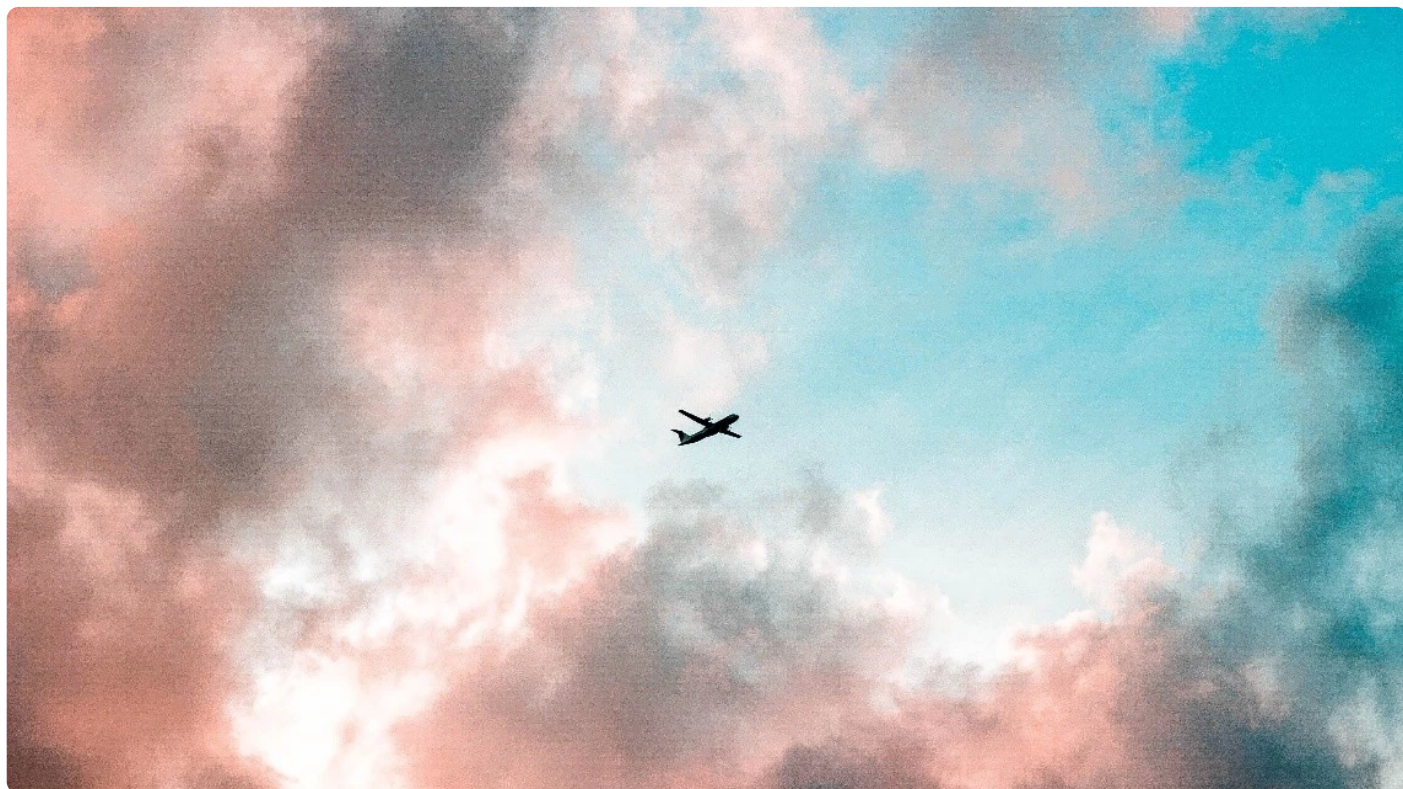


## 11 | 应用托管服务：Web应用怎样在云上安家？

2020-03-27 何恺铎 来自北京

《深入浅出云计算》



你好，我是何恺铎。今天我们来谈谈云上的应用托管服务。

从互联网诞生开始，网站就一直是人接触内容的主要窗口，是互联网应用的主要形态。所以许多的编程语言和技术栈在争夺开发者的时候，都是以支持网站应用的开发作为主要的发力点。

这个浪潮催生了各类动态网站编程技术，和各种 Web 后端框架的兴起。而随着 AJAX 应用和移动互联网的到来，Web 已经不只是网站了，它还包括各种各样的 API。我们可以把它们统称为 Web 应用。

Web 应用，显然是一个极为普遍的需求，也是一个巨大的市场。所以，作为承载一切的云计算，当然需要为 Web 应用的运行提供最好的场所、能力和辅助。

不过，你当然也可以使用虚拟机和其他 IaaS 组件来搭建你的网站。但用 IaaS，你需要操心的事情比较多，包括虚拟机的创建、运行环境的搭建和依赖安装、高可用性和水平扩展的架构等

等。而且一旦应用的规模大了，每次应用的更新升级也会是件麻烦事，另外你还要操心 Web 漏洞的弥补修复。

**那么，能不能有一个平台服务，来帮助我们解决所有这些基础架构问题，让我们只需要专注于应用构建本身就好了呢？**当然是有的，这就是云上应用托管 PaaS 服务的由来。

## 什么是应用托管服务？

和每一项云服务一样，应用托管类服务也是从简单到复杂、从功能单一到丰富强大这样一路走来的。

在云计算发展的早期，就已经出现了“**建站类服务**”，这正是应用托管服务的雏形。当时的建站类服务，会自动为你分配好服务器，安装好相应语言的 Web 环境以供你使用。在部署层面，服务通常会开放 FTP 端口，以便你上传服务器端的代码、脚本和资源。这是应用服务的一种轻量形式。

注意：目前仍然有云厂商提供了这种轻量 Web 服务的产品形态，比如阿里云的轻量应用服务器、百度云虚拟主机服务等。这些服务仍然可用，而且由于使用简单且成本低廉，在合适的场景下（比如中小企业建站）也是不错的选择。

而更现代的应用托管服务，已经今非昔比，不但在细节选项、自动化程度上进步了许多，还包含了大量的增值服务。如果你的运用得当，它绝对是一个利器，能为你节省许多的时间精力。

这类现代应用托管服务现在是各个云上的标配，AWS 上对应的云服务为 Elastic Beanstalk，阿里云对应的服务为 Web 应用托管服务（Web+），Azure 上称之为 Azure 应用服务（Azure App Service）。

接下来，我就以国际版的 Azure 应用服务为例，把我们在 [第 7 讲](#)中使用过的计算**斐波那契数列**的 Web 应用，移植到 Azure 云的 PaaS 服务上来。

首先，我们来创建一个应用服务的实例：

## 项目详细信息

选择一个订阅来管理部署的资源和本地。使用文件夹等资源组组织和管理你的所有资源。

订阅 \* ⓘ

资源组 \* ⓘ

geektime-hellocloud

新建

## 实例详细信息

名称 \*

fibonodejs

.azurewebsites.net

发布 \*

代码 Docker 容器

运行时堆栈 \*

Node 10 LTS

操作系统 \*

Linux Windows

区域 \*

Japan East

找不到应用服务计划? 请尝试其他区域。

## 应用服务计划

应用服务计划定价层确定与你的应用相关的位置、功能、成本和计算资源。 [了解详细信息](#)

Linux 计划 (Japan East) \* ⓘ

(新项) Plan-geektimehellocloud

新建

SKU 和大小 \*

标准 S1  
100 总 ACU, 1.75 GB 内存  
[更改大小](#)

在上图中，我填写了一些重要信息，比如把这个实例称为 **fibonodejs**，系统会自动给它一个免费的域名 **fibonodejs.azurewebsites.net**。另外，我还选取了 Node 技术栈，以及 Linux 操作系统。在运行配置方面，我选取了**标准 S1**，对应一个 vCPU 和 1.75GB 内存的计算资源。

**这里你需要注意**，除了 Node 技术栈之外，一般云厂商都会提供 Java、PHP 等多个常用语言和框架的支持。你可以事先确认一下，你偏爱的语言是否在云厂商的支持列表中。下面我还列出了 Azure 应用服务支持的部分环境，可以看到，同一个语言或框架还支持很多不同的版本，所以我们的选择是很丰富的。

<b>.NET Core</b>	Node 10.1
.NET Core 3.1 (LTS)	Node 10.0
.NET Core 3.0 (Current)	<b>PHP</b>
.NET Core 2.1 (LTS)	PHP 7.3
<b>ASP.NET</b>	PHP 7.2
ASP.NET V4.7	<b>Python</b>
ASP.NET V3.5	Python 3.8
<b>Java 11</b>	Python 3.7
Java SE	Python 3.6
Tomcat 8.5	<b>Ruby</b>
Tomcat 9.0	Ruby 2.6
<b>Java 8</b>	Ruby 2.5
Java SE	Ruby 2.4
WildFly 14 (Preview)	Ruby 2.3

实例创建完毕后，就相当于我们已经建好了房子，接下来的关键就是要**邀请应用入住**。


我们在第 7 讲中使用过的 app.js 中的源码，在这里基本不需要修改就可以直接使用了，唯一需要适配一下的是最后监听的**端口号**，我们需要从固定值 80 修改为**动态值 process.env.PORT**。这个动态值会在应用运行时，从 PaaS 服务的环境变量中得到。

 复制代码

```
1 app.listen(process.env.PORT);
```


随后，我们只需要把应用的代码打包上传就可以了。我们这个 Node 应用比较简单，只是包含一个 app.js 和 package.json 配置文件。**package.json 配置文件**的内容也很简洁，只有少

数包依赖，以及一个初始化的启动命令：

 复制代码


```
1 {
2   "name": "fibo-app",
3   "version": "0.0.1",
4   "private": true,
5   "scripts": {
6     "start": "node ./app.js"
7   },
8   "dependencies": {
9     "express": "4.0.0",
10    "ip": "1.1.5"
11  }
12 }
```

我们再新建一个 **.deployment** 文件，其中，设置 `SCM_DO_BUILD_DURING_DEPLOYMENT` 参数为 **true**。这个设置会告诉 Azure PaaS 端，在部署时帮我们自动执行 `npm install` 来安装依赖项。

 复制代码

```
1 [config]
2 SCM_DO_BUILD_DURING_DEPLOYMENT=true
```

然后，我们把上面提到的这三个文本文件一起**打包为 zip**：

 复制代码

```
1 [client@clientVM fiboapp]$ zip fiboapp.zip app.js package.json .deployment
2 updating: app.js (deflated 48%)
3 updating: package.json (deflated 30%)
4 updating: .deployment (stored 0%)
```

接下来十分关键的一步，是我们要用 Azure CLI 中的 **webapp** 相关命令，完成 zip 文件的上传。



```
1 [client@clientVM fiboapp]$ az webapp deployment source config-zip --resource-group 复制代码
2 Getting scm site credentials for zip deployment
3 Starting zip deployment. This operation can take a while to complete ...
```

我们需要做的就只有这些。当 Azure 应用服务收到 zip 包后，就会在一个隔离环境中自动解压、安装相关依赖项，并开始运行我们的应用了。

尝试一下网站服务，已经在正常地工作了：

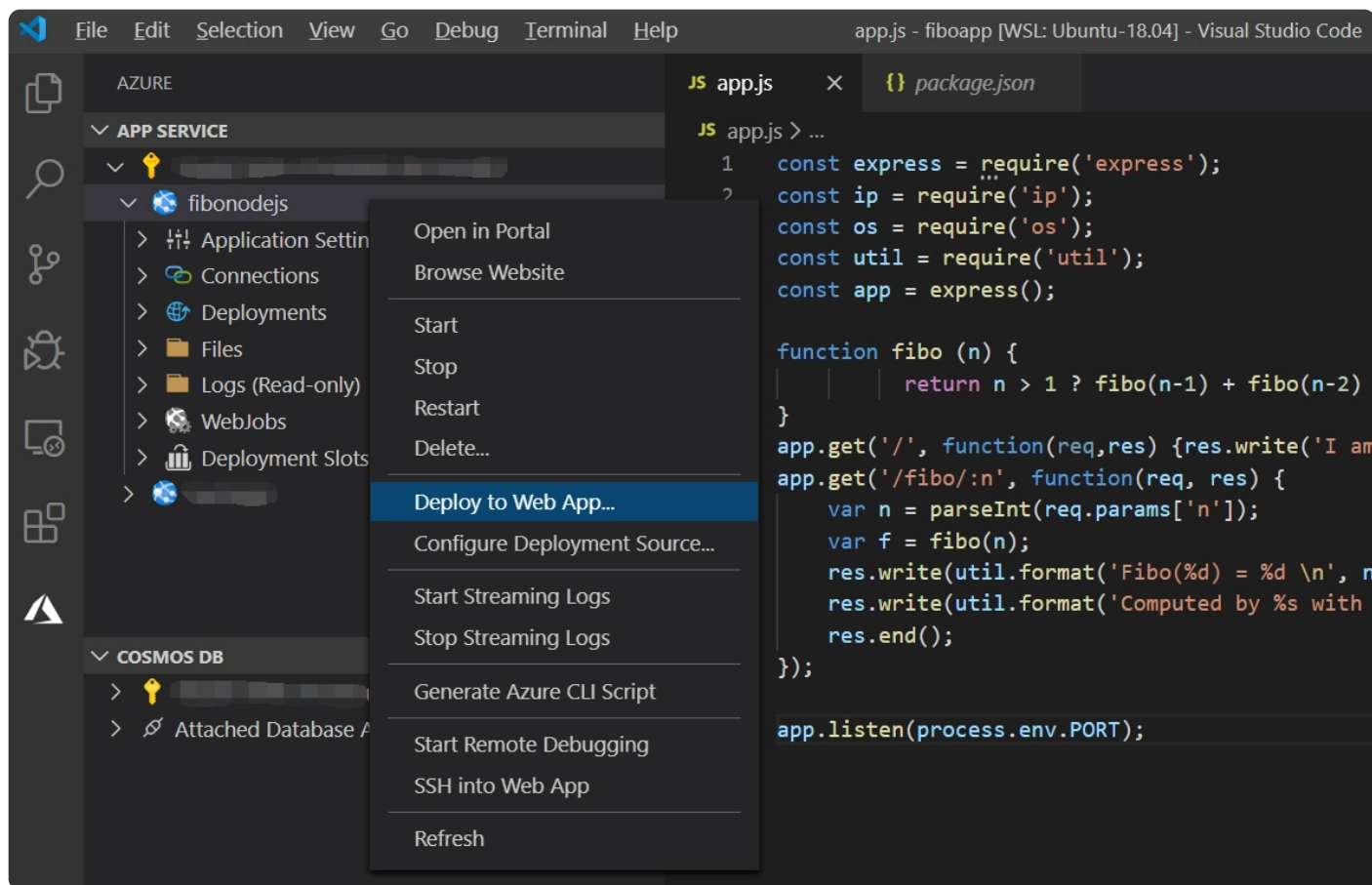
```
1 [client@clientVM ~]$ curl https://fibonodejs.azurewebsites.net/
2 I am healthy
3 [client@clientVM ~]$ curl https://fibonodejs.azurewebsites.net/fibo/35
4 Fibo(35) = 14930352
5 Computed by 49fe1eef783e with private ip 172.16.1.4 复制代码
```

通过这个例子，你能够看出，**应用服务的本质就是为你的应用提供一个隔离的独立运行环境**。作为用户来讲，你可以只专注于业务逻辑，不需要来手动创建这个环境，更不需要运维这个环境。

小提示：应用托管服务背后采用的隔离技术对用户一般是不可见的，它可能是虚拟机，可能是 Docker，或者是自研的其他容器类技术。通过查看后台日志，你可以发现，Azure App Service 在 Linux 上的后台，其实是使用了 Docker 来封装运行我们的 node.js 程序。

另外，为了便于说明操作步骤，上面的实操中，我使用了**命令行**的方式来进行应用的打包和部署。如果你觉得这个过程有点儿麻烦，也完全可以借助一些 IDE 工具来完成它。比如说，你可以使用 Visual Studio Code 配合 App Service 插件，来进行开发和部署。

而且，上面的打包上传等步骤，我们在集成环境中，只需要按下 **“Deploy to Web App”**，一键就可以完成了，如下图所示：



所以，云厂商为了提高 PaaS 服务的易用性，往往会有比较成熟的开发配套工具链可以选用，这也是应用托管服务的一大优势。

## 应用托管的增值服务

上面我通过例子，给你演示了一下应用服务的基本功能，但它的能力远不止此。成熟的应用服务还能够提供许多**增值服务**，来进一步地满足我们在实际开发运维 Web 应用时，产生的各个层面的需求。

**第一项增值服务就是监控**，尤其是针对 Web 应用的特点而进行的 HTTP 层面的应用监控。所以，你不仅能看到计算资源的占用率，如 CPU、内存使用率等，还能看到许多应用层指标，比如总请求数、错误响应数、并发连接数、响应时间等等。这些都是你在监控应用运行时非常有帮助的信息，而这一切都是 PaaS 服务自动提供、开箱即用的功能。

而且，基于这些监控的指标，你还能够在云上制定相应的报警规则，当某些指标达到你设定的阈值时，会及时发送警报。这同样是一个非常实用的功能。

第二个方面是**扩展**，也就是底层计算资源和流量需求的匹配。这里既包含了底层机器配置的**垂直扩展**，也包含了机器数量层面的**水平扩展**。一旦你有调整需求，只需要动动手指发出指令，就可以随时升级相应的机器配置，并无缝切换。

特别是水平扩展的存在，它相当于同时包含了我们第 7 讲中提到的**负载均衡和弹性伸缩**，把它们都一股脑儿集成到了托管服务中。这意味着应用托管服务不是只能对应一台机器，而是能够创建多台机器来承接请求，并会在前端均衡地分发到多个实例上去。这里你同样可以指定自动伸缩的规则，来让应用服务自动地调整实例数量。

下面，我给出了一个实际可用的 **Azure 应用服务的伸缩规则示例**，它设置了 CPU 使用率的阈值，当实际 CPU 占用超出阈值后，服务会自动地增加或减少实例数量。假如我们的 fibonodejs 服务，需要面向公众正式服务的话，就可以施加类似的配置。

默认值

Auto created scale condition

删除警告

不能删除最新的或者默认的定期规则。但是，可以禁用自动缩放，从而关闭自动缩放。

缩放模式

☒ 基于指标缩放

☐ 缩放为具体实例数

规则

横向扩展

何时

Plan-geektimehelloclo... (平均值) CpuPercentage > 70

计数增加 1 个

缩小

何时

Plan-geektimehelloclo... (平均值) CpuPercentage < 30

计数减少 1 个

+ 添加规则

实例限制

最小值

1

最大值

10

默认值

2

计划

此缩放条件在没有其他任何缩放条件匹配时执行

第三个方面是**集成**，这里是指与其他 PaaS 的集成。这是所有 PaaS 服务的优势，各个服务间可以互相帮助、联合作战，应用托管类服务也不例外。比如在监控数据方面，它可以和云监控

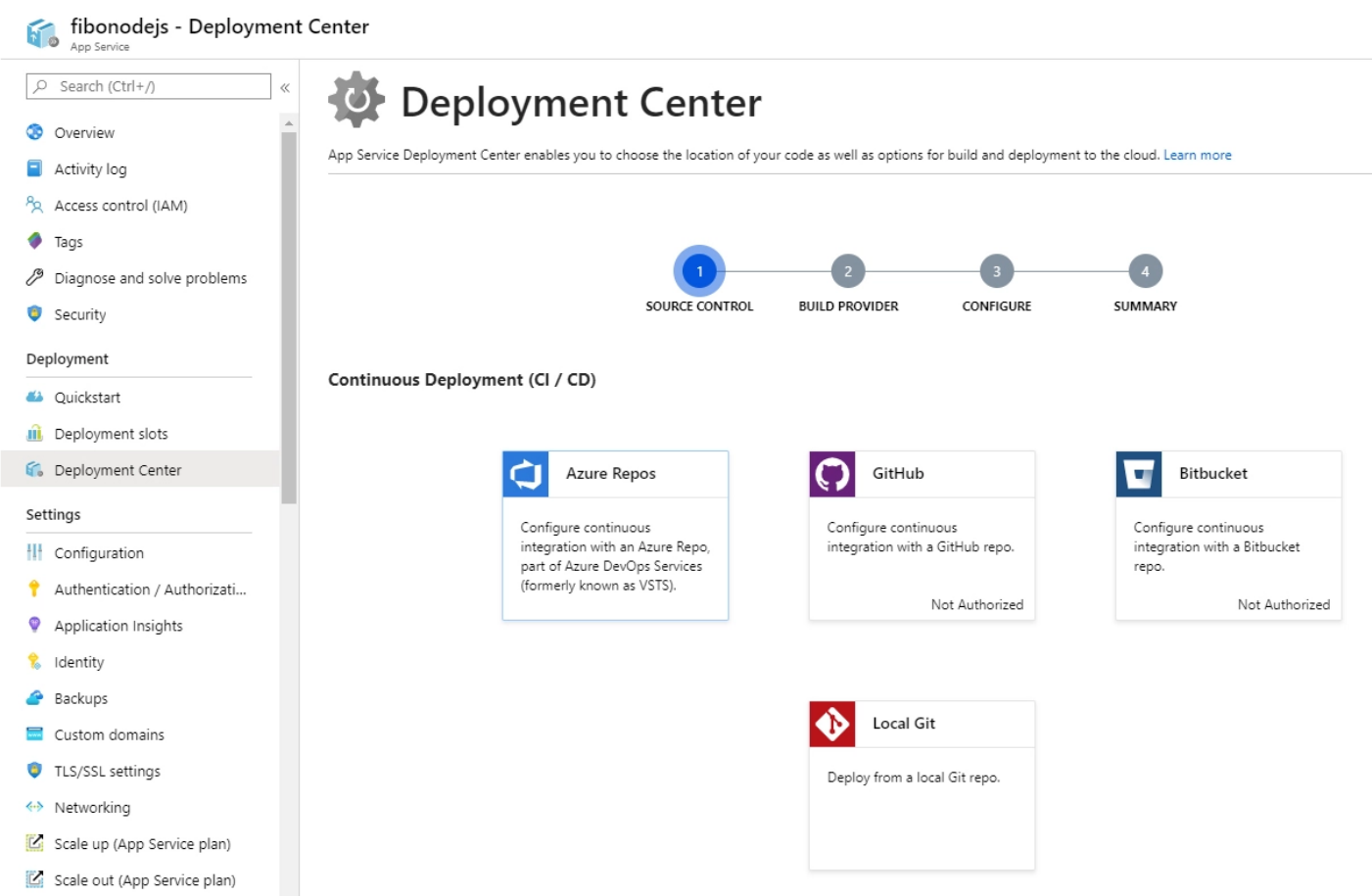


系统进行衔接；再比如，有些云允许 Web 应用以目录的形式，挂载对象存储中的文件等等。

其中，应用托管类服务还有一项非常重要的集成能力，就是**应用服务与云上 DevOps 组件和流程的无缝对接**。它意味着应用服务可以作为整个应用生命周期管理的一部分，嵌入到持续集成的流程中去。借助和源代码管理设施的联动，你的应用就可以轻松实现自动化的部署和升级。

比如说，我可以把上面 node.js 应用程序的代码，放到 Azure Repo 或者 GitHub 这样的代码仓库中，之后只要通过满足触发条件的 git push 操作，就能够自动地触发构建，并直接更新线上的应用程序。

下图展示了 Azure App Service 中相关配置的入口：



除了我在上面列举的这三方面的增值服务，应用托管服务还有许多有用的特性我没有展开讨论，比如说远程调试和诊断、运行环境自动补丁升级、私有网络内的部署、Web 防火墙防护等等。这些都是它能够给你带来的强大附加价值。

和其他 PaaS 服务一样，对于应用托管服务，**我建议你先充分地查阅文档，建立比较系统全面的认识，然后一定要通过实操，进行探索 and 实验**，这样你就会有比较深入的理解和感受，便于你作出正确的选型决策。

## 课堂总结与思考

今天，我主要给你介绍了云上的应用托管服务，这也是最受欢迎的 PaaS 服务之一。你现在应该已经知道，它是用来支撑我们 Web 应用的一个平台，它既包含了应用的基础架构和运行环境，也包括了许多增值服务，大大方便了我们 Web 应用搭建和维护过程中的各种典型诉求。

到目前为止，我所说的都是它的优点和长处。那么，你可能会问，**应用托管服务有没有弊端呢？**

我认为，这主要取决于**云服务的封装程度**，以及你**是否有环境定制的需求**。有些人的确不喜欢受限的封装环境，觉得难以做一些细节的微调和必要的 hack，也有一定道理。

好在现代 Web 托管服务，在进行高度封装的同时，也都部分地开放了内部运行环境，甚至允许你**远程登录**到相应的服务器上去，开展你所需要的改动或排查。这在一定程度上减轻了封装可能带来的“黑盒”问题，比如你可以查看到一些底层操作系统级别的错误日志。

另外你需要注意的一点，就是**价格**。不同云厂商，对于应用托管服务的收费标准都是不同的。至于它们的收费是否合理，你可以与纯虚拟机的搭建来做一个对比。特别是如果你的网站部署规模不小的话，花一些时间来做比较是非常值得的。

不过在进行比较时，你一定不能忘记，应用托管服务能够给你带来的工作效率提升，和维护负担的减轻。因为开发、部署和维护相关的人力和时间成本，都是**总体拥有成本**（Total Cost of Ownership, TCO）中的一部分。**这也是我在许多时候，会给应用托管服务投上一票的重要原因。**

**这一讲，我留给你的思考讨论题是：**

Azure 应用托管服务还有一个**部署槽**（Deployment Slot）功能，AWS 中类似的概念则称为**环境**（Environment），这同样是一个非常实用的能力。你能说说这个特性是做什么的

吗？如果你还了解其他好用的特性，也欢迎你和大家分享。

对于应用托管服务，通过代码打包上传来发布应用，一直是主流的方式，我在前面举例时也采用了这种形式。不过近来，通过**容器**来部署 Web 应用，也成为了云上应用托管服务普遍支持的形式。那你知道后者解决了前者的什么问题吗？

好了，今天我们就到这里。如果觉得有帮助，欢迎你把这篇文章分享给你的朋友。我是何恺铎，感谢你的阅读，我们下期再见。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

## 精选留言 (12)



何恺铎 置顶

2020-03-27

[上讲问题参考回答]

1. 如果要临时对外分享对象，一般对象存储都有一个生成临时链接的功能。它会通过签名生成相关token（token会附着在生成的url中），并允许你设置一个过期时间，到期以后这个链接会立刻失效。你要保存好这个私密的url，发送给你授权的用户来使用。
2. 关于大数据量上传云端，不少同学都提到了并发上传、压缩上传等最佳实践，但这些无法突破网络带宽的瓶颈。为了解决上传速度慢的问题，云厂商其实都提供了离线数据上传的服务，比如AWS Snowball、Azure Data Box、阿里云闪电立方等。这些服务通过和专用的物理存储设备配合，能实现PB级数据的迁移，俗称“寄硬盘”。



17



军舰

2020-03-29

问题二：应用托管服务将运行环境与应用进行了分离，只需要发布应用，让应用部署更轻量化；容器解决了应用部署对运行环境依赖的一致性，运行环境和应用打包在一起。应用托管服务往往解决的是一种单一运行环境的需求，对于需要定制化和多运行环境的依赖就不适合了，容器很好的解决了这个问题，运行环境和应用的适配在开发的时候已经解决了，部署后开箱即用。

作者回复：很棒，提到了容器对于运行环境的强定制能力。



13



**睡觉也在笑**

2020-03-27

老师，我是银行科技人员。因为行业的原因一般不会选用公有云的产品，您知道那些私有云产品也支持这么完备的云服务吗？我们也想交流和了解一下。

作者回复：和公有云体验最接近的，是公有云厂商推出的私有云/混合云方案，如AWS Outposts, Azure Stack, 阿里专有云等等。另外，也有一些兼容不同底层IaaS的PaaS平台解决方案，典型的如Cloud Foundry，国内也有不少厂商有自研的PaaS产品。如果真的需要选，可以通过招标详细比对：)

共 2 条评论 >



6



**八哥**

2020-03-27

Beanstalk技术上类似Google App Engine吗？paas环境最大弊端就是不支持文件写到本地磁盘，只有临时磁盘，导致应用迁移过来，要修改代码。

作者回复：是的。Web应用服务中尽量不要依赖本地磁盘，因为它的运行实例可能有多个而且会动态创建，可考虑使用一些存储类服务来替代。



3



**David Mao**

2020-03-28

老师，最近在做IaaS的技术选型，私有云部分有腾讯云Tstack, Ucloud, 华三, Dell,前两者是openstack,后两者是VMware,从技术角度，老师倾向于哪个？谢谢



1



**艾利特-G**

2020-04-10

公有云之间应用托管服务的配置应该都有区别，我想容器服务也会有区别，用惯了一家再用另一家，还得熟悉下，虽然应该也是大同小异。

容器服务下，提交的artifacts是容器，运行时没有编程语言限制，只有一个通用的容器运行时，而应用服务还有编程语言运行时的种类。早期的容器其实也没有解决跨语言应用的配置不同的问题，Docker公司解决了容器统一打包这个问题。





戴斌

2020-03-28

期待更新



Christopher

2020-03-27

阿里云貌似应用托管服务这一块功能做的没有azure和aws多啊



吴小智

2020-03-27

根绝托管服务，比较适合与前端页面的托管。如果系统比较复杂，比如说需要用到数据库、消息队列等组件，这种方式还适合嘛？如何与其他系统交互呢？

共 2 条评论 >



Harvey

2020-03-27

- 1 我猜是用来支持持续交付各阶段不同环境部署的需求
- 2 解决环境一致问题



leslie

2020-03-27

这个其实就让我想到了阿里的POLARDB和OceanBase：集群建好了，基本监控搭建好了；你只需要用就行，给你个接口去导入\导出数据，不过因此不少深层的分析就需要付费。

Docker其实最大的便利之处在于多应用的隔离：企业的各组织之间更新代码的速度不同；Docker便于彼此之间隔离使得服务器上安装的应用不那么乱且便于管理。

谢谢老师今天的分享：期待后续的课程。

作者回复：PolarDB和OceanBase恰好下一讲会谈到的。



潘政宇

2020-03-27

不错的服务



