustache文件);
- ・web.xml应用配置文件。
- WEB-INF
 Classes
 Claipe

 lib
 jar库文件

 views
 页面模板

 web.xml
 web应用配置文件

2.2 使用Web+部署Tomcat应用

Web+的Tomcat技术栈是一组环境配置,用于Tomcat Web容器中运行的Java Web应用。在本 教程中,您将学习如何开发一个简单的Tomcat应用,并将其部署到Web+。

准备工作

在进入本教程之前, 请确保您已经安装并配置好了以下工具和容器:

- IntelliJ IDEA
- Maven
- JDK
- Tomcat 8.5及以上版本

步骤一: 创建Tomcat Demo工程

- 1. 启动IntelliJ IDEA。
- 2. 选择File->New->Project,新建一个工程。
- 3. 选择Spring Initializr, 然后单击Next。



4. 输入工程信息,并选择打包方式为War,设置完成后单击Next。

🔛 New Project	
Project Metad	ata
<u>G</u> roup:	com.example
<u>A</u> rtifact:	demo
<u>Т</u> уре:	Maven Project (Generate a Maven based project archive)
<u>L</u> anguage:	Java 👻
Packaging:	War 🔻
<u>J</u> ava Version:	8 -
<u>V</u> ersion:	0.0.1-SNAPSHOT
Na <u>m</u> e:	demo
<u>D</u> escription:	Demo project for Spring Boot
Pac <u>k</u> age:	com.example.demo
	and the second

5. 在Dependencies页面单击Web并勾选Spring Web Starter,然后单击Next。

Dependencies Q	Spring Boot 2.1.7 ▼	Selected De
Developer Tools	Spring Web Starter	Web
Web	Spring Reactive Web	Spring V
Template Engines	Rest Repositories	
Security	Spring Session	
SOL	Rest Repositories HAL Browser	
NoSOI	Spring HATEOAS	
Moscoging	Spring Web Services	
wessaging	Jersey	
170	Spring REST Docs	
Ops	Vaadin .	سير جد است

6. 输入工程名称,并单击Finish完成创建。

🔛 New Project	
Project n <u>a</u> me:	webplus-java-demo
Project <u>l</u> ocation:	~/workspace/webplus-java-demo

步骤二:配置应用

1. 新建一个Controller类。



- 2. 参照以下步骤, 配置本地运行环境。
 - a. 在项目页面选择DemoApplication> Edit Configurations。



b. 在页面左上角单击+, 然后选择Tomcat Server > Local。

Run/Debug Configurations		
+ - E + · Name:	DemoApplication	are 🗌 Allow running in <u>p</u> arallel
In JUnit Kotlin	juration Code Coverage Logs	
Kotlin script	lass: com.example.demo.DemoApplication	
MXUnit npm	iron <u>m</u> ent	
 NW.js OpenShift Deployment OSGi Bundles 	ng Boot able <u>d</u> ebug output 🔄 <u>H</u> ide banner 📝 Enable launch optimizatio	on ✓ Enable JM <u>X</u> agent
Plugin Protractor Protractor	ng Application Update Policies n ' <u>U</u> pdate' action: Do nothing	
Remote	n frame deactivation: Do nothing	
A SAE on Alibaba Cloud Spring Boot	profiles:	
Spring dmServer	Je <u>p</u> arameters:	
🛱 Spy-js for Node.js No TestNG	Name Van	
	Add New 'Tomcat Server' Configuration Interes added.	
	ОК	Cancel Apply

c. 在Server页签配置8.5及以上版本的Tomcat。

Run/Debug Configurations	
	Name: Unnamed
 Spring Boot Tomcat Server 	Server Deployment Logs Code Coverage Startup/Connection
 Fremplates 	Application server: Tomcat 9.0.24
	Open browser
	✓ After launch Operault ✓ Image: Second
	URL: http://localhost:8080/wp_tomcat_demo_war_exploded/
	<u>V</u> M options:
	On 'Update' action: Restart server
	On <u>f</u> rame deactivation: Do nothing -
	JRE: Default (1.8 - project SDK)
	Tomcat Server Settings
	HTTP port: 8080 Deploy applications configured in Tomcat instance
	HTTPs port: Preserve sessions across restarts and redeploys
	JMX port: 1099
	OK Cancel Apply

d. 在Deployment页签选择+>Artifacts配置部署方式。

Run/Debug Configurations		
+ - 🖻 🖋 🔺 💌 📭 🐙		Share
 ▶ Spring Boot ▼ Z Tomcat Server Unnamed 	Server Deployment Logs Code Coverage Startup/Connection	
► <i>F</i> Templates	Deploy at the server startup	
		Artifact
		External Source
	Nothing to deploy	
	Warning: No artifacts marked for deployment	- Fix
?	OK Cancel	Apply

e. 在Select Artifacts to Deploy对话框中选择部署方式为war exploded,然后单击OK。



- 3. 单击调试、Run或Debug按钮,启动应用。
- 4. 在浏览器中输入http://localhost:8080来访问应用。

$\leftrightarrow \rightarrow G$	i localhost:8080
hello, webplus	:!

步骤三:打包应用

1. 单击右侧的Maven选项卡,在弹出的页面中选择Demo > Lifecycle > Package进行打包。



2. 打包完成后可在工程的target目录下看到一个war包(如demo-0.0.1-SNAPSHOT.war),接下来需要将这个应用程序部署到Web+应用中。



步骤四: 创建应用并完成部署

- 1. 登录 Web+控制台,并在页面左上角选择所需地域。
- 2. 在概览页最近更新的部署环境区域的右上角单击新建。

3. 在应用基本信息页面选择技术栈类型为Tomcat,设置应用基本信息,设置完成后单击下一步。

技术栈类型*	Tomcat 运行在Tomcat容器中的Java应用, 支持WAR和ZIP类型的邮署程序包。	Java 普通Java应用,支持FatJAR和ZIP类 型的部署程序包。	Node.js 普通Node.js应用,支持ZIP类型的部 署程序包。
	Go 编译为可执行文件的Go应用,支持 ZIP类型的部署程序包。安装有Go语 言运行时环境。	PHP 普通PHP应用,支持ZIP类型的部署 程序句。	Python 普通Python应用,支持ZIP类型的部 著程序包。
	ASP.NET Core ASP.NET Core应用,支持Razor和 MVC类型的Web应用。	Ruby Ruby应用,支持ZIP类型的部署程序 包,支持Ruby on Rails应用。	Native 原生应用,支持ZIP类型的部署程序 包。
应用名称 *	doc-test		8/64
应用描述	文档测试		4/1024
下一步			

- 在部署环境信息页面设置部署环境名称,部署包来源选择上传本地程序,上传您刚打包的demo-0.0.1-SNAPSHOT.war,设置部署包版本后单击完成创建。
- 在完成创建页面单击查看该应用或完成创建可进入应用详情页面。单击部署环境名称进入部署环境
 境详情页面,然后单击公网访问地址右侧的链接进入应用首页。



连接数据库

Tomcat类型的应用程序的连接数据库方法同Java类型的应用程序的访问数据库方式基本一

致,具体操作请参见向Java应用的部署环境中添加RDS实例,连接数据库的方法示例可参考部署包 alibabacloud-webplus-tomcat-demo。

更多信息

- · 在Web+控制台快速部署应用的视频演示请参见#unique_8。
- · 在控制台部署应用的详细配置步骤请参见#unique_9。
- ・使用CLI完成应用创建和部署的操作请参见#unique_10。
- ·完成应用托管之后的应用的管理操作请参见#unique_11。
- ·管理应用所在的部署环境的操作请参见#unique_12。

3 Java

3.1 设置Java开发环境

在本地开发环境测试Java应用,需准备好相关的开发环境。本文将介绍Java开发环境的设置步骤,并提供相关工具的安装链接。

安装Java开发工具包

选择您的开发平台,参考以下方式安装JDK。

Linux

- 1. 在Oracle官网下载二进制安装包,例如jdk-8uversion-linux-x64.tar.gz。
- 2. 进入JDK安装包所在目录。
- 3. 执行以下命令解压安装包:

tar zxvf jdk-8uversion-linux-x64.tar.gz

- 4. 按以下方式配置环境变量。
 - a. 执行以下命令打开配置文件。

vim ~/.bashrc

b. 在配置文件中添加以下内容。

```
JAVA_HOME=/Java安装路径
CLASSPATH=$JAVA_HOME/lib/
PATH=$PATH:$JAVA_HOME/bin
export PATH JAVA_HOME CLASSPATH
```

c. 执行以下命令使配置生效。

source ~/.bashrc

d. 检查是否安装成功。

```
java -version
javac -version
```

macOS

- 1. 在Oracle官网下载安装包,例如jdk-8uversion-macosx-x64.dmg。
- 2. 进入JDK安装包所在目录,双击安装包按照提示指令进行安装。

3. 按以下方式配置环境变量。

a. 执行以下命令打开配置文件。

vim ~/.bashrc

b. 在配置文件中添加以下内容。

```
JAVA_HOME=/Library/Java/JavaVirtualMachines/jdk1.8.0_151.jdk/
Contents/Home
CLASSPAHT=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
PATH=$JAVA_HOME/bin:$PATH:
export JAVA_HOME
export CLASSPATH
export PATH
```

c. 执行以下命令使配置生效。

source ~/.bashrc

d. 检查是否安装成功。

java -version javac -version

Windows

- 1. 在Oracle官网下载安装包,例如jdk-8version-windows-x64.exe。
- 2. 进入JDK安装包所在目录,双击安装包按照提示指令进行安装。
- 3. 按以下方式配置环境变量。
 - a. 设置JAVA_HOME环境变量。

```
C:\Program Files\Java\jdk1.8.0
```

b. 修改环境变量PATH,在PATH变量后添加如下内容

{**系统中原PATH环境变量**};%JAVA_HOME%\bin;%JAVA_HOME%\jre\bin

c. 添加CLASSPATH环境变量。

%JAVA_HOME%\lib;%JAVA_HOME%\lib\dt.jar;%JAVA_HOME%\lib\tools.jar;

```
d. 检查是否安装成功。
```

```
java -version
```

安装Tomcat

访问Apache Tomcat, 根据Apache Tomcat Versions的说明下载合适的Tomcat版本。

安装IDE

集成开发环境(IDE)是用于提供程序开发环境的应用程序,一般包括代码编辑器、编译器、调 试器和图形用户界面等工具。如果您还没使用IDE进行过Java开发,请根据个人开发习惯下载安 装Eclipse或IntelliJ IDEA,下载链接如下:

- Eclipse
- IntelliJ IDEA

3.2 使用Spring Boot开发应用

Spring Boot是一个轻量级框架,可以用来轻松地创建独立的、生产级的、基于Spring且能直接 运行的应用。在本教程中,您将学习如何开发一个简单的Spring Boot应用,并将其部署到Web +。

准备工作

在进入本教程之前,请确保您已经安装并配置好了以下3个工具:

- IntelliJ IDEA
- Maven
- JDK

步骤一: 创建Demo工程

- 1. 启动IntelliJ IDEA。
- 2. 选择File->New->Project,新建一个工程。

3. 对项目进行配置并完成创建。

a. 在左侧导航栏选择Spring Initializr, 然后单击Next。



b. 设置工程信息,设置完成后单击Next。

🛄 New Project	And the party of the party of the party state of the party of the part	
Project Metad	ata	
<u>G</u> roup:	webplus-java-demo	
<u>A</u> rtifact:	demo	
<u>T</u> ype:	Maven Project (Generate a Maven based project archive)	
<u>L</u> anguage:	Java 🔻	R
Packaging:	Jar 💌	
Java Version:	8 💌	
<u>V</u> ersion:	0.0.1-SNAPSHOT	
Na <u>m</u> e:	demo	1
<u>D</u> escription:	Demo project for Spring Boot	ļ
Pac <u>k</u> age:	webplusjavademo.demo	ł

c. 在Dependencies页面单击Web并勾选Spring Web Starter,然后单击Next。

Project	the service party produces to the service of the two participants	
Dependencies Q	Spring Boot 2.1.7 🔻	Selected Depen
Developer Tools	✓ Spring Web Starter	Web
Web	Spring Reactive Web	Spring Web
Template Engines	Rest Repositories	
Security	Spring Session	
SQL	Rest Repositories HAL Browser	
NoSOL	Spring HATEOAS	
Messaging	Spring Web Services	
NC SSaging	Jersey	
170	Spring REST Docs	· · · · · · · · · · · · · · · · · · ·
Ops	Vaadin	مستعرب هررا بعبرا

d. 输入工程名称,并单击Finish完成创建。

Project	
Project n <u>a</u> me:	webplus-java-demo
Project <u>l</u> ocation:	~/workspace/webplus-java-demo

e. 打开工程目录下的pom.xml文件,并在其中加入下图所示圈注部分的配置。



步骤二:配置应用

1. 新建一个Controller类。

📭 demo321332 🔪 🖿 target 🔪 📗 demo-0.0.1-SNAPS	HOT.jar
E Project 🔻 🙂 🛨 🗭	M demo 🗴 🌀 DemoApplication.java 🗴 🕜 HelloWebPlus.java 🗵
 Image: demo321332 ~/webplus-demo/demo3213 Image: begin be	<pre>3 1 package com.example123.demo; 2</pre>
	<pre>import org.springframework.web.bind.annotation.RequestMapping; import org.springframework.web.bind.annotation.RestController; @RestController public class HelloWebPlus { @RequestMapping("/") public String hello() { </pre>
target 2 ► In classes 4 ► In generated-sources 1 ► In generated-test-sources	10 return "hello, <u>webplus</u> "; 11
 Interven-activer Interven-status Interven-status Interven-status Interven-status Interven-status 	
demo-0.0.1-SNAPSHOT jar demo-0.0.1-SNAPSHOT jar.original gitignore demo321332.iml HELP.md dmwnw	

- 2. 单击调试、Run或Debug按钮,启动应用。
- 3. 在浏览器中输入localhost:8080来访问应用。

\leftarrow	\rightarrow	C	() I	ocalhost:8080/hellowp
hello	o, we	ebplus	6	

步骤三:构建可执行JAR

1. 单击右侧的Maven选项卡,在弹出的页面中单击Execute Maven Goal按钮。在弹出的对话框 中输入package命令,然后单击Execute。

	Maven	\$ -	
<pre>l version="1.0" encoding="UTF-8"?> ject xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XM xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/P</pre>	ର ଲ ± + ⊳ ଲେ # ଡ 표 ÷ ୬ ▼ ଲademo କାଳାମ ।		
<pre><modelwersion>4.0.0</modelwersion> <pre><pre></pre></pre></pre>	Gutecycle Grugins Grugins Grupendencies	1	m Maven
Execute Maven Goal C:/Users/anyu.hp/workspace/webplus-java-demo C:/Users/anyu.hp/workspace/webplus-java-demo Command line package			😵 Bean Validation
Chain	Cancel		🗯 Ant Build

2. 打包完成后可在工程的target目录下看到一个Jar包(如demo-0.0.1-SNAPSHOT.jar),接 下来需要将这个应用程序部署到Web+应用中。

 main java jeve java jeve plusjavademo.demo DemoApplication HelloWebPlus resources test target classes generated-sources generated-test-sources generated-test-sources maven-archiver maven-status surefire-reports test-classes demo-0.0.1-SNAPSHOT.jar.original gitignore HELP.md 	•	webplus-java-demo ~/workspace/webplus-java-demo idea .mvn src
 target classes generated-sources generated-test-sources maven-archiver maven-status surefire-reports test-classes demo-0.0.1-SNAPSHOT.jar demo-0.0.1-SNAPSHOT.jar.original gitignore HELP.md 		 main java webplusjavademo.demo DemoApplication HelloWebPlus resources test
i mvnw	•	 target classes generated-sources generated-test-sources maven-archiver maven-status surefire-reports test-classes demo-0.0.1-SNAPSHOT.jar demo-0.0.1-SNAPSHOT.jar.original gitignore HELP.md mvnw

步骤四: 创建应用并完成部署

- 1. 登录 Web+控制台,并在页面左上角选择所需地域。
- 2. 在概览页最近更新的部署环境区域的右上角单击新建。

- 技术栈类型 * Tomcat Java Node.js ((IS 8 运行在Tomcat容器中的Java应用 普通Java应用,支持FatJAR和ZIP类 普通Node.js应用,支持ZIP类型的部 0 ę 支持WAR和ZIP类型的部署程序包。 型的部署程序包。 署程序包。 Go PHP Python) (编译为可执行文件的Go应用,支持 (php) 普通PHP应用,支持ZIP类型的部署 普通Python应用 , 支持ZIP类型的部 ZIP类型的部署程序包。安装有Go语 程序句. 署程序句. 言运行时环境。 Native 原生应用,支持ZIP类型的部署程序 句.. 应用名称 * 8/64 doc-test 文档测试 应用描述 4/1024
- 3. 在应用基本信息页面选择技术栈类型为Java,设置应用基本信息,设置完成后单击下一步。

 在部署环境信息页面设置部署环境名称,部署包来源选择上传本地程序,上传您刚打包的demo-0.0.1-SNAPSHOT.jar,设置部署包版本后单击完成创建。

✓ 应用基本信息	2 部署环境信息	3 配置
技术栈版本 🚯 *	★ Java 8 / Aliyun Linux 2.1903	~
部署环境名称*	请输入您想要创建的部署环境名称,支持大小写字母、数字、"_"和"-",长度不超过64个字符。	0/64
部署环境描述	请输入一段描述信息帮助您识别这个部署环境,长度不超过1024个字符。	
		0/1024
部署包来源*	 上传本地程序 使用样例程序 	
上传文件 *	选择文件	
部署包版本 *	20190813.161221	15/64
版本描述	请输入一段描述信息帮助您识别这个版本,长度不超过1024个字符。	
		0/1024
上一步下一步	完成的现象	

在完成创建页面单击查看该应用或完成创建可进入应用详情页面。单击部署环境名称进入部署环境
 境详情页面,然后单击公网访问地址右侧的链接进入应用首页。



更多信息

- ・ 在Web+控制台快速部署应用的视频演示请参见#unique_8。
- · 在控制台部署应用的详细配置步骤请参见#unique_9。
- ·使用CLI完成应用创建和部署的操作请参见#unique_10。
- ·完成应用托管之后的应用的管理操作请参见#unique_11。
- ·管理应用所在的部署环境的操作请参见#unique_12。

3.3 向Java应用的部署环境中添加RDS实例

您可以使用阿里云云数据库RDS的数据库实例来存储应用中需持久保存的数据。本文以给基 于Spring Boot框架开发的Java应用添加RDS MySQL数据库为例,展示如何为Java应用添加数据 库并验证应用与数据库是否连接。

环境变量

Web+会将数据库连接的相关信息存放在环境变量中,以便应用进行读取,相关环境变量请参考下表。

变量名	变量值	变量说明
WP_RDS_ENGINE	MySQL	RDS数据库引擎
WP_RDS_CON NECTION_ADDRESS	rm-***.mysql.rds.aliyuncs .com	RDS内网连接地址
WP_RDS_PORT	3306	RDS端口号
WP_RDS_ACC OUNT_NAME	webplus	RDS数据库账号名称
WP_RDS_ACC OUNT_PASSWORD	自定义	RDS账号密码
WP_RDS_DATABASE	webplus	RDS数据库

添加依赖和修改配置文件

1. 打开SpringBoot工程中的pom.xml文件,添加JDBC依赖和MySQL依赖。

</dependency>

2. 打开工程中的application.properties配置文件,您可以按以下方式使用环境变量配置

JDBC的连接参数,其中NONE可以修改为默认数值。

```
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.datasource.url=jdbc:mysql://${WP_RDS_CONNECTION_ADDRESS:NONE
}:${WP_RDS_PORT:3306}/rdsitem?useUnicode=true&characterEncoding=utf-
8
spring.datasource.username=${WP_RDS_ACCOUNT_NAME:NONE}
spring.datasource.password=${WP_RDS_ACCOUNT_PASSWORD:NONE}
```

连接数据库

修改依赖和配置后,启动应用时SpringBoot会根据配置文件自动连接数据库,下面的代码片段展示了如何在Web+中的应用访问数据库。

```
@Autowired
private JdbcTemplate jdbcTemplate;
// 从数据库item中查询
public List<Item> fetchItems() {
    final String sql="select id,title,completed from item";
    RowMapper<Item> rowMapper=new BeanPropertyRowMapper<>(Item.
class);
    return jdbcTemplate.query(sql, rowMapper);
}
```

其中, Item类定义如下:

```
public class Item {
    private String id;
    private String title;
    private boolean completed;
    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getTitle() {
        return title;
    }
    public void setTitle(String title) {
        this.title = title;
    }
    public boolean getCompleted() {
        return completed;
    }
    public void setCompleted(boolean completed) {
        this.completed = completed;
    Item() {
    @Override
    public String toString() {
        return id + " " + title + " " + completed;
    }
```

}

4 Node.js

4.1 设置Node.js开发环境

在本地开发环境测试Node.js应用,需要准备相关的开发环境。本文将介绍Node.js开发环境的设置步骤,并提供相关工具的安装页面链接。

安装Node.js

在Node.js官方网站下载安装包。



为了和Web+的技术栈版本更好兼容,建议您下载Node.js 10.16.x或Node.js 8.16.x。

Linux

1. 进入Node.js安装包(例如node-v10.16.3-linux-x64.tar.xz)所在目录,执行以下命令将安 装包解压到/usr/local目录下。

sudo tar -C /usr/local -xzf node-v10.16.3-linux-x64.tar.xz

2. 执行以下命令创建软链接/usr/local/node指向刚解压的安装包路径。

sudo ln -s /usr/local/node-v10.16.3-linux-x64 /usr/local/node

3. 解压后将可执行文件目录配置到Path环境变量,将以下命令添加到\$HOME/.profile。

export PATH=\$PATH:/usr/local/node/bin

4. 执行以下命令使环境变量立即生效。

source \$HOME/.profile

5. 使用以下命令验证Node.js是否安装成功。

node --version && npm --version

如果显示如下信息,则说明安装包已成功安装。

v10.16.0 6.9.0

macOS

执行以下命令使用brew来快速安装Node.js。

brew update && brew install node

Windows

进入Node.js安装包所在目录,运行下载的.msi文件即可安装,无需其他配置。

安装IDE

集成开发环境(IDE, Integrated Development Environment)是用于提供应用开发环境的 应用程序,一般包括代码编辑器、编译器、调试器和图形用户界面等工具,可以显著提高开发效 率。以下是Node.js开发中常用的IDE,但下列IDE可能需要通过安装插件来支持Node.js开发。

```
    Visual Studio Code
```

• Atom

· WebStorm (商业软件)

4.2 将Express应用部署到Web+

Express是一个快速开发Node.js应用的Web框架,可以用来快速开发API、Web、后端服务等各种应用。本文档介绍了如何开发一个简单的Express应用,并将其部署至Web+。

前提条件

已配置好了Node.js的开发环境,相关操作请参见设置Node.js开发环境。

步骤一:安装express-generator

本文将使用express-generator来快速生成Express项目。请执行以下命令安装express-generator。

```
npm install -g express-generator
```

▋ 说明:

如果您安装了Node.js 8.2.0及以上版本,可跳过此步在创建应用时直接使用npx命令运行express-generator。

步骤二: 创建应用

执行以下命令创建名为webplus-express-app的应用。

express webplus-express-app



如果您安装了Node.js 8.2.0及以上版本,可执行npx express-generator webplus-

express-app命令直接运行express-generator而无需安装。

执行上述命令会创建一个名为webplus-express-app的目录,结构如下:



步骤三:安装本地依赖

1. 执行以下命令进入创建好的应用目录。

cd webplus-express-app

2. 执行以下命令安装本地依赖。

npm install

步骤四: 在本地运行应用

1. 执行以下命令在本地运行应用,以验证其是否可以正常工作。

npm start

当您看到命令行页面上显示以下信息时,则表示应用启动成功。

> webplus-express-app@0.0.0 start /home/admin/webplus-express-app

- > node ./bin/www
- 2. 查看运行结果:
 - ・ 在浏览器中输入http://localhost:3000来访问应用。

	localhost	
Express		
Welcome to Express		

・执行curl http://localhost:3000命令, 查看返回的运行结果:

] 说明:

为方便查看,此处对返回的信息进行了格式化,原始结果是一整行文本。

- 3. 查看应用的运行结果之后,可以使用CTRL+C停止服务。
- 步骤五:打包应用

执行以下命令将上面生成的应用的项目工程打包。

zip -r webplus-express-app.zip .

! 注意:

生成的部署包需包含node_modules目录,且压缩包不能包含第一级目录,打包示例如下:

Name Modified Size Kind Packed Attributes Inde bin Today, 09:57 2 KB Folder 712 B drwxr- node_modules Today, 10:01 6.1 MB Folder 1.8 MB drwxr- public Today, 09:57 111 B Folder 107 B drwxr- package-lock.json Today, 09:57 307 B JSON Document 7 KB rw-r-r-r package.json Today, 09:57 307 B JSON Document 183 B rw-r-r-r	Archive						
Today, 09:57 2 KB Folder 712 B drwxr- Image: node_modules Today, 10:01 6.1 MB Folder 1.8 MB drwxr- Image: node_modules Today, 09:57 111 B Folder 107 B drwxr- Image: node_modules Today, 09:57 111 B Folder 107 B drwxr- Image: node_modules Today, 10:01 26 KB JSON Document 7 KB -rw-rr Image: node_json Today, 09:57 307 B JSON Document 183 B -rw-rr	Name	Modified	Size	Kind	Packed	Attributes	Inde
Image: modules Today, 10:01 6.1 MB Folder 1.8 MB drwxr- Image: modules Today, 09:57 111 B Folder 107 B drwxr- Image: modules Today, 09:57 111 B Folder 107 B drwxr- Image: modules Today, 10:01 26 KB JSON Document 7 KB -rw-rr Image: modules Today, 09:57 307 B JSON Document 183 B -rw-rr	▶ 💼 bin	Today, 09:57	2 KB	Folder	712 B	drwxr-	
Depublic Today, 09:57 111 B Folder 107 B drwxr- D package-lock.json Today, 10:01 26 KB JSON Document 7 KB -rw-rr D package.json Today, 09:57 307 B JSON Document 183 B -rw-rr	▶ 💼 node_modules	Today, 10:01	6.1 MB	Folder	1.8 MB	drwxr-	
package-lock.json Today, 10:01 26 KB JSON Document 7 KB -rw-rr package.json Today, 09:57 307 B JSON Document 183 B -rw-rr	▶ 💼 public	Today, 09:57	111 B	Folder	107 B	drwxr-	
Dpackage.json Today, 09:57 307 B JSON Document 183 B -rw-rr	package-lock.json	Today, 10:01	26 KB	JSON Document	7 KB	-rw-rr	
	package.json	Today, 09:57	307 B	JSON Document	183 B	-rw-rr	
▶ views Today, 09:57 275 B Folder 229 B drwxr-	▶ 💼 views	Today, 09:57	275 B	Folder	229 B	drwxr-	
Today, 09:57 408 B Folder 299 B drwxr-	▶ i routes	Today, 09:57	408 B	Folder	299 B	drwxr-	
P app.js Today, 09:57 1 KB JavaScript Source File 469 B -rw-rr	🝸 app.js	Today, 09:57	1 KB	JavaScript Source File	469 B	-rw-rr	

步骤六:将应用部署至Web+

1. 登录 Web+控制台,并在页面左上角选择所需地域。

- 2. 在概览页最近更新的部署环境区域的右上角单击新建。
- 3. 在应用基本信息页面选择技术栈类型为Node.js,设置应用基本信息,设置完成后单击下一步。

技术栈类型*	Tomcat 运行在Tomcat容器中的Java应用, 支持WAR和ZIP类型的部署程序包。	Java 普通Java应用,支持FatJAR和ZIP类 型的邮署程序包。	Node.js 普通Node.js应用,支持ZIP类型的部 署程序包。
	Go 编译为可执行文件的Go应用,支持 ZP类型的部署程序包。安装有Go语 言运行时环境。	PHP 普通PHP应用,支持ZIP类型的部署 程序包。	Python 普通Python应用,支持ZIP类型的部 署程序包。
	ASP.NET Core ASP.NET Core应用,支持Razor和 MVC类型的Web应用。	Ruby Ruby应用,支持ZIP类型的部署程序 包,支持Ruby on Rails应用。	Native 原生应用,支持ZIP类型的部署程序 包。
应用名称 *	doc-test		9/64
应用描述	文档测试		4/1024
下一步			

- 4. 在部署环境信息页面设置部署环境名称,部署包来源选择上传本地程序,上传您刚打包的 webplus-express-app.zip,设置部署包版本后单击完成创建。
- 在完成创建页面单击查看该应用或完成创建可进入应用详情页面。单击部署环境名称进入部署环境
 境详情页面,然后单击公网访问地址右侧的链接进入应用首页。

	Not Secure — C
Express	
Welcome to Express	

更多信息

- · 在Web+控制台快速部署应用的视频演示请参见#unique_8。
- ・在控制台部署应用的详细配置步骤请参见#unique_9。
- ·使用CLI完成应用创建和部署的操作请参见#unique_10。
- ·完成应用托管之后的应用的管理操作请参见#unique_11。
- ·管理应用所在的部署环境的操作请参见#unique_12。

4.3 向Node.js应用部署环境中添加RDS数据库实例

您可以添加RDS数据库实例来存储应用中需持久保存的数据。本文在将Express应用部署到Web+的基础上,展示如何为Node.js应用添加数据库并读写其中的数据。

环境变量

Web+会将数据库连接的相关信息存放在环境变量中,以便应用进行读取,相关环境变量请参考下 表。

变量名	变量值	变量说明
WP_RDS_ENGINE	MySQL	RDS数据库引擎
WP_RDS_CON NECTION_ADDRESS	rm-***.mysql.rds.aliyuncs .com	RDS内网连接地址
WP_RDS_PORT	3306	RDS端口号
WP_RDS_ACC OUNT_NAME	webplus	RDS数据库账号名称
WP_RDS_ACC OUNT_PASSWORD	自定义	RDS账号密码
WP_RDS_DATABASE	webplus	RDS数据库

添加数据库驱动

进入应用的项目工程目录,例如进入将Express应用部署到Web+应用的webplus-express-app目

录,执行以下命令添加MySQL数据库驱动。

npm install mysql

添加数据库访问功能

打开routes/users.js文件, 修改如下:

```
var express = require('express');
var mysql = require('mysql');
var router = express.Router();
router.get('/', function(req, res, next) {
  var connection = mysql.createConnection({
    host: process.env.WP_RDS_CONNECTION_ADDRESS,
    user: process.env.WP_RDS_ACCOUNT_NAME,
    password: process.env.WP_RDS_ACCOUNT_PASSWORD,
    database: process.env.WP_RDS_DATABASE
  });
  connection.connect();
```

```
connection.query('SELECT "Tom" AS user_name', function(error,
results) {
    if (error) {
        throw error;
    }
    res.send('User name queried from database: ' + results[0].
    user_name);
    });
});
module.exports = router;
```

更多信息

关于如何使用Web+来管理RDS,可参考#unique_20。

5 Go

5.1 设置Go开发环境

在本地开发环境测试Go应用,需准备好相关的开发环境。本文将介绍Go开发环境的设置步骤,并 提供相关工具的安装页面链接。

安装Go

根据您的开发平台,从Go官方网站下载对应的版本,对于各个主流平台,Go都有很完善的支持,下载完成后请分别参考以下方式安装。

Linux

1. 使用以下命令将安装包解压到/usr/local/go目录下:

tar -C /usr/local -xzf go\$VERSION.\$0S-\$ARCH.tar.gz

2. 解压后将可执行文件目录配置到PATH环境变量,将以下命令添加到/etc/profile或\$HOME/. profile。

export PATH=\$PATH:/usr/local/go/bin

3. 执行以下命令使环境变量立即生效。

source \$HOME/.profile

macOS

执行以下命令使用brew来快速安装Go。

brew update && brew install go

Windows

进入Go安装包所在目录,运行下载的.msi文件即可安装,无需其他配置。

配置GOPATH

环境变量GOPATH标识了工作区的目录,通常情况下您需要设定此环境变量来指定工作区目录。各 个平台对应设置方法请参见文档Setting GOPATH。

安装IDE

集成开发环境(IDE, Integrated Development Environment)是用于提供应用开发环境的 应用程序,一般包括代码编辑器、编译器、调试器和图形用户界面等工具,可以显著提高开发效 率。以下是Go开发中常用的IDE, 但下列IDE可能需要通过安装插件来支持Go开发。

- Eclipse
- Visual Studio Code
- ・ GoLand (商业软件)

5.2 将Beego应用部署到Web+

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、后端服务等各种应用。本文档介绍了如何开发一个简单的Beego应用,并将其部署至Web+。

步骤一:安装Beego

1. 执行以下命令来安装Beego和工具bee。

```
go get -u github.com/astaxie/beego
go get -u github.com/beego/bee
```

2. 执行以下命令将\$GOPATH/bin目录加入\$PATH环境变量。

```
echo 'export PATH="$GOPATH/bin:$PATH"' >> ~/.profile
source >> ~/.profile
```

步骤二: 创建应用

1. 打开终端,进入\$GOPATH/src所在的目录,执行以下命令来快速创建一个命名

为webplusdemo的项目。

bee new webplusdemo

上述命令会创建一个名为webplusdemo的目录,结构如下:

```
webplusdemo
    conf
        app.conf
        controllers
        default.go
        main.go
        models
        routers
        router.go
        static
        css
        img
        js
        reload.min.js
```



2. 进入项目目录,执行bee run。

the second se	
Ι λ	
_//	
\ / _ \ / _ \	
_// / /	
└/ \ \ v1.10.0	
2019/07/31 12:54:33 INFO	▶ 0001 Using 'webplusdemo' as 'appname'
2019/07/31 12:54:33 INFO	▶ 0002 Initializing watcher
2019/07/31 12:54:35 SUCCESS	▶ 0003 Built Successfully!
2019/07/31 12:54:35 INFO	▶ 0004 Restarting 'webplusdemo'
2019/07/31 12:54:35 SUCCESS	▶ 0005 './webplusdemo' is running
2019/07/31 12:54:35.976 [I]	[asm_amd64.s:1337] http server Running on http://:8080

3. 打开浏览器, 输入http://localhost:8080来访问项目。



步骤三:打包应用

1. 在项目目录下新建Procfile来给Go应用指定启动命令,填写以下内容,并保存。

web: chmod +x webplusdemo; ./webplusdemo

2. 使用bee工具来对打包工程。

bee pack -be GOOS=linux -be GOARCH=amd64 -f zip

执行完上述命令,将会在项目目录下生成一个名为webplusdemo.zip的压缩包,按照下述步骤 来将该Go应用部署至Web+。
步骤四: 创建应用并完成部署

- 1. 登录 Web+控制台,并在页面左上角选择所需地域。
- 2. 在概览页最近更新的部署环境区域的右上角单击新建。
- 3. 在应用基本信息页面选择技术栈类型为Go,设置应用基本信息,设置完成后单击下一步。

技术栈美型。	Tomcat 运行在Tomcat容器中的Java应用, 支持WAR和ZIP类型的部署程序包。	Java 普通Java应用,支持FatJAR和ZIP类 型的部署程序包。	Node.js 普通Node.js应用,支持ZIP类型的部 署程序包。
	Go 编译为可执行文件的Go应用,支持 ZIP类型的部署程序包。安装有Go语 言运行时环境。	PHP 普通PHP应用,支持ZIP类型的部署 程序包。	Python 普通Python应用,支持ZIP类型的部 署程序包。
	Native 原生应用,支持ZIP类型的部署程序 包。		
应用名称 *	请输入您想要创建的应用名称,支持大小写字母、数4	字、"_"和"-",长度不超过64个字符。	0/64
应用描述	请输入一段描述信息帮助您识别这个应用,长度不超过	过1024个字符。	
			0/1024
下一步			

- 4. 在部署环境信息页面设置部署环境名称,部署包来源选择上传本地程序,上传您刚打包的 webplusdemo.zip,设置部署包版本后单击完成创建。
- 在完成创建页面单击查看该应用或完成创建可进入应用详情页面。单击部署环境名称进入部署环境
 境详情页面,然后单击公网访问地址右侧的链接进入应用首页。



常见问题

健康检查失败导致网站无法访问怎么办?

如果使用SLB,请在健康检查URL的controller中接受head请求,否则可能会导致健康检查失

败,网站无法访问。

```
package controllers
import (
    "github.com/astaxie/beego"
)
type MainController struct {
    beego.Controller
}
func (c *MainController) Get() {
    c.Data["Website"] = "beego.me"
    c.Data["Email"] = "astaxie@gmail.com"
    c.TplName = "index.tpl"
}
func (c *MainController) Head() {
    c.Ctx.Output.Body([]byte(""))
}
```

更多信息

- · 在Web+控制台快速部署应用的视频演示请参见#unique_8。
- · 在控制台部署应用的详细配置步骤请参见#unique_9。
- ·使用CLI完成应用创建和部署的操作请参见#unique_10。
- ·完成应用托管之后的应用的管理操作请参见#unique_11。
- ·管理应用所在的部署环境的操作请参见#unique_12。

5.3 向Go应用的部署环境中添加RDS实例

您可以使用阿里云云数据库RDS的数据库实例来存储应用中需持久保存的数据。本文将介绍如何 为Go应用添加数据库并验证应用与数据库是否连接。

环境变量

Web+会将数据库连接的相关信息存放在环境变量中,以便应用进行读取,相关环境变量请参考下表。

变量名	变量值	变量说明
WP_RDS_ENGINE	MySQL	RDS数据库引擎

变量名	变量值	变量说明
WP_RDS_CON NECTION_ADDRESS	rm-***.mysql.rds.aliyuncs .com	RDS内网连接地址
WP_RDS_PORT	3306	RDS端口号
WP_RDS_ACC OUNT_NAME	webplus	RDS数据库账号名称
WP_RDS_ACC OUNT_PASSWORD	****	RDS账号密码
WP_RDS_DATABASE	webplus	RDS数据库

安装数据库驱动

执行以下命令,安装MySQL数据库驱动:

go get github.com/go-sql-driver/mysql

添加数据库

添加数据库的操作请参照以下样例代码进行配置。

```
package main
import (
  "database/sql"
  "fmt"
   "github.com/go-sql-driver/mysql"
  "os"
)
func main() {
  user := os.Getenv("WP_RDS_ACCOUNT_NAME")
  passwd := os.Getenv("WP_RDS_ACCOUNT_PASSWORD")
  host := os.Getenv("WP_RDS_CONNECTION_ADDRESS")
  port := os.Getenv("WP_RDS_PORT")
  connStr := fmt.Sprintf("%s:%s@tcp(%s:%s)/?timeout=30s", user, passwd
, host, port)
 db, _ := sql.Open("mysql", connStr)
defer db.Close()
  sqlTxt := "select 'OK' as result"
  rows, _ := db.Query(sqlTxt)
  var result string
  for rows.Next(){
    _ = rows.Scan(&result)
  }
  // output "OK"
  fmt.Println(result)
```

}

更多信息

关于如何使用Web+来管理RDS,可参考#unique_20。

6 PHP

6.1 设置PHP开发环境

在本地开发环境测试PHP应用,需准备好相关的开发环境。本文将介绍PHP开发环境的设置步骤,并提供相关工具的安装页面链接。

安装PHP

请根据以下操作安装PHP和一些常用扩展。如果您没有特别的要求,请获取最新版本。

Linux

1. 在PHP官网下载安装包,例如php-7.3.8.tar.bz2 (sig)。

2. 进入安装包所在目录。

- 3. 执行以下命令安装包:
 - \$ sudo yum install php

macOS

- 1. 在PHP官网下载安装包,例如php-7.3.8.tar.bz2 (sig)。
- 2. 执行以下命令安装包:
 - \$ brew install php

Windows

- 1. 在PHP For Windows下载Windows系统安装包,例如PHP 7.3 (7.3.8)。
- 2. 进入PHP安装包所在目录,运行下载的文件即可安装,无需其他配置。

安装Composer

Composer 是用于PHP的依赖项管理器。您可以使用它来安装库、跟踪应用程序的依赖项并为热 门PHP框架生成项目。 1. 使用来自getcomposer.org的PHP脚本安装Composer。

\$ curl -s https://getcomposer.org/installer | php

2. 安装程序将在当前目录中生成PHAR文件。将此文件移动到环境PATH中的位置以便将此文件用 作可执行文件。

\$ mv composer.phar ~/.local/bin/composer

3. 使用require命令安装库。

\$ composer require twig/twig

Composer 会将您在本地安装的库添加到您的项目 *composer.json*文件。在部署项目代码 时,Web+将使用Composer在您的环境中的应用实例上安装此文件中列出的库。如果您在安 装Composer时遇到问题,请访问*Composer*官网。

安装IDE

集成开发环境(IDE,Integrated Development Environment)是用于提供应用开发环境的 应用程序,一般包括代码编辑器、编译器、调试器和图形用户界面等工具,可以显著提高开发效 率。以下是PHP开发中常用的IDE。

• Eclipse

• PhpStorm

6.2 使用Laravel框架开发应用

Laravel是一套简洁、优雅的PHP Web开发框架。本文档将演示如何使用Laravel创建一个应用和 添加MySQL数据库,并将其部署到Web+上。

前提条件

· 设置PHP开发环境。

(!) 注意:

使用Laravel框架开发应用需要PHP 5.5.9或更高版本。

步骤一: 创建应用

执行以下命令使用Composer工具来创建一个名为webplusdemo的项目,该过程可能需要几分钟。

composer create-project --prefer-dist laravel/laravel webplusdemo

2. 执行以下命令使用PHP内置的开发服务器来运行此项目。

php artisan serve

3. 打开浏览器输入下图地址访问应用。

~ -	→ C	(i) localhos	t:8000						
				1 21	rav				
				Lai	av	CI			
		DOCS	LARACASTS	NEWS	BLOG	NOVA	FORGE	GITHUB	

步骤二:打包应用

1. 进入项目目录,执行以下命令来激活Laravel内置的用户权限管理功能。

php artisan make:auth

2. 修改.env文件中数据库相关的配置,使其关联Web+的相关环境变量。

```
DB_CONNECTION=mysql
DB_HOST=${WP_RDS_CONNECTION_ADDRESS}
DB_PORT=${WP_RDS_PORT}
DB_DATABASE=${WP_RDS_DATABASE}
DB_USERNAME=${WP_RDS_ACCOUNT_NAME}
DB_PASSWORD=${WP_RDS_ACCOUNT_PASSWORD}
```

3. 执行以下命令,安装所有的依赖。

```
composer install
```

4. 使用zip命令打包项目下所有的内容生成压缩包webplusdemo.zip。

```
zip -r webplusdemo.zip ./
```

步骤三: 部署应用至Web+

1. 登录 Web+控制台,并在页面左上角选择所需地域。

- 2. 在概览页最近更新的部署环境区域的右上角单击新建。
- 3. 在应用基本信息页面选择技术栈类型为PHP,设置应用基本信息,设置完成后单击下一步。

技术栈类型*	Tomcat 运行在Tomcat容器中的Java应用, 支持WAR和ZIP类型的部署程序包。 Java 普通Java应用,支持FatJAR和ZIP类型的部署程序包。	Node.js 普通Node.js应用,支持ZIP类型的部 署程序包。
	Go PHP 编译为可执行文件的Go应用,支持 ZIP类型的部署程序包。安装有Go语 言运行时环境。 程序包。	Python 普通Python应用,支持ZIP类型的部 署程序包。
	Native 原生应用,支持ZIP类型的部署程序 包。	
应用名称 *	doc-test	8/64
应用描述	请输入一段描述信息帮助您识别这个应用,长度不超过1024个字符。	
		0/1024
下一步		

4. 在部署环境信息页面设置部署环境名称,部署包来源选择上传本地程序,上传您刚打包的 webplusdemo.zip,设置部署包版本后单击下一步。

技术栈版本	★ Python 3.7.4 / Aliyun Linux 2.1903(最新版本)
部署环境名称 *	请输入您想要创建的部署环境名称,支持大小写字母、数字、"_"和"-",长度不超过64个字符。
部署环境描述	请输入一段描述信息帮助您识别这个部署环境,长度不超过1024个字符。
部署包来源*	 上传本地程序 使用样例程序
上传文件 *	选择文件
部署包版本 *	20190819.174715
版本描述	请输入一段描述信息帮助您识别这个版本,长度不超过1024个字符。
上—————————————————————————————————————	完成创建

5. 在配置页面选择预设配置为自定义模式。

→ 应用基本信息 -	
应用名称	webplusdemo
部署环境名称	webplusdemo
* 预设配置	○ 低成本 低成本配置仅包含一台在当前可用区中可购买的小规格的ECS实例。
	○ 高可用 高可用配置包含在当前可用区中可购买的两台小规格的ECS实例和一台性能共享型的SLB实例。
	● 自定义 支配置将允许您按照需求自定义部署环境中的资源和软件。
	平台

6. 展开云数据库RDS,按图所示配置云数据库类型为MySQL,并选择数据库版本、系列和类型等数据库基本信息。

~ 云数据库RDS NEW				
开启云数据库RDS 阿里云关系型数据库(Relational Database Service,简称RDS)是一种稳定可靠、可弹性伸缩的在线数据库服务,支持MySQL、SQL Server、PostgreSQL、PPAS和 MariaDB TX引擎,并且提供了容灾、备份、恢复、监控、迁移等方面的全套解决方案。详情可参考什么是云数据库RDS。				
实例来源 "代购"实例为Web+帮你购买及维护的实例,"导入"实例为您自己购买并维护的实例				
● 代购 ○ 导入				
数据库类型 RDS支持MySQL、PostgreSQL、SQL Server、MariaDB TX和PPAS(Postgre Plus Advanced Server,高度兼容Oracle数据库)等数据库类型。				
MySQL PostgreSQL SQLServer MariaDB PPAS				
数据库版本 请根据您的业务需要选择适合的数据库引擎版本 8.0 5.7 5.6 5.5				
系列 云数据库RDS的实例包括四个系列:基础版、高可用版、集群版和金融版。具体详情可参考产品系列概述。 高可用版 基础版				
存储类型 为满足不同场景的需求,云数据库RDS提供三种数据存储类型:本地SSD盘、SSD云盘和ESSD云盘。推荐使用本地SSD盘将数据存储于本地SSD盘,可以降低I/O延时。具体 可参考存储类型。				
本地SSD盘 (推荐) SSD云盘 ESSD云盘				

7. 展开生命周期挂钩,在PostPrepareApp编辑框内输入以下内容。

cd \$APP_HOME && /usr/local/php/bin/php artisan migrate

8. 在配置页面最下方单击完成创建。

在完成创建页面单击查看该应用或完成创建可进入应用详情页面。单击部署环境名称进入部署环境
 境详情页面,然后单击公网访问地址右侧的链接进入应用首页。



更多信息

- ・关于如何使用Web+来管理RDS,可参考云数据库RDS。
- · 在控制台部署应用的详细配置步骤请参见<mark>部署应用</mark>。
- ·使用CLI完成应用创建和部署的操作请参见在CLI快速部署应用。

6.3 使用Symfony开发应用

Symfony是一个基于MVC模式的面向对象的PHP框架,本文档将演示如何使用Symfony创建一个应用,并将其部署到Web+上。

前提条件

· 设置PHP开发环境。

(!) 注意:

Symfony 3需要PHP 5.5.9或更高版本,以及PHP的intl 扩展。

步骤一:安装Symfony

1. 执行以下命令安装Symfony CLI。

curl -sS https://get.symfony.com/cli/installer | bash

2. 将CLI的可执行文件移动到系统命令目录下。

mv ~/.symfony/bin/symfony /usr/local/bin/symfony

步骤二: 创建应用

1. 执行以下命令使用CLI工具快速创建一个Symfony演示项目。

symfony new --demo webplusdemo

创建过程需要几分钟,创建完成后将生成名为webplusdemo的项目。

2. 进入项目目录执行以下命令安装依赖。

composer install

3. 执行以下命令,使用CLI工具附带的开发服务器启动服务。

symfony server:start

4. 打开浏览器访问http://localhost:8000,进入Symfony示例应用首页。

O localhost:8000/zh_CN	*
欢迎使用 Symf	ony 示例 应用
浏览示例应用的 公共部分 . ♣ 浏览博客前台	浏览示例应用的 管理后台 . ▲ 浏览管理后台

步骤三:打包应用

1. Web+支持您使用Nginx或Apache来作为Web服务器,若您希望使用Apache,可通过在项目 目录下执行以下命令生成.htaccess文件,如果使用Nginx,则可跳过此步骤。

composer require symfony/apache-pack

2. 执行以下命令完成应用打包, 生成部署包文件webplusdemo.zip。

```
zip -r webplusdemo.zip ./
```

步骤四:部署应用至Web+

- 1. 登录 Web+控制台,并在页面左上角选择所需地域。
- 2. 在概览页最近更新的部署环境区域的右上角单击新建。

3. 在应用基本信息页面选择技术栈类型为PHP,设置应用基本信息,设置完成后单击下一步。

技术栈类型*	Tomcat 运行在Tomcat容器中的Java应用, 支持WAR和ZIP类型的部署程序包。 Java 普通Java应用,支持FatJAR和ZIP类型的部署程序包。 Nodejs 普通Nodejs应用,支持ZIP类型的部
	Go PHP 編译为可执行文件的Go应用,支持 ZP类型的部署程序包。安装有Go语 言运行时环境。 PHP 普通PHP应用,支持ZIP类型的部署 程序包。
	Native 原生应用,支持ZIP类型的部署程序 包。
应用名称 *	doc-test 8/64
应用描述	请输入一段描述信息帮助您识别这个应用,长度不超过1024个字符。
	0/1024
下一步	

4. 在部署环境信息页面设置部署环境名称,部署包来源选择上传本地程序,上传您刚打包的 webplusdemo.zip,设置部署包版本后单击完成创建。

技术栈版本 🚹 *	★ Python 3.7.4 / Aliyun Linux 2.1903(最新版本)
部署环境名称 *	请输入您想要创建的部署环境名称,支持大小写字母、数字、"_"和"-",长度不超过64个字符。
部署环境描述	请输入一段描述信息帮助您识别这个部署环境,长度不超过1024个字符。
部署包来源*	 上传本地程序 使用样例程序
上传文件 *	选择文件
部署包版本 *	20190819.174715
版本描述	请输入一段描述信息帮助您识别这个版本,长度不超过1024个字符。
上一步下一步	完成创建

5. 在完成创建页面单击查看该应用或完成创建可进入应用详情页面。单击部署环境名称进入部署环境 境详情页面,然后单击公网访问地址右侧的链接进入应用首页。

→ C ① 不安全 1 0/zh_Cl			\$
	欢迎使用 Symfo	ony 示例 应用	
浏览示	列应用的 公共部分 . 3 浏览博客前台	浏览示例应用的 管理后台 . ▲ 浏览管理后台	

更多信息

- · 在控制台部署应用的详细配置步骤请参见#unique_9。
- ・使用CLI完成应用创建和部署的操作请参见#unique_10。
- ·想了解更多Symfony信息,请进入Symfony官方网站查看。

7 Python

7.1 设置Python开发环境

在本地开发环境测试Python应用,需准备好相关的开发环境。本文将介绍Python开发环境的设置 步骤,并提供相关工具的安装页面链接。

安装Python

根据您的开发平台,从Python官网下载安装包。



为了和Web+的技术栈版本更好兼容,建议您下载Python 3.7.4或Python 2.7.16。

Linux

1. 进入Python安装包 (例如Python-3.7.4.tgz) 所在目录,执行以下命令解压安装包。

tar xvf Python-3.7.4.tgz

2. 进入Python安装包所在目录,执行以下命令进行编译和安装。

```
./configure --with-ensurepip=install
make && make install
```

macOS

执行以下命令来使用brew来快速安装Python。

brew update && brew install python

Windows

进入Python安装包所在目录,双击安装包按照提示指令进行安装。

虚拟环境

当同时开发多个项目时,使用Python虚拟环境来隔离这些项目是比较好的实践,这可以让您避免 项目之间的依赖包的版本可能冲突的问题。

虚拟环境的具体用法请参见虚拟环境和包。

使用IDE

使用集成开发环境 (IDE) 提供了便于应用程序开发的大量功能,可以显著提高开发效率,以下是常见于Python开发中使用的IDE:

- Eclipse
- · PyCharm (商业软件)

7.2 使用Flask开发应用

Flask是Python的一个轻量级Web应用框架。本文档将演示如何使用Flask创建一个应用,并将 其部署到Web+上。

步骤一: 创建应用

1. 按照以下目录结构创建一个名为webplusdemo的目录,并在该目录下创建一个名为

application.py的文件。

webplusdemo _____ application.py

2. 在application.py文件中输入以下内容。

```
from flask import Flask
app = Flask(__name__)
@app.route('/')
def hello_world():
    return 'Hello World!'
```

!) 注意:

- Web+会自动识别名为application.py的文件和全局变量app,若您需要使用其他文件
 名,需要通过Procfile或启动命令来指定需要的启动命令,具体操作请参见使用Procfile配置
 应用进程和#unique_32。
- · Web+默认会使用gunicorn作为Flask应用的服务器。

步骤二:打包应用

进入webplusdemo目录,执行以下命令完成应用打包,生成部署包文件webplusdemo.zip。

```
zip -r webplusdemo.zip ./
```

步骤三: 部署应用至Web+

- 1. 登录 Web+控制台,并在页面左上角选择所需地域。
- 2. 在概览页最近更新的部署环境区域的右上角单击新建。

3. 在应用基本信息页面选择技术栈类型为Python,设置应用基本信息,设置完成后单击下一步。

技术栈类型。	Tomcat Java 运行在Tomcat容器中的Java应用, 黃山 支持WAR和ZIP类型的部署程序包。 董通Java应用,支持FatJAR和ZIP类型的部署程序包。	P类型的部
	Go PHP 普通PHP应用,支持ZIP类型的部署 Python 编译为可执行文件的Go应用,支持 文持 音通PHP应用,支持ZIP类型的部署 管通Python应用,支持ZIP 百运行时环境。 算程序包。 普通Python应用,支持ZIP 普通Python应用,支持ZIP	ν类型的部
	Native 原生应用,支持ZIP类型的部署程序 包。	
应用名称 *	doc-test	8/64
应用描述	请输入一段描述信息帮助您识别这个应用,长度不超过1024个字符。	0/1024
下一步		

4. 在部署环境信息页面设置部署环境名称,部署包来源选择上传本地程序,上传您刚打包的 webplusdemo.zip,设置部署包版本后单击完成创建。

技术栈版本 🚺 *	★ Python 3.7.4 / Aliyun Linux 2.1903(最新版本)
部署环境名称 *	请输入您想要创建的部署环境名称,支持大小写字母、数字、"_"和"-",长度不超过64个字符。
部署环境描述	请输入一段描述信息帮助您识别这个部署环境,长度不超过1024个字符。
部署包来源*	 上传本地程序 使用样例程序
上传文件 *	选择文件
部署包版本 *	20190819.174715
版本描述	请输入一段描述信息帮助您识别这个版本,长度不超过1024个字符。
上一步下一步	完成创建

在完成创建页面单击查看该应用或完成创建可进入应用详情页面。单击部署环境名称进入部署环境
 境详情页面,然后单击公网访问地址右侧的链接进入应用首页。

$\leftarrow \rightarrow$	G	① 不安全 'si 'si'si 'si'si 'si 'si 'si 'si 'si 'si 'si 'si 'si'
Hello Wo	orld!	

更多信息

- · 在控制台部署应用的详细配置步骤请参见#unique_9。
- ·使用CLI完成应用创建和部署的操作请参见#unique_10。
- ・想了解更多Flask信息,请进入Flask官方网站查看。

7.3 使用Django开发应用

Django是Python的一个开放源代码的Web应用框架。本文档将演示如何使用Django创建一个应用和给应用添加MySQL数据库,并将其部署到Web+上。

步骤一:安装Django

执行以下命令安装Django。由于在本文档中将使用MySQL,因此需要安装pymysql模块。

pip install Django pymysql

(!) 注意:

使用Django(2.2版本以上)需要Python 3.5以上版本,在本文档将使用3.7.4版本作为示例。

步骤二: 创建应用

1. 执行以下命令执行django-admin命令来快速创建一个项目。

django-admin startproject webplusdemo

2. 可以看到创建的目录结构如下。

L___ wsgi.py

3. 执行以下命令执行django-admin命令来快速创建一个项目。

django-admin startproject webplusdemo

步骤三:打包应用

1. 执行以下命令来修改settings.py中ALLOWED_HOSTS配置项,允许所有域名的访问。

```
ALLOWED_HOSTS = ['*']
```

2. 执行以下命令改写settings.py中的数据库配置。Django默认使用sqlite数据库,本示例中 将使用RDS中的MySQL数据库。

```
# Database
# https://docs.djangoproject.com/en/2.2/ref/settings/#databases
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': os.environ['WP_RDS_DATABASE'],
        'USER': os.environ['WP_RDS_ACCOUNT_NAME'],
        'USER': os.environ['WP_RDS_ACCOUNT_PASSWORD'],
        'PASSWORD': os.environ['WP_RDS_CONNECTION_ADDRESS'],
        'PORT': os.environ['WP_RDS_PORT'],
    }
}
```

3. 进入webplusdemo目录,执行以下命令完成应用打包,生成部署包文件webplusdemo.zip。

```
zip -r webplusdemo.zip ./
```

步骤四: 部署应用至Web+

- 1. 登录 Web+控制台,并在页面左上角选择所需地域。
- 2. 在概览页最近更新的部署环境区域的右上角单击新建。

3. 在应用基本信息页面选择技术栈类型为Python,设置应用基本信息,设置完成后单击下一步。

技术栈类型。	Tomcat 运行在Tomcat容器中的Java应用, 支持WAR和ZIP类型的部署程序包。	Java 普通Java应用,支持FatJAR和ZIP类 型的部署程序包。	Node.js 普通Node.js应用,支持ZIP类型的部 署程序包。
	Go 编译为可执行文件的Go应用,支持 ZIP类型的部署程序包。安装有Go语 言运行时环境。	PHP 普通PHP应用,支持ZIP类型的部署 程序包。	Python 普通Python应用,支持ZIP类型的部 署程序包。
	Native 原生应用,支持ZIP类型的部署程序 包。		
应用名称 *	doc-test		8/64
应用描述	请输入一段描述信息帮助您识别这个应用,长度不超过	1024个字符。	0/1024
下一步			

4. 在部署环境信息页面设置部署环境名称,部署包来源选择上传本地程序,上传您刚打包的 webplusdemo.zip,设置部署包版本后单击下一步。

技术栈版本 🚺 *	★ Python 3.7.4 / Aliyun Linux 2.1903(最新版本)
部署环境名称 *	请输入您想要创建的部署环境名称,支持大小写字母、数字、"_"和"-",长度不超过64个字符。
部署环境描述	请输入一段描述信息帮助您识别这个部署环境,长度不超过1024个字符。
部署包来源*	● 上传本地程序● 使用样例程序
上传文件 *	选择文件
部署包版本 *	20190819.174715
版本描述	请输入一段描述信息帮助您识别这个版本,长度不超过1024个字符。
上一步下一步	完成创建

5. 在配置页面选择预设配置为自定义模式。

→ 应用基本信息 -	
应用名称	webplusdemo
部署环境名称	webplusdemo
* 预设配置	○ 低成本 低成本配置仅包含一台在当前可用区中可购买的小规格的ECS实例。
	○ 高可用 高可用配置包含在当前可用区中可购买的两台小规格的ECS实例和一台性能共享型的SLB实例。
	● 自定义 j5配置将允许您按照需求自定义部署环境中的资源和软件。
	平台

6. 展开云数据库RDS,按图所示配置云数据库类型为MySQL,并选择数据库版本、系列和类型等数据库基本信息。

~ 云数据库RDS NEW	~ ~
开启云数据库RDS 阿里云关系型数据库(Relational Database Service,简称RDS)是一种稳定可靠、可弹性伸缩的在线数据库服务,支持MySQL、SQL Server、PostgreSQL、PPAS和 MariaDB TX引擎,并且提供了容灾、备份、恢复、监控、迁移等方面的全套解决方案。详情可参考什么是云数据库RDS。	
实例来源	1
"代购"实例为Web+帮你购买及维护的实例,"导入"实例为您自己购买并维护的实例	(
数据库类型	
RDS支持MySQL、PostgreSQL、SQL Server、MariaDB TX和PPAS(Postgre Plus Advanced Server,高度兼容Oracle数据库)等数据库类型。	
MySQL PostgreSQL SQLServer MariaDB PPAS	
数据库版本	
请根据您的业务需要选择适合的数据库引擎版本	
8.0 5.7 5.6 5.5	
系列	1
云数据库RDS的实例包括四个系列:基础版、高可用版、集群版和金融版。具体详情可参考 <mark>产品系列概述</mark> 。	4
高可用版 基础版	
	-
17-104天坐 为满足不同场景的需求,云数据库RDS提供三种数据存储类型:本地SSD盘、SSD云盘和ESSD云盘。推荐使用本地SSD盘将数据存储于本地SSD盘,可以降低I/O延时。具体	
可参考存储类型。	1
本地SSD盘(推荐) SSD云盘 ESSD云盘	1

7. 展开生命周期挂钩,在PostPrepareApp编辑框内输入以下内容。配置完成后单击完成创建。

source /etc/bashrc && cd \$APP_HOME && python manage.py migrate

在完成创建页面单击查看该应用或完成创建可进入应用详情页面。单击部署环境名称进入部署环境
 境详情页面,然后单击公网访问地址右侧的链接进入应用首页。

1.0				
	C ① 不安全	121.76.81.219		
		django		
			A A	
		The	install worked successfully! Congr	ratulations!
			You are seeing this page because DEBUG=Tr your settings file and you have not configure	ue is in ed any
			UILLS.	
		Django Documentation Topics, references, & how-to's	Cet started with Django	Conr

上面步骤配置了数据库,因此可以访问登录页。您可以通过在数据库写入用户表或执行python

manage.py createsupersuer的方式来创建用户。

③ 不安全 1 / /		0-7	l≊ ☆
	Django administration		
	Please enter the correct username and password for a staff account. Note that both fields may be case-sensitive.		
	Username:		
	admin		
	Password:		
	Log in		

更多信息

- · 在控制台部署应用的详细配置步骤请参见<mark>部署应用</mark>。
- ·使用CLI完成应用创建和部署的操作请参见在CLI快速部署应用。
- ·想了解更多Django信息,请进入Django官方网站或Django Github项目查看。

8 ASP.NET Core

8.1 设置ASP.NET Core开发环境

在本地开发环境测试ASP.NET Core应用,需要准备相关的开发环境。本文将介绍ASP.NET Core开发环境的设置步骤,并提供相关工具的安装页面链接。

安装Visual Studio

对于开发ASP.NET Core应用, Visual Studio系列工具提供了很多便捷的功能, 推荐使用macOS或者Windows的用户在Visual Studio官网下载并安装Visual Studio。

安装SDK

使用Linux的开发者需要安装.NET Core SDK,以在CentOS 7系统中安装.NET Core 2.2 SDK 为例。

执行以下命令来安装SDK。 或参考Install .NET Core 2.2 SDK on Linux CentOS 7 - x64文档来完成安装。

```
sudo rpm -Uvh https://packages.microsoft.com/config/centos/7/packages-
microsoft-prod.rpm
sudo yum update
sudo yum install dotnet-sdk-2.2
```

8.2 部署ASP.NET Core应用至Web+

Razor 页面是 ASP.NET Core MVC 的一个新功能,它可以使基于页面的编码方式更简单高效。 本文将以ASP.NET Core Razor页面Web应用为例,介绍如何创建一个ASP.NET Core 应用并将 其部署到Web+。

前提条件

设置ASP.NET Core开发环境

	新建坝日		
为新项目选择一个模板			
 ③ 最近使用 ③ 多平台 库 Э.NET Core 应用 库 测试 ⑨ 云 常规 ● 其他 	 常規 シ 控制台应用程序 ASP.NET Core シ 空 API シ Web 应用程序 ● Web 应用程序(模型视图控制器) ● Razor 类库 ● Angular 	C#	していた し
.NET 杂项 取消	・ 「 React.js React.js和Redux		上一步

2. 在配置新的Web应用程序页面填写项目名称为webplusdemo, 然后单击创建。

	新建项目			
配置新的 🛚	/eb 应用程序			
项目名称: 解决方案名称: 位置: 版本控制:	webplusdemo /Users/zhouwenb/Projects ✓ 在解决方案目录内创建项目目录。 ● 将 git 用于版本控制。 ✓ 创建 .gitignore 文件以忽略不重要的文件。	浏览	预览 /Users/zhouwenb/Projects webplusdemo webplusdemo webplusdemo webplusdemo.csproj	
取消			上一步 创建	*

- 3. Web+可以在在反向代理或SLB中配置HTTPS,在此处您可以将应用自身的HTTPS URL去
 - 掉,请按以下示例修改Properties/launchSettings.json文件中的applicationURL配
 - 置,仅保留HTTP的访问链接。

Oebug > Default		Visual Stu	udio Community	2019 for Mac		Q~ 按"跳."进行搜索		
■ 解決方案 □ × < >	launchSettings.jse	in o						T
▼ ■ webplusdemo Sche	ma: http://json.sche	mastore.org/launchset	tings					日月前
▶ 圖 依赖项	1 {							-
Pages	2 "iisSet 3 "wind	ings": { wsAuthentication"	: false,					洞口
Properties D launchSettings,ison	4 "anon 5 "iisE	/mousAuthenticatio (press": {	n": true,					-
www.root	6 "ap 7 "ss	licationUrl": " <u>ht</u> Port": 44373	tp://localho	<u>st:31394</u> ",				反大性
appsettings.json appsettings Development icon	8 } 9 }.							当大纲
 Program.cs 	10 "profile	es": {						4
(i) Startup.cs	12 "coi 13 "la	mandName": "IISEx	press",					「事」
	14 "en	/ironmentVariables	, ": {	1				近照
	16 }	SPNETCORE_ENVIRON	MENT: Deve	copment				
	17 18 "webp	lusdemo": {						
	19 "coi 20 "la	<pre>imandName": "Proje inchBrowser": true</pre>	ct",					
	21 "en 22 "	<pre>/ironmentVariables \SPNETCORE_ENVIRON</pre>	": { MENT": "Deve	lopment"				
	23 }, 24 "api	licationUrl": "ht	tp://localho	st:5000"				
	25 }							
	27 }							
		❸ 错误 ✓ 任务	■ 包控制台	MET Core 证书管理器	📟 应用程序输出 - dotnetra	zor 💴 应用程序输出 - webplusdemo	正在发布到文件:	ŧ

4. 打开浏览器输入http://localhost:5000/来访问项目。

① localhost:5000			x 📀 😘
Use this space to summarize your privacy and cooki	ie use policy.		Learn More Accept
Microsoft A	Desoft's Azure cloud platform all scale web apps.	lows you to build, deploy, an	d
Application uses • Sample pages using ASP.NET Core Razor Pages • Theming using Bootstrap • Use log • Add pa • Target produc	O O O O O O O O O O O O O O O O O O O	rview R nceptual overview of what is P.NET Core ndamentals of ASP.NET Core ch as Startup and middleware. wrking with Data curity ent side development velop on different platforms ad more on the documentation e	 Run & Deploy Run your app Run tools such as EF migrations and more Publish to Microsoft Azure App Service

步骤二:打包应用

1. 在Visual Studio顶部菜单栏选择生成 > 发布到文件夹,并指定一个目标路径。

Visual Studio Community 文件 编辑	查看 搜索 耳	页目 <mark>生成</mark> 运行 版本 1	空制 工具 窗口 帮助
Oebug > Default	II A Ŧ	全部生成 全部重新生成	жв ^жв
■解决方案 □×	< > launch	_{Settin} 全部清理	
▼ 📄 webplusdemo		生成 webplusdem	o ೫K
🔻 🔲 webplusdemo	Schema: http:/	^{//json.} 重新生成 webplus	demo ^೫K
▶ 💿 依赖项	1 {	清理 webplusdem	o 쇼 윎K
▶ 📄 Pages	2	"ii:	<u> </u>
▼ ■ Properties	4	"; 行止	₩₩
launchSettings.json	5	"发布到文件夹	
▶ 📄 wwwroot	6	发布到 Azure	<u>ost:31394</u> ",
appsettings.json	8	}	
appsettings.Development.json	9	},	
Program.cs	10	"profiles": {	
	11	"IIS Express": {	C.F
In Statup.cs	12	"commandName": "11	SExpress",
	13	"launchBrowser": t	rue,
	14	"environmentVariab	les": {

2. 通过终端访问部署包的目标路径,可以看到已经成功生成了文件。

-rw-rr	1	-	staff	146	9	20	21:05	appsettings.Development.json
-rw-rr	1		staff	105	9	20	21:05	appsettings.json
-rw-rr	1		staff	460	9	20	21:21	web.config
-rw-rr	1		staff	79360	9	20	21:21	webplusdemo.Views.dll
-rw-rr	1		staff	5504	9	20	21:21	webplusdemo.Views.pdb
-rw-rr	1		staff	223260	9	20	21:21	webplusdemo.deps.json
-rw-rr	1		staff	9216	9	20	21:21	webplusdemo.dll
-rw-rr	1		staff	2064	9	20	21:21	webplusdemo.pdb
-rw-rr	1		staff	213	9	20	21:21	webplusdemo.runtimeconfig.json
drwxr-xr-x	7		staff	224	9	20	21:21	www.root
(python3.6)	+	newProj						

3. 执行以下命令完成打包,在当前目录下得到部署包文件webplusdemo.zip。

```
zip -r webplusdemo.zip ./
```

- 步骤三:将应用部署至Web+
 - 1. 在应用基本信息页面选择技术栈类型为ASP.NET Core,设置应用基本信息,设置完成后单击下

技术栈类型。	Tomcat 运行在Tomcat容器中的Java应用, 支持WAR和ZIP类型的部署程序包。	Java 普通Java应用,支持FatJAR和ZIP类 型的部署程序包。	Node.js 普通Node.js应用,支持ZIP类型的部 署程序包。
	Go 编译为可执行文件的Go应用,支持 ZP类型的部署程序包。安装有Go语 音运行时环境。	PHP 普通PHP应用,支持乙P类型的部署 程序包。	Python 普通Python应用,支持ZIP类型的部 署程序包。
	ASP.NET Core ASP.NET Core应用,支持Razor和 MVC类型的Web应用。	Ruby Ruby应用,支持ZIP类型的邮署程序 包,支持Ruby on Rails应用。	Native 原生应用,支持ZIP类型的部署程序 包。
应用名称 *	doc-test		8/6
应用描述	请输入一段描述信息帮助您识别这个应用,长度不超过10	024个字符。	0/102
一步			

在完成创建页面单击查看该应用或完成创建可进入应用详情页面。单击部署环境名称进入部署环境
 境详情页面,然后单击公网访问地址右侧的链接进入应用首页。

① 不安全	o1i7qN99			\$	0 ©	• •
Use this space to summarize your pr	ivacy and cookie use policy.			Learn More	Accept	
ASP.NE	T Core Wind	dows Linux that can run anywhere. Learn Ma ° °	OSX	>		
Application uses	How to	Overview	Run & D	Deploy		
 Sample pages using ASP.NET Core Razor Pages Theming using Bootstrap 	 Working with Razor Pages. Manage User Secrets using Secret Manager. Use logging to log a message. Add packages using NuGet. Target development, staging or production environment. 	 Conceptual overview of what is ASPNET Core Fundamentals of ASP.NET Core such as Startup and middleware. Working with Data Security Client side development Develop on different platforms Read more on the documentation site 	 Run your a Run tools and more Publish to Service 	app such as EF migra Microsoft Azure /	tions App	
© 2019 - webplusdemo						

更多信息

- · 在控制台部署应用的详细配置步骤请参见#unique_9。
- ·使用CLI完成应用创建和部署的操作请参见#unique_10。
- · .NET Core相关操作指南,请参见.NET Core 指南。

9 Ruby

9.1 配置Ruby开发环境

在本地开发环境测试Ruby应用,需要准备相关的开发环境。本文将介绍Ruby开发环境的设置步骤,并提供相关工具的安装页面链接。

安装 Ruby

您可以参考Ruby官网文档来安装Ruby,建议您安装Web+支持的版本Ruby 2.6.3。

Linux

Linux下使用系统包管理是安装Ruby最简单的方法,以Centos 7为例,执行以下命令即可完成安装:

yum install ruby

若您希望安装最新版本或指定版本的Ruby,则需要下载对应的源代码版本进行编译安装,下载解 压后执行以下命令:

```
./configure
make && make install
```

macOS

执行以下命令,使用brew来快速安装Ruby。

brew update && brew install ruby

Windows

如果您使用Windows系统,可以借助RubyInstaller来完成Ruby的安装。

配置GEM

Gem为Ruby的包管理工具,中国内地网络访问默认Gem源的速度偏慢,可以通过使用镜像站点来 加速访问,安装好Ruby后请执行以下命令来配置GEM。

```
gem sources --add https://gems.ruby-china.com/ --remove https://
rubygems.org/
```

bundle config mirror.https://rubygems.org https://gems.ruby-china.com

安装IDE

集成开发环境 (IDE) 提供了便于应用程序开发的大量功能,可以显著提高开发效

率。RubyMine是常见在Ruby开发中使用的IDE,请访问链接下载软件安装包并安装:*RubyMine* (商业软件)。

9.2 部署Ruby on Rails应用至Web+

Ruby on Rails(后文简称Rails)是一个使用Ruby语言开发的开源Web应用框架,本文将介绍 如何创建一个简单的Rails项目并将其部署到Web+。

步骤一:安装Rails

执行以下命令安装Rails。

gem install rails

步骤二: 创建Rails应用

1. 执行以下命令使用rails命令行工具来创建一个空应用。

rails new webplusdemo

执行完后将生成名为webplusdemo的工程目录。

2. 进入webplusdemo的工程目录,执行以下命令来启动服务。

rails server

3. 打开浏览器, 输入http://localhost:3000来访问应用首页。



步骤三:打包应用

切换到项目路径下执行以下命令完成打包。

```
zip -r webplusdemo.zip ./
```

步骤四: 部署Ruby应用至Web+

1. 在应用基本信息页面选择技术栈类型为Ruby,设置应用基本信息,设置完成后单击下一步。

技术栈类型。	Tomcat 运行在Tomcat容器中的Java应用, 支持WAR和ZIP类型的部署程序包。	Java 普通Java应用,支持FatJAR和ZIP类 型的部署程序包。	Node.js 普通Node.js应用,支持ZIP类型的部 署程序包。
	Go 编译为可执行文件的Go应用,支持 ZIP类型的配署程序包。安装有Go语 音运行时环境。	PHP 普通PHP应用,支持ZIP类型的部署 程序包。	Python 普通Python应用,支持ZIP类型的部 署程序包。
	ASP.NET Core ASP.NET Core应用,支持Razor和 MVC类型的Web应用。	Ruby Ruby应用,支持ZIP类型的部署程序 包,支持Ruby on Rails应用。	Native 原生应用,支持ZIP类型的部署程序 包。
应用名称 *	请输入您想要创建的应用名称,支持大小写字母、数字	"_"和"-",长度不超过64个字符。	0/64
应用描述	请输入—段描述信息帮助您识别这个应用,长度不超过	1024个字符。	
			0/1024
下一步			

在完成创建页面单击查看该应用或完成创建可进入应用详情页面。单击部署环境名称进入部署环境
 境详情页面,然后单击公网访问地址右侧的链接进入应用首页。



更多信息

- ・在控制台部署应用的详细配置步骤请参见#unique_9。
- ・使用CLI完成应用创建和部署的操作请参见#unique_10。
- ・若想详细了解 Ruby on Rails,请访问*Rails*官网。

10 Native

10.1 部署原生应用到Web+

Web+提供了丰富的多语言应用的支持,如果您无法找到适合的技术栈类型,您可以选择原生应用 来完成部署。本文将介绍原生应用部署到Web+的设置步骤。

原生应用简介说明

- ·Web+不会自动安装基础软件,您可通过#unique_32来自定义安装步骤,比如 在PostPrepareEnv挂钩中使用命令安装所需软件或依赖。
- ・Web+不提供默认的启动命令,因此必须通过配置#unique_32/

unique_32_Connect_42_section_pks_hk7_jtq或参考文档使用Procfile配置应用进程来指定启动命 令。

- Web+默认原生应用的服务端口为8080,若您的应用没有启动在此端口,请将服务端口配置到 实际启动的端口,您可使用环境变量\$WP_SERVICE_PORT来配置应用的服务端口。如果应用的 服务端口跟启动端口不一致可能导致应用健康检查失败。
- ・您的应用必须是在对应的部署环境中(目前操作系统只支持AliyunLinux2.1903)下可以正 常运行的。
- ·原生应用也可以使用数据库,配置方法请参考#unique_20。您可以从环境变量读取到Web+设置的数据库选项,选择合适的驱动程序即可实现数据库访问,环境变量请参考#unique_42。

打包原生应用

本文以一个简单的HTTP服务程序simpleserver作为示例,该程序只包含一个可执行文件,启动 后将监听在8080端口,并接收GET请求,输出OK的响应。

— simpleserver

1. 在应用项目目录下创建Procfile, 文件中写入以下内容:

web: ./simpleserver

2. 执行以下命令打包应用,即可生成可在Web+使用的部署包。

zip -r simpleserver.zip ./

将原生应用部署至Web+

1. 登录 Web+控制台,并在页面左上角选择所需地域。

- 2. 在概览页最近更新的部署环境区域的右上角单击新建。
- 3. 在应用基本信息页面选择技术栈类型为Native,设置应用基本信息,设置完成后单击下一步。

技术栈类型。	Tomcat 运行在Tomcat容器中的Java应用, 支持WAR和ZIP类型的邮署程序包。	Java 普通Java应用,支持FatJAR和ZIP类型的部署程序包。	Node.js 普通Node.js应用,支持ZIP类型的部 署程序包。			
	Go 编译为可执行文件的Go应用,支持 ZIP类型的部署程序包。安装有Go语 吉运行时环境。	PHP 普通PHP应用,支持口P类型的部署 程序句。	Python 普通Python应用,支持ZIP类型的部 著程序包。			
	ASP.NET Core ASP.NET Core应用,支持Razor和 MVC类型的Web应用。	Ruby Ruby应用,支持ZIP类型的部署程序 包,支持Ruby on Rails应用。	Native 原生应用,支持ZIP类型的邮署程序 包。			
应用名称*	请输入您想要创建的应用名称,支持大小写字母、数字、"_"和"",长度不超过64个字符。 () 请输入一段描述信息帮助您识别这个应用,长度不超过1024个字符。					
于一步			0/1024			

在部署环境信息页面设置部署环境名称,部署包来源选择上传本地程序,上传您刚打包的部署
 包,设置部署包版本后单击完成创建。

访问应用

在完成创建页面单击查看该应用或完成创建可进入应用详情页面。单击部署环境名称进入部署环境 详情页面,然后单击公网访问地址右侧的链接进入应用首页。



更多信息

- · 在Web+控制台快速部署应用的视频演示请参见#unique_8。
- · 在控制台部署应用的详细配置步骤请参见#unique_9。
- ·使用CLI完成应用创建和部署的操作请参见#unique_10。
- ·完成应用托管之后的应用的管理操作请参见#unique_11。
- ·管理应用所在的部署环境的操作请参见#unique_12。