Microsoft Azure Cloud Service

操作手册

目录

[1. 前言 5](#_Toc453664199)

[2. 读者 5](#_Toc453664200)

[3. Azure Cloud Service相关服务 5](#_Toc453664201)

[3.1 Azure Cloud Service的带宽问题 5](#_Toc453664202)

[3.2 什么是Azure Cloud Service 5](#_Toc453664203)

[3.3 Azure Cloud Service和Virtual Machine的关系 6](#_Toc453664204)

[3.4 Azure Cloud Service和其他计算服务的区别 7](#_Toc453664205)

[3.1 Azure Cloud Service支持的开发语言 9](#_Toc453664206)

[3.2 Azure Cloud Service如何解决大并发 10](#_Toc453664207)

[3.3 Azure Cloud Service单个实例大小 10](#_Toc453664208)

[3.3.1 A系列计算节点 10](#_Toc453664209)

[3.3.2 D系列计算节点 10](#_Toc453664210)

[3.4 Azure Cloud Service使用静态资源 11](#_Toc453664211)

[3.5 Azure Cloud Service成本分析 12](#_Toc453664212)

[3.6 Azure Cloud Service停止计费(非常重要) 12](#_Toc453664213)

[3.7 Azure Cloud Service限制 13](#_Toc453664214)

[4. 开始创建Azure Cloud Service 13](#_Toc453664215)

[4.1 规划好Azure订阅(非常重要) 13](#_Toc453664216)

[4.2 选择订阅(非常重要) 13](#_Toc453664217)

[4.3 Azure Emulator模拟器 14](#_Toc453664218)

[4.4 Web Role / Worker Role 14](#_Toc453664219)

[4.5 创建一个Azure Cloud Service 17](#_Toc453664220)

[4.6 查看Azure Cloud Service 19](#_Toc453664221)

[4.6.1 Cloud Service 19](#_Toc453664222)

[4.6.2 Web Role 20](#_Toc453664223)

[4.6.3 Worker Role 21](#_Toc453664224)

[4.6.4 回顾 23](#_Toc453664225)

[4.7 cscfg和csdef 23](#_Toc453664226)

[4.7.1 ServiceConfiguration.Local.cscfg 24](#_Toc453664227)

[4.7.2 ServiceConfiguration.Cloud.cscfg 24](#_Toc453664228)

[4.7.3 csdef文件 24](#_Toc453664229)

[4.7.4 注意事项 24](#_Toc453664230)

[4.8 配置Azure Cloud Service 24](#_Toc453664231)

[4.8.1 配置Web Role 24](#_Toc453664232)

[4.8.1.1 Configuration 25](#_Toc453664233)

[4.8.1.2 Settings 26](#_Toc453664234)

[4.8.1.3 Endpoints 26](#_Toc453664235)

[4.8.1.4 Local Storage 27](#_Toc453664236)

[4.8.1.5 Certificates 27](#_Toc453664237)

[4.8.1.6 Caching 28](#_Toc453664238)

[4.8.1.7 回顾 28](#_Toc453664239)

[4.8.2 配置Worker Role 28](#_Toc453664240)

[4.8.2.1 Configuration 29](#_Toc453664241)

[4.8.2.2 Settings 30](#_Toc453664242)

[4.8.2.3 Endpoints 30](#_Toc453664243)

[4.8.2.4 Local Storage 31](#_Toc453664244)

[4.8.2.5 Certificates 31](#_Toc453664245)

[4.8.2.6 Caching 31](#_Toc453664246)

[4.8.2.7 回顾 31](#_Toc453664247)

[4.9 查看csdef和cscfg(非常重要) 32](#_Toc453664248)

[4.9.1 查看ServiceDefinition.csdef 32](#_Toc453664249)

[4.9.2 查看ServiceConfiguration.Cloud.cscfg(非常重要) 33](#_Toc453664250)

[4.10 使用模拟器调试(非常重要) 34](#_Toc453664251)

[4.10.1 修改配置 34](#_Toc453664252)

[4.10.2 修改代码(非常重要) 35](#_Toc453664253)

[4.10.3 观察模拟器 36](#_Toc453664254)

[4.10.4 代码运行结果(非常重要) 37](#_Toc453664255)

[4.11 Azure Cloud Service打包(非常重要) 38](#_Toc453664256)

[4.11.1 Cloud Service打包 39](#_Toc453664257)

[4.11.2 CSPKG 39](#_Toc453664258)

[4.11.3 CSCFG 41](#_Toc453664259)

[4.12 Azure Cloud Service发布 41](#_Toc453664260)

[4.12.1 使用Azure PowerShell下载发布凭据 41](#_Toc453664261)

[4.12.2 使用Visual Studio导入发布凭据 42](#_Toc453664262)

[4.12.3 发布Cloud Service并配置远程桌面连接 44](#_Toc453664263)

[4.12.4 观察发布过程和发布结果 48](#_Toc453664264)

[4.12.4.1 发布过程 48](#_Toc453664265)

[4.12.4.2 发布结果 49](#_Toc453664266)

[4.12.5 Cloud Service部署和运行过程(非常重要) 51](#_Toc453664267)

[4.12.5.1 Role在虚拟机上部署和运行的过程 51](#_Toc453664268)

[4.12.5.1 Role Instance的状态 52](#_Toc453664269)

[4.12.6 远程连接Cloud Service 53](#_Toc453664270)

[4.13 Azure Cloud Service是非持久化虚拟机(非常重要) 54](#_Toc453664271)

[4.14 总结(非常重要) 55](#_Toc453664272)

[5. Java开发者必读 55](#_Toc453664273)

[5.1 第一部分 55](#_Toc453664274)

[5.2 第二部分 65](#_Toc453664275)

[6. 高级内容 75](#_Toc453664276)

[6.1 Cloud Service是非持久化虚拟机 75](#_Toc453664277)

[6.2 添加外部组件 75](#_Toc453664278)

[6.3 注册第三方组件 75](#_Toc453664279)

[6.4 修改时区 75](#_Toc453664280)

[6.5 修改IIS托管管道模式为4.0经典模式 76](#_Toc453664281)

[6.6 Cloud Service虚拟目录 76](#_Toc453664282)

[6.7 加快部署速度 76](#_Toc453664283)

[6.8 配置SSL证书 76](#_Toc453664284)

[6.9 固定Cloud Service内网IP和公网IP 76](#_Toc453664285)

[6.10 收集日志记录 76](#_Toc453664286)

[6.11 Web Role和Worker Role交互 76](#_Toc453664287)

版本变更：

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 版本号 | 变更内容 | 作者 |
| 2015-12-3 | 1.00 | 文档创建，设置文档目录结构 | leizha |
| 2015-06-15 | 1.01 | 增加Steven Lian写的Java内容 | leizha |

# 前言

**个人分享，仅作为参考资料，非官方解答。**

**Demo不给力？请访问我的个人博客**

[**http://www.cnblogs.com/threestone/archive/2012/01/06/2382322.html**](http://www.cnblogs.com/threestone/archive/2012/01/06/2382322.html)

# 读者

Windows Azure是平台产品，本文只详细介绍如何创建和管理Azure Cloud Service。本文适合开发人员阅读。

另外本文会牵涉到部分Azure订阅相关内容，请读者先阅读《Azure管理手册》。<http://www.cnblogs.com/threestone/p/4627388.html>

# Azure Cloud Service相关服务

## Azure Cloud Service的带宽问题

这个问题长话短说，请参考我的博客：

<http://www.cnblogs.com/threestone/p/4497625.html>

## 什么是Azure Cloud Service

我们在平时开发Web应用程序的时候，都需要IT人员准备Windows OS或者Linux OS的Web Server，安装相应的Web组件，比如IIS, Tomcat等等。然后开发人员把相应的代码部署到Web Server上并进行配置。

对于IT运维人员来说，Web Server是IaaS (Infrastructure-as-a-Service，基础设置即服务)，IT运维人员需要维护Web Server的操作系统等内容。

对于开发人员来说，Web Server是PaaS(Platform-as-a-Service，平台即服务)，开发人员只需要维护Web Application即可。底层的操作系统等交给IT运维人员。

同样的道理，Azure Cloud Service提供PaaS服务。开发人员只需要把开发的代码直接部署到Azure Web Apps，无需管理操作系统，降低了管理的成本。

## Azure Cloud Service和Virtual Machine的关系

最近在和一些客户聊天，常常被问到这样的问题:

1. 问题一：我在创建一个新的Windows Azure Virtual Machine的虚拟机时候，会同时创建同样名称的Cloud Service(云服务)。

我看微软的报价里说虚拟机会收费，云服务也会收费。这样的话，我使用虚拟机，是不是收取我虚拟机+云服务=2倍的费用？

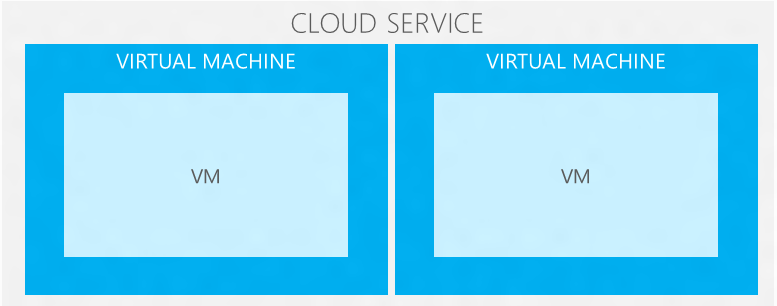
1. 问题二：我在使用VS2013，将asp.net的应用程序部署到微软的PaaS平台的时候，只会有Cloud Service，没有Virtual Machine，是为什么呢?

在这里，我就以上的问题简单的回答一下。

首先，什么是Cloud Service？

从我个人的理解，Cloud Service其实有两层的含义：

1. 第一层含义，对于IaaS平台(Azure虚拟机)来说，Cloud Service是容纳虚拟机的容器(container)。如下图



对于上图来说，Cloud Service是一个容器，可以同时容纳两个虚拟机。

当我们新建一个虚拟机的时候，因为不存在容纳这个虚拟机的容器，所以Windows Azure会默认创建一个新的cloud service，然后将虚拟机加入到这个容器当中去。

那微软是否会收取Cloud Service + Virtual Machine两份费用呢?这个我专门求证过，微软只会收取Virtual Machine的费用。对于上图中的例子来说，微软只会收取2个virtual machine的费用。

1. 第二层含义，对于PaaS Cloud Service来说，Cloud Service是一个多层的Web应用程序。

**本文主要介绍的是第二层含义，PaaS服务，多层的Web应用程序**

用户可以定义前端的Web Role，用来响应客户端的请求；还可以定义后端的Worker Role，用来处理复杂的业务逻辑。

因为Azure PaaS平台使用的是Web Role和Worker Role，并不存在任何的虚拟机。所以在使用PaaS平台的时候，不会创建虚拟机。

假设一个用户在使用PaaS平台的虚拟机，创建了2台A2的Web Role和2台A2的Worker Role，那该用户需要支持的费用=2 \* A2单价 + 2 \* A2单价=4 \* A2的单价费用。

接下来是DEMO时间，笔者将介绍如何查看一个Cloud Service，背后是否是PaaS服务，还是IaaS虚拟机呢?

1. 首先我们登陆Azure Management Portal <https://manage.windowsazure.cn>
2. 点击Cloud Service(云服务)，如下图：



如果发现云服务，实例中，显示生产、过渡，且名称无法点击跳转的，则这个服务器必定是PaaS Cloud Service服务。

1. 点击Cloud Service(云服务)，如下图：



如果发现云服务，实例中，没有显示生产、过渡，且名称可以点击跳转的，则这个服务器必定是IaaS虚拟机。

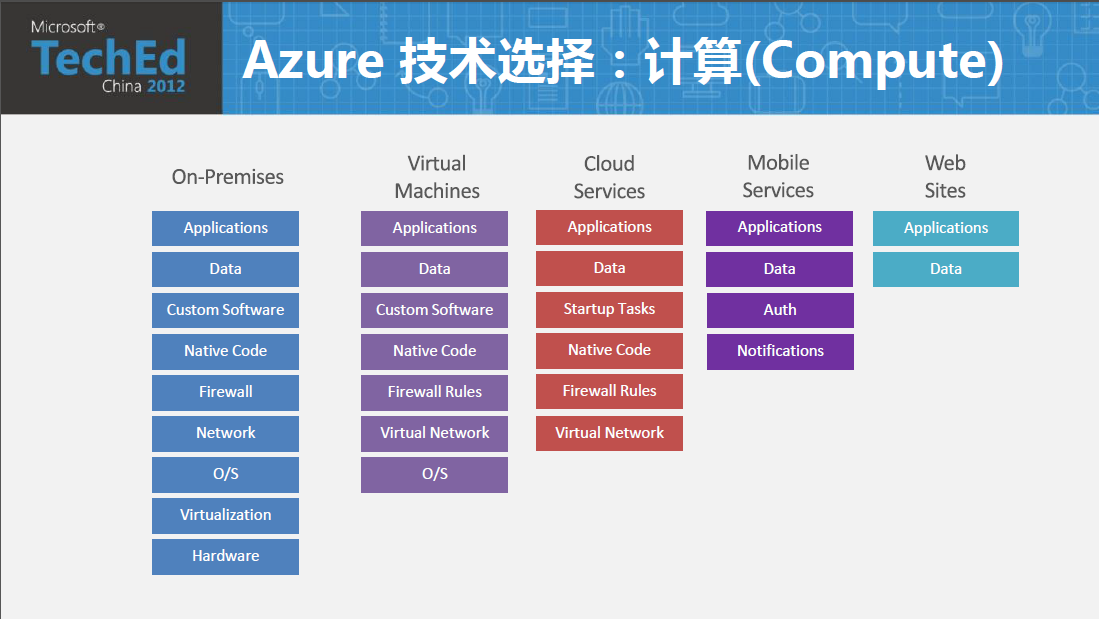
## Azure Cloud Service和其他计算服务的区别

所谓计算服务，就是提供CPU和内存的计算能力。

熟悉Windows Azure Platform的读者都知道，在Azure平台的计算服务分为三种：

1. Azure Web Apps
2. Azure Virtual Machine
3. Azure Cloud Service

如果我是一个企业级的用户，面对这三种不同的托管服务，我应该选择哪一种服务，将现有的企业应用迁移到Windows Azure云计算平台上呢？

在这里我不妨给大家介绍一下Windows Azure三种服务的区别，如下图：

以上三种计算服务，优势和使用场景如下表：

|  |  |
| --- | --- |
| 计算服务 | 优势、劣势和使用场景 |
| Virtual Machine | **优势：**   1. 同时支持Windows和Linux虚拟机 2. 支持托管机房Hyper-V VHD迁移到云端 3. 支持Virtual Network，将企业内网与云端网络连接，实现混合云 4. 支持高级存储(SSD,固态硬盘) 5. 单个虚拟机最大计算资源16Core / 112GB   **劣势：**   1. 需要IT运维人员，管理操作系统 2. 使用成本相对其他计算服务来说，比较昂贵   **使用场景：**   1. 适合传统应用的迁移 2. 需要使用Linux操作系统的应用部署 3. 将多个应用程序部署在同一台VM上，比如SQL Server, MySQL, MongoDB, SharePoint 4. 企业内网+云端的混合云部署 |
| Web App | **优势：**   1. 在Windows Azure上构建高度可扩展的Web站点 2. 通过FTP, Git或者TFS快速部署   **劣势：**   1. 不支持管理操作系统 2. 单个节点最大计算资源4Core / 7GB 3. 横向扩展最多10个节点负载均衡 4. 国内由世纪互联运维的Azure China，暂时不支持将Web App加入到Virtual Network虚拟网络里   **使用场景：**   1. 新应用的开发部署 2. 持续开发。直接从源代码库使用Git或者Team Foundation进行部署 |
| Cloud Service | 优势：   1. 利用丰富的PaaS环境，创建高度可用的，可扩展的应用程序和服务。支持高级的多层架构，自动部署和弹性计算 2. 使用Web Role和Worker Role，Web Role可以响应前端的显示，而将复杂的业务放在后端进行处理 3. 支持Virtual Network，将企业内网与云端网络连接，实现混合云 4. 相比 Web App，可以使用Startup Task在PaaS VM注册第三方组件、安装外部软件 5. 单个计算节点，最大计算资源16Core / 112GB   劣势：   1. 不支持管理操作系统 2. 非持久化虚拟机   使用场景：   1. 新应用的开发部署 2. 企业内网+云端的混合云部署 3. 多层的应用程序，每层都可以自我扩展。使用Web Role和Worker Role，Web Role可以响应前端的显示，而将复杂的业务放在后端进行处理。 |

## Azure Cloud Service支持的开发语言

Azure Web Apps支持的开发语言包括：.NET, Java, PHP, Python, Node.js, Ruby。

开发Azure Cloud Service必须先安装Azure SDK，下载地址是：

<http://www.windowsazure.cn/downloads/>

## Azure Cloud Service如何解决大并发

我们建议使用多台Azure Cloud Service，利用横向扩展的方式来解决大量的并发。

单个节点向上扩展是有限的，这是因为受限于现有的CPU制造技术，我们无法将大量的计算资源都堆积到1台300 Core甚至400 Core的计算节点上。对于需要大量的计算资源的情况下，我们可以通过横向扩展的方法来解决。

所谓横向扩展，就是由1个计算节点，横向扩展到多个计算节点上并行计算，比如50个、100个计算节点。比如一个互联网业务需要大量的计算资源，那可以将这些计算需求由100台4 Core的计算节点进行并行计算。

## Azure Cloud Service单个实例大小

Azure Cloud Service单个实例，分为A系列和D系列：

注意：Azure Cloud Service计算节点的CPU和RAM是固定搭配的，不可以按照用户的想法随意更改。

### A系列计算节点

A系列计算节点的类型如下：

|  |  |  |
| --- | --- | --- |
| 虚拟机类型 | CPU | 内存 |
| A0 | 共享 | 768 MB |
| A1 | 1 | 1.75 GB |
| A2 | 2 | 3.5GB |
| A3 | 4 | 7 GB |
| A4 | 8 | 14 GB |
| A5 | 2 | 14 GB |
| A6 | 4 | 28 GB |
| A7 | 8 | 56 GB |

除了A0的虚拟机类型，它的CPU是和别的用户共享的。其他类型的虚拟机，比如A1-A7，它的CPU是独占的，不是和别的用户共享的。

### D系列计算节点

D系列虚拟机的类型如下：

|  |  |  |
| --- | --- | --- |
| 虚拟机类型 | CPU | 内存 |
| D1 | 1 | 3.5 GB |
| D2 | 2 | 7 GB |
| D3 | 4 | 14 GB |
| D4 | 8 | 28 GB |
| D11 | 2 | 14 GB |
| D12 | 4 | 28 GB |
| D13 | 8 | 56 GB |
| D14 | 16 | 112 GB |

相比A系列的Azure Virtual Machine，D系列的优势在于：

1. **单台VM更多的CPU Core**

相比A系列单台VM最大8Core/56GB RAM的配置，D系列虚拟机单台最大的配置为16Core/112GB RAM

1. **D系列的CPU性能比A系列提升约60%**

我从后台得到的消息，Azure A系列的虚拟机，Intel E5的CPU是经过调试的，性能是人为降低的。

因为Azure数据中心在建设的时候，有AMD的CPU和Intel的CPU，为了让Intel CPU性能和AMD CPU性能接近，保证运算能力的一致性，人为降低了Intel CPU的计算能力。

所以各位如果用Super PI等测试A系列的虚拟机，会发现性能与物理机是有差距的。

最新的D系列虚拟机，是100%体现Intel E5的处理能力，CPU性能比A系列提升60%。

1. **Azure Cloud Service所在的计算资源是无状态的**

熟悉Windows Azure Virtual Machine虚拟机的用户都知道，Azure虚拟机分为不同的类型，分别是A系列、D系列和DS系列。

这里的Azure Cloud Service单个实例大小，是与虚拟机的类型和配置相同的。但是请记住，Azure Cloud Service是没有本地磁盘的概念。

在下面的内容中，读者将看到，虽然我们可以通过Visual Studio，将Cloud Project发布到Azure Cloud Service，同时可以配置远程桌面连接。但是保存在Azure Cloud Service本地磁盘上的资源，都是非持久化保存的。

如果需要保存持久化的文件，一定要将文件使用Windows Azure Storage，切记切记！

## Azure Cloud Service使用静态资源

我遇到很多开发人员向我抱怨，我的Web项目文件包含了很多视频内容，大概1TB，发布到Azure Cloud Service非常慢，我应该怎么办？

我建议把Web项目文件的静态资源，比如图片、照片、视频等都保存在Azure Storage里，这样的好处有以下几点：

1. 加快网站部署速度

去除掉静态资源(图片、照片、视频)的Web项目文件只包含了源代码信息，文件容量小了，上传速度就加快了

1. 静态资源可以直接访问Azure Storage

通过将静态内容请求发送到Azure Storage，将动态内容的请求发送到Azure 云主机，就可以大大减少云主机独享带宽的压力。

可以参考之前项目的案例分享：

<http://www.cnblogs.com/threestone/p/4497625.html>

## Azure Cloud Service成本分析

Azure Cloud Service是按照分钟计费的，计费的单价为每小时。

Azure Cloud Service在运行的时候，会收取计算费用。关闭Cloud Service后，会继续收取计算费用。

举例来说，比如某个客户采用10台A7配置的微软Azure Cloud Service，对外提供服务，每天开始10小时。

**每天计算费用=10 (Azure Cloud Service数量) X 每台每小时费用 X 10小时**

## Azure Cloud Service停止计费(非常重要)

**最后强调一下，当我们通过Azure管理界面，关闭Azure Cloud Service的时候，还是会继续计费的。如下图：**

****

**在上图中，虽然Web Role和Worker Role状态变为”已停止”，但是Cloud Service还是会继续收费。**

**为了避免继续产生费用，请将不使用的Azure Cloud Service删除。这样才不会继续收费。**

## Azure Cloud Service限制

1. 单个Azure Cloud Service大小

单个Azure Cloud Service实例，最大为D14 (16Core / 112GB)。

1. 只支持Windows Server 2012 R2操作系统

Azure Cloud Service背后运行的是非持久化虚拟机，操作系统是Windows Server 2012 R2。

1. 如果客户需要的计算单元，必须使用Linux操作系统，请使用Azure虚拟机。
2. 不支持安装软件

Azure Cloud Service背后运行的非持久化虚拟机，任何安装软件的操作都是没有意义的。

1. 非持久化虚拟机

虽然我们可以通过Visual Studio，将Cloud Service发布到Azure Cloud Service，同时可以配置远程桌面连接。但是保存在Azure Cloud Service本地磁盘上的资源，都是非持久化保存的。

1. 如果需要保存持久化的文件，一定要将文件使用Windows Azure Storage，切记切记！

# 开始创建Azure Cloud Service

## 规划好Azure订阅(非常重要)

订阅是进行Azure账单分拆的最小单位。

如果企业内部需要进行内部成本核算，例如IT部门、销售部门、市场部门均需要使用Azure，并且根据不同的部门的Azure实际使用量进行内部成本核算，就需要实现规划好三个不同的Azure订阅。在创建Azure IaaS相关资源的时候，将这些资源创建在不同的订阅下。

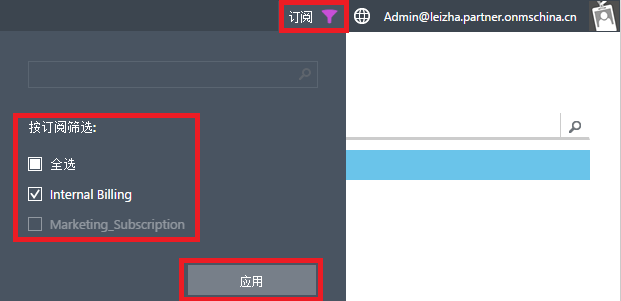
笔者之前写了一篇《Azure企业门户管理手册》，具体请参考

<http://www.cnblogs.com/threestone/p/4627388.html>

## 选择订阅(非常重要)

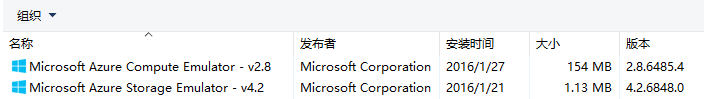
我们登陆Azure管理界面(http://manage.windowsazure.cn)，输入账户和密码。

点击右上角的订阅按钮，如下图:



## Azure Emulator模拟器

当我们安装完Azure SDK后，默认会安装Azure Emulator模拟器。如下图：



上图中，模拟器分为两种：

1. Compute Emulator

计算模拟器。通过该模拟器，模拟Cloud Service在云端执行的情况

1. Storage Emulator

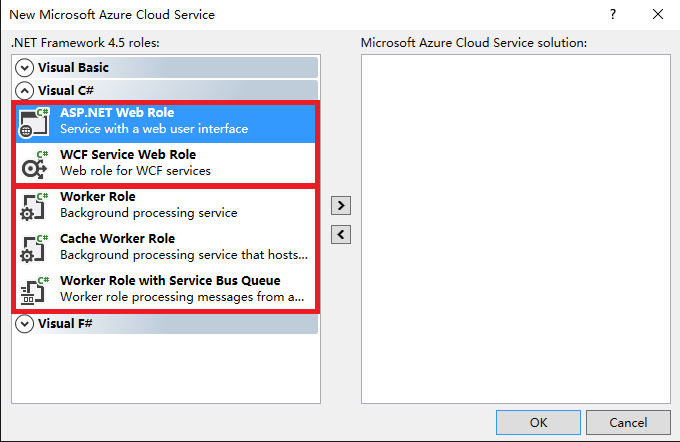
存储模拟器。通过该模拟器，模拟Azure Storage执行情况

当我们需要在本地调试的时候，可以直接使用Azure Emulator，模拟Cloud Service在云端执行的大部分功能。

## Web Role / Worker Role

我们知道，在开发Azure Cloud Service项目，有一个非常重要的概念，就是Web Role和Worker Role。

在创建Azure Cloud Service的时候，会提示如下的选项：



1. Web Role是响应客户端的HTTP请求
2. Worker Role则是在后台执行的应用程序，概念上类似于Windows Service
3. Web Role可以通过Azure Storage Queue，向Worker Role发送队列消息 (Queue.AddMessage())，以便让Worker Role在后端执行自己需要的逻辑。

一个Cloud Service可以

1. 只有Web Role，没有Worker Role
2. 只有Worker Role，没有Web Role
3. 既有Web Role，又有Worker Role

注意：上图中的Worker Role分为三类(Worker Role, Cache Worker Role和Worker Role with Service Bus Queue)。本文只会介绍Worker Role。

为什么微软要有Web Role/Worker Role？它的好处在哪里？在这里我举个例子您就能明白，比如我们有一个信息管理系统，需要上传Excel文档来进行解析和处理，从软件设计的角度来说有2种方法来解决

1. 在ASP.NET应用程序里新建个upload control,在upload control里面写函数：一旦Excel文件上传完毕，则在.cs文件执行对于Excel的处理工作。

但这样会有一个缺点，如果Excel文件里包含的内容非常大的话，需要时间来处理这些内容，所以前台的ASP.NET的页面会停滞或者无响应。虽然我们也可以通过增加progressbar或者loading图片来增强用户的体验，但是从软件设计上来说不是最好的。

1. 前端还是用原来的处理方式，使用upload control。服务器端增加一个Windows Service，时序的查询某一个文件夹，一旦发现前端页面上传了一个Excel文件，则Windows Service执行处理Excel的工作。这样前端的页面会及时的响应并且得到更好的用户体验。

但是这还是有一个缺陷，前端的页面和windows service是一对一的关系，如果附件上传的数量很大的话会出现Windows Service来不及处理的情况。

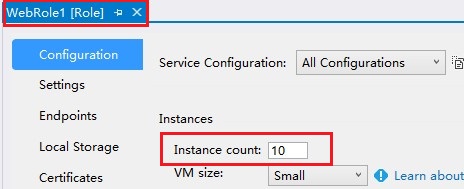
1. 有了Worker Role，我们可以让一个ASP.NET页面后端有多个Worker Role来进行分布式的计算，是一对多的关系，能够有效的利用云上的计算资源，Worker Role可以处理高负载的数据访问。

这样的架构好处有亮点：

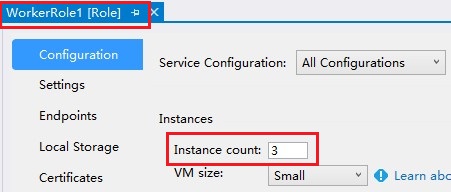
1. 异步处理，Web Role只响应客户端的HTTP请求，进行快速的响应。而Worker Role在后端处理Web Role发送过来的消息(Queue)，两者是松耦合的。

在传统的Web应用中，如果我们把复杂的处理逻辑写在ASPX页面，则ASPX页面可能会停滞，造成的用户体验不好。Windows Azure PaaS使用Web Role和Worker Role，Web Role只关注于响应客户端的HTTP请求；而Worker Role可以在后端处理业务逻辑，进行异步处理。

1. Web Role和Worker Role的关系是多对多的，比如我可以在Web Role的配置中，设置Instance count为10。如下图：



在Worker Role的配置中，设置Instance Count为3



这种架构就好比一个餐厅，里面有10个服务员(Web Role)和3个厨师(Worker Role)。

1. 服务员(Web Role)只负责招待客人(响应客户端请求)，并将客人的点菜信息通过队列消息交给厨房(Queue.AddMessage())
2. 厨师(Worker Role)读取点菜信息(Queue.GetMessage)，随后负责烧菜做饭(进行后端逻辑)，当饭菜准备完毕后，则删除点菜信息。
3. 如果前端压力过大(客户端请求过多)，则Web Role可以横向扩展

如果后端压力过大(后端逻辑处理过多)，则Worker Role也可以横向扩展

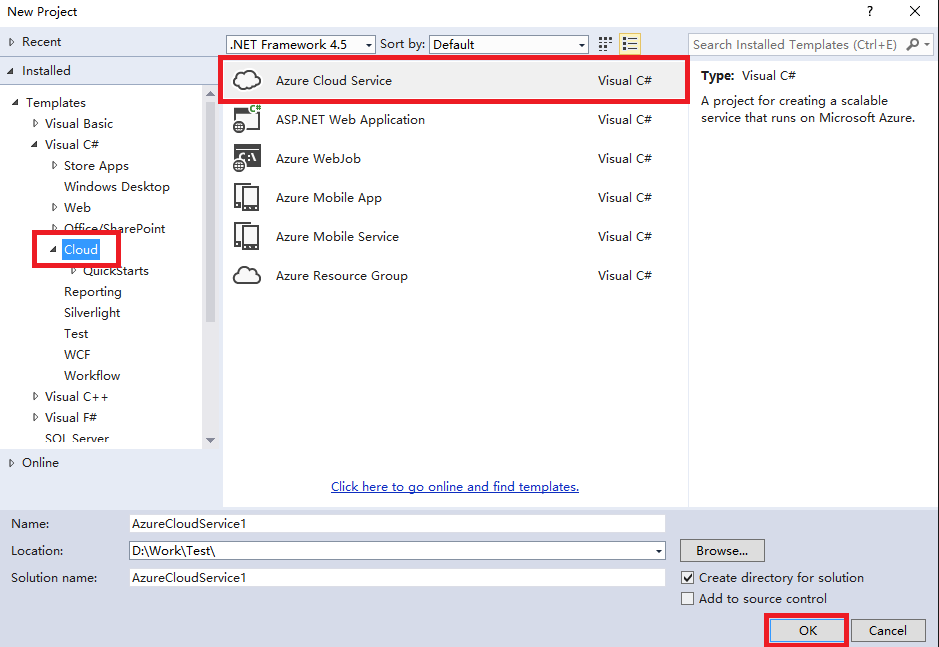
1. 这种松耦合，多对多的关系非常适合企业级应用架构

可以看到，Web Role和Worker Role的进行通信的重要途径是通过Azure Storage Queue，了解Queue对于Azure PaaS架构设计非常重要。

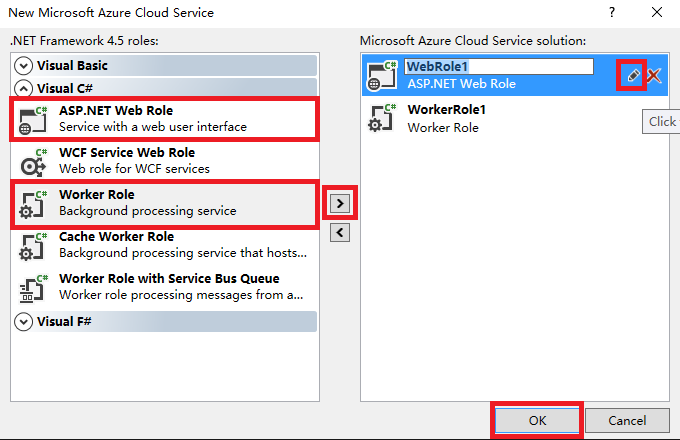
## 创建一个Azure Cloud Service

在开始以下内容之前，请读者准备：

1. Azure China账户
2. Visual Studio 2013
3. 安装Azure SDK，下载地址：<http://www.windowsazure.cn/downloads/>
4. 首先我们以管理员身份，运行Visual Studio 2013
5. 点击File -> New -> Project
6. 在弹出的界面中，选择Cloud -> Azure Cloud Service



1. 点击上图的OK按钮。依次添加ASP.NET Web Role和Worker Role。



上图中，我们还可以对Web Role和Worker Role进行重命名。

一个Cloud Service可以

1. 只有Web Role，没有Worker Role
2. 只有Worker Role，没有Web Role
3. 既有Web Role，又有Worker Role

上图中，请注意：

1. Azure Cache Worker Role功能会在2016年11月30日下线。

<https://azure.microsoft.com/zh-cn/documentation/articles/cache-dotnet-how-to-use-in-role/>

如果客户想使用Cache缓存服务，请使用Azure Redis Cache。

参考资料：<http://www.cnblogs.com/threestone/category/741409.html>

如果客户有兴趣想了解Cache Worker Role，请参考：

[Windows Azure Cloud Service (42) 使用Azure In-Role Cache缓存(1)Co-located Role](http://www.cnblogs.com/threestone/p/4367859.html)

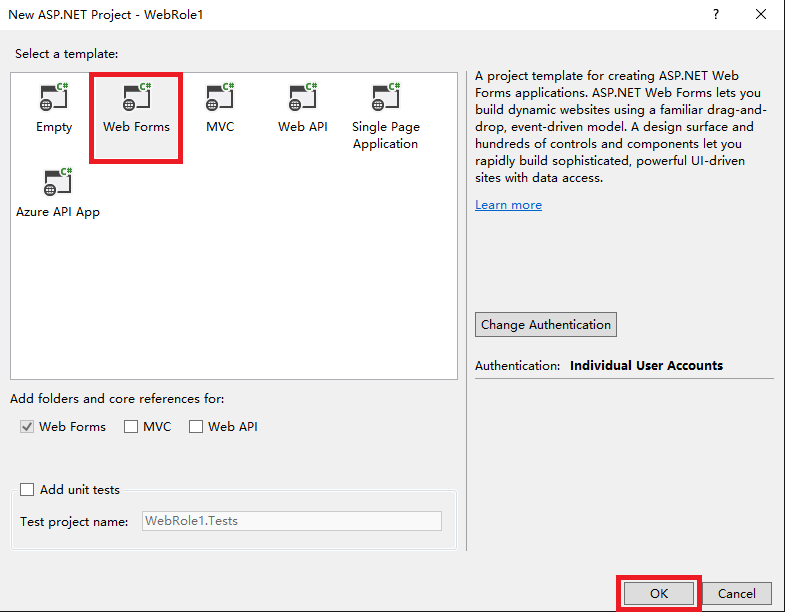
[Windows Azure Cloud Service (43) 使用Azure In-Role Cache缓存(2)Dedicated Role](http://www.cnblogs.com/threestone/p/4368633.html)

1. 有关Worker Role with Service Bus Queue，请参考

[Windows Azure Service Bus (2) 队列(Queue)入门](http://www.cnblogs.com/threestone/p/3968442.html)

[Windows Azure Service Bus (3) 队列(Queue) 使用VS2013开发Service Bus Queue](http://www.cnblogs.com/threestone/p/4308998.html)

1. 然后我们可以根据需求，选择相应的ASP.NET模板。这里我们选择Web Forms，如下图：



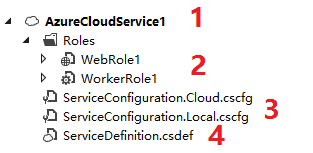
## 查看Azure Cloud Service

一个Azure Cloud Service Solution解决方案，包含三个Project，分别是:

1. Azure Cloud Service
2. Web Role
3. Worker Role

### Cloud Service

如下图所示，是Azure Cloud Service的项目结构：



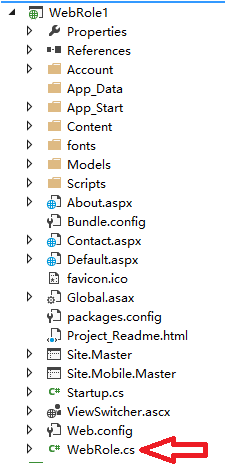
从目录结构上，我们可以观察到：

1. 这个项目不同于传统的Visual Studio Project，是一个Cloud Service。(带一个云的Logo ☺)
2. 这个Cloud Service包含2个Role，分别是Web Role和Worker Role
3. 这个Cloud Service包含了2个配置文件(CSCFG, Cloud Service Configuration)，文件名分别是Cloud和Local
4. 这个Cloud Service包含了1个定义文件(CSDEF, Cloud Service Definition)

那Web Role和Worker Role的代码在哪里呢？请往下看

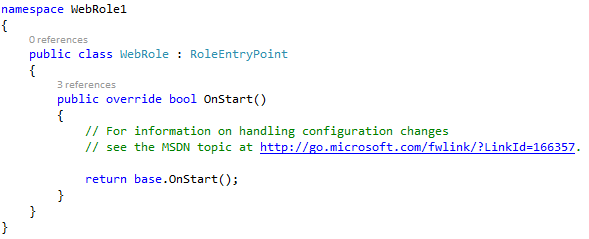
### Web Role

Web Role项目结构的话，请大家看下图：



上图中，就是我们熟悉的Web Form项目，有Default.aspx，Global.asax，Web.config。

但是这个项目文件包含了WebRole.cs，代码如下：



WebRole.cs的代码逻辑是：

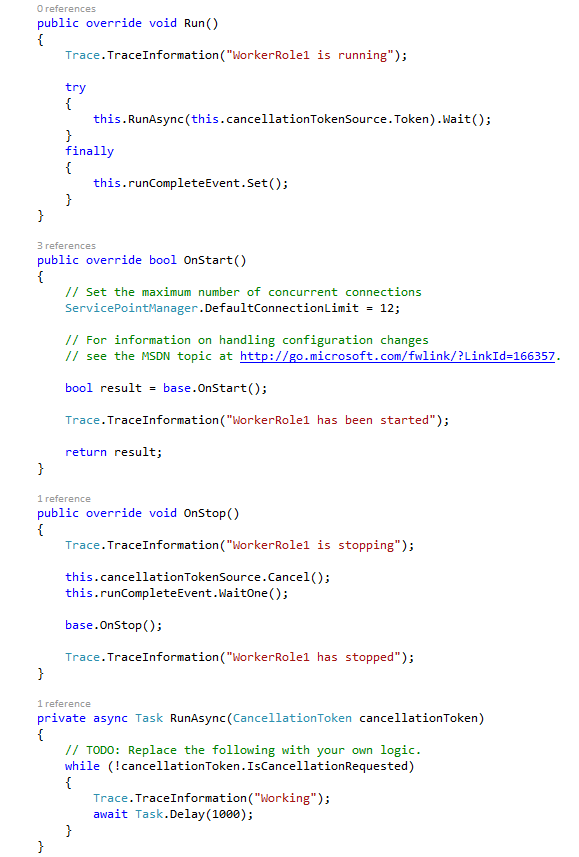
1. OnStart()方法，在Web Role启动的时候执行
2. 我们还可以根据自己的需求，override OnStop()方法和Run()方法

### Worker Role

Worker Role的项目文件，请参考下图：



WorkerRole.cs的代码片段如下：



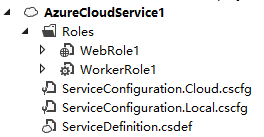
上图的2个代码片段我们可以看到，Worker Role的代码逻辑是：

1. OnStart()方法，在Worker Role启动的时候执行
2. Run()方法，在Worker Role运行的时候执行
3. Run()方法，包含了一个异步执行的方法RunAsync。具体的业务逻辑是，每1000毫秒，输出内容”Working”。我们可以根据自己的需求，修改RunAsync()方法的具体业务逻辑。
4. OnStop()方法，在Worker Role关闭的时候执行

从上面的代码片段中，我们可以观察到Worker Role.cs的代码逻辑就像是一个Windows Service，自己在背后默默的运行。

### 回顾

我们回顾一下本节的内容：



1. 我们创建的Cloud Service，有2个Role，Web Role和Worker Role
2. Web Role是前端进行展示的用户界面，使用的ASP.NET Web Form代码
3. Worker Role是在后端默默执行的函数，概念上类似于Windows Service，只有一个类WorkerRole.cs

## cscfg和csdef

上一节中，我们对Azure Cloud Service项目做了一个简单的介绍，了解了项目的基本构成，接下来我们需要了解cscfg文件和csdfg文件。

在介绍下面的内容之前，笔者先给大家介绍2个开发的概念，接口(Interface)和类(Class)。

1. 接口定义了行为
2. 类继承了接口，实现了接口中具体的行为

cscfg和csdef也是相同的概念：

1. csdef，全称是Cloud Service Definition，是定义了Cloud Service的参数，概念上类似于接口
2. cscfg，全称是Cloud Service Configuration，是配置了Cloud Service的参数，概念上类似于类

在Azure Cloud Serivce里，配置文件(Cloud Service Configuration)分为两种，Cloud和Local。

### ServiceConfiguration.Local.cscfg

当我们需要在本地通过Azure Emulator模拟器，调试Azure Cloud Service的时候，可以把调试的参数信息保存到Local.cscfg里。

### ServiceConfiguration.Cloud.cscfg

当Cloud Service在本地调试通过后，需要部署到Azure云端执行的时候，可以把Azure云端的参数，保存到Cloud.cscfg里。

通过使用Local.cscfg和Cloud.cscfg，一个Azure Cloud Service可以对应本地模拟器环境和云端环境，分别读取不同的配置参数信息。方便我们进行debug调试和生产部署。

### csdef文件

csdef文件，全称是Cloud Service Definition。定义了Cloud Service的参数，概念上类似于接口。这里不再做介绍。

### 注意事项

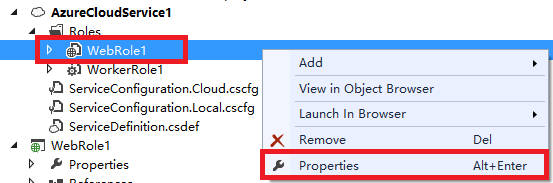
请读者注意，我们在修改cscfg和csdef的时候，最好通过Visual Studio等集成开发环境(Integrated Development Environment, IDE)来进行修改。不要通过记事本等进行修改。

一旦我们在csdef中定义的参数，没有在cscfg中进行配置，会产生Visual Studio无法识别cscfg和csdef，而造成配置失败的情况。

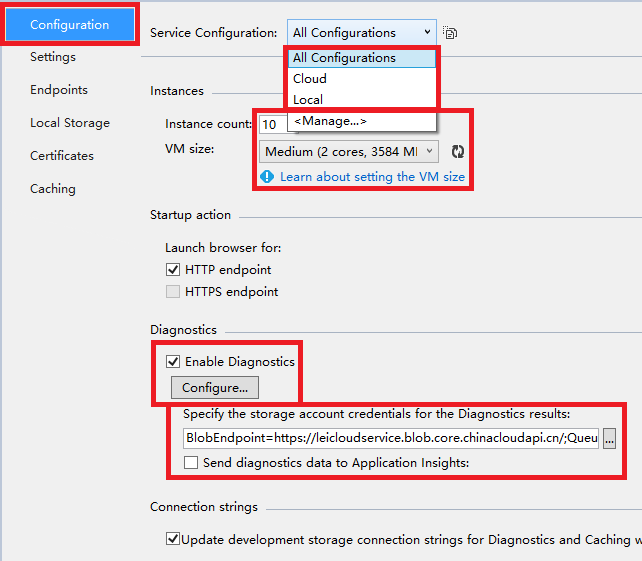
## 配置Azure Cloud Service

### 配置Web Role

我们点击Web Role，右键，属性。如下图：



#### Configuration

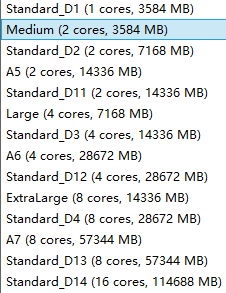


上图中，说明了Azure Cloud Service的Web Role配置信息:

1. Service Configuration，表示配置文件保存到Cloud.cscfg还是Local.cscfg
2. Instance Count，表示Web Role横向扩展的节点数量
3. VM Size，表示Web Role在横向扩展的时候，每一个计算节点的CPU内存信息
4. Enable Diagnostics，表示在Azure Cloud Service Web Role启用诊断功能，我们可以点击Configure按钮，设置需要诊断的具体参数
5. Specify the storage account credentials for the Diagnostics result，表示我们需要把诊断的结果数据，保存到Azure Storage Table中。这里我们需要配置Azure Storage的连接字符串。

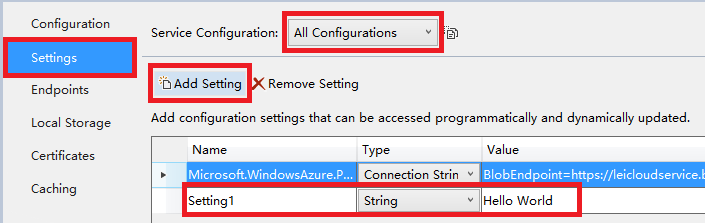
请注意，Azure Cloud Service背后运行的是非持久化虚拟机，任何保存在非持久化虚拟机本地磁盘的文件都会有丢失风险。所以我们需要把诊断数据保存到Azure Storage Table中。

1. 以上图为例，我们把Web Role横向扩展为10台，每台计算节点的CPU内存为2Core/3.5GB (A系列)。如果是D系列虚拟机，显示如下：



#### Settings

我们点击Settings页面，显示如下：

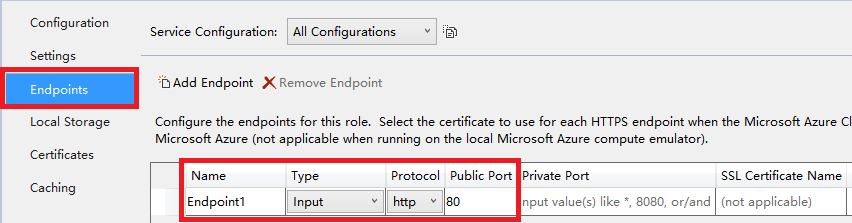


上图中：

1. Service Configuration，表示配置文件保存到Cloud.cscfg还是Local.cscfg
2. 一般情况下，我们把ASP.NET Web Form的配置信息保存到Web.config文件里。但是如果多台Web Role在横向扩展的时候，我们需要手工一台一台更新每个Web.config文件，费时费力。
3. 笔者建议对于Web Role的配置信息，以前是保存到Web.config文件里。现在可以保存到Web Role的Setting页面里。这样修改了配置文件，可以在多台的Web Role实例同时生效。我会在后面有DEMO演示
4. 在上图中，笔者点击Add Setting，增加一个参数名Setting1，参数值为Hello World

#### Endpoints

我们点击Endpoints，如下图:

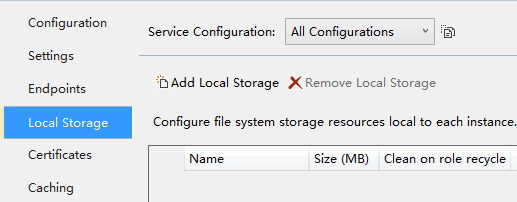


上图中：

1. 默认只有Endpoint1，Public Port为80
2. 表示Web Role默认打开的端口为80，协议为http
3. 我们可以根据需要，设置打开8080等其他端口，并且可以修改协议为https

#### Local Storage

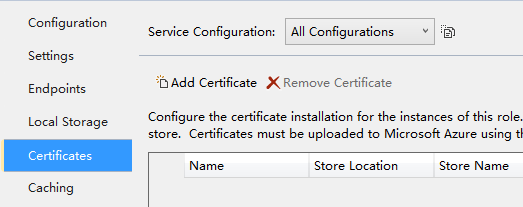
我们点击Local Storage，如下图:



使用默认的设置即可，不需要进行修改

#### Certificates

我们点击Certificates，如下图：



上图中，我们可以上传Web Role通过HTTPS访问需要的证书。具体笔者就不在这里演示了，请参考笔者的Blog:

<http://www.cnblogs.com/threestone/p/3951227.html>

#### Caching

该配置项是为了Azure Cache Worker Role使用的，

但是Azure Cache Worker Role功能会在2016年11月30日下线。

<https://azure.microsoft.com/zh-cn/documentation/articles/cache-dotnet-how-to-use-in-role/>

如果客户想使用Cache缓存服务，请使用Azure Redis Cache。

参考资料：<http://www.cnblogs.com/threestone/category/741409.html>

#### 回顾

本节中，我们对Web Role进行了以下配置。

在Configuration进行了如下的配置：

1. Web Role需要横向扩展，一共10个节点
2. 每个节点的硬件配置为A2 (2Core / 3.5GB)
3. Enable Diagnostics，表示在Azure Cloud Service启用诊断功能
4. Specify the storage account credentials for the Diagnostics result，表示我们需要把诊断的结果数据，保存到Azure Storage Table中

在Settings进行了如下的配置：

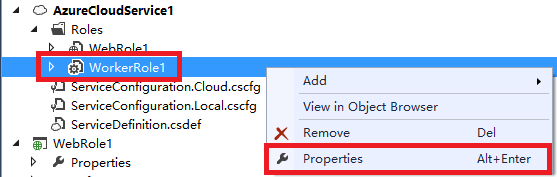
1. 增加了一个参数，参数名Setting1，参数值为Hello World

在Endpoints进行了如下配置：

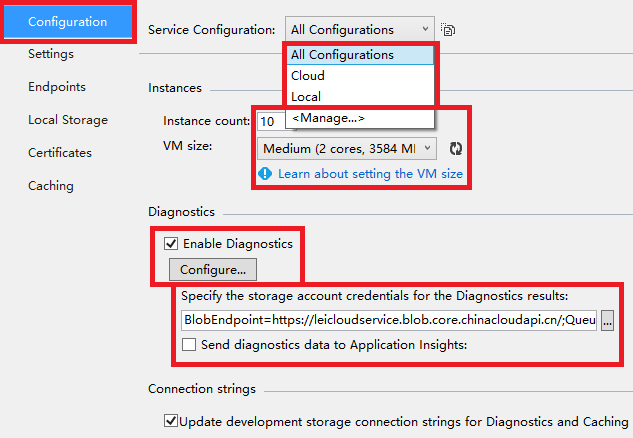
1. Web Role默认打开的端口为80，协议为http

### 配置Worker Role

我们点击Worker Role，右键，属性。如下图：



#### Configuration

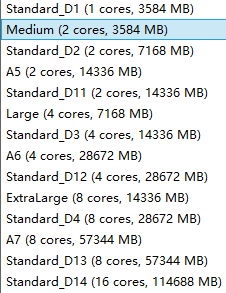


上图中，说明了Azure Cloud Service的Worker Role配置信息：

1. Service Configuration，表示配置文件保存到Cloud.cscfg还是Local.cscfg
2. Instance Count，表示Worker Role横向扩展的节点数量
3. VM Size，表示Worker Role在横向扩展的时候，每一个计算节点的CPU内存信息
4. Enable Diagnostics，表示在Azure Cloud Service Worker Role启用诊断功能，我们可以点击Configure按钮，设置需要诊断的具体参数
5. Specify the storage account credentials for the Diagnostics result，表示我们需要把诊断的结果数据，保存到Azure Storage Table中。这里我们需要配置Azure Storage的连接字符串。

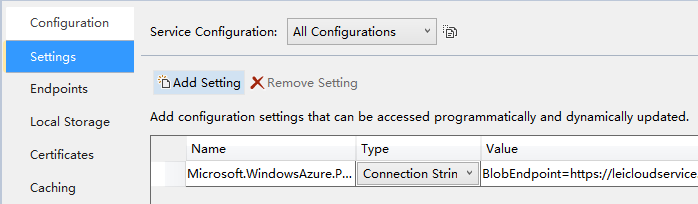
请注意，Azure Cloud Service背后运行的是非持久化虚拟机，任何保存在非持久化虚拟机本地磁盘的文件都会有丢失风险。所以我们需要把诊断数据保存到Azure Storage Table中。

1. 以上图为例，我们把Worker Role横向扩展为10台，每台计算节点的CPU内存为2Core/3.5GB (A系列)。如果是D系列虚拟机，显示如下：



#### Settings

我们点击Settings页面，显示如下：

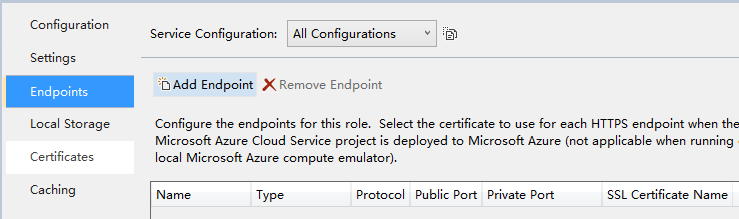


上图中：

1. Service Configuration，表示配置文件保存到Cloud.cscfg还是Local.cscfg
2. 我们这里不增加任何的参数

#### Endpoints

我们点击Endpoints，如下图：



上图中：

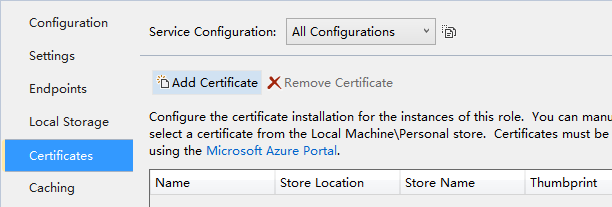
1. 默认没有任何的Endpoint，也没有Public Port
2. Internet用户无法直接访问到Azure Worker Role

#### Local Storage

使用默认的设置即可，不需要进行修改

#### Certificates

我们点击Certificates，如下图：



上图中，我们可以上传Web Role通过HTTPS访问需要的证书。具体笔者就不在这里演示了，请参考笔者的Blog:

<http://www.cnblogs.com/threestone/p/3951227.html>

#### Caching

该配置项是为了Azure Cache Worker Role使用的，

但是Azure Cache Worker Role功能会在2016年11月30日下线。

<https://azure.microsoft.com/zh-cn/documentation/articles/cache-dotnet-how-to-use-in-role/>

如果客户想使用Cache缓存服务，请使用Azure Redis Cache。

参考资料：<http://www.cnblogs.com/threestone/category/741409.html>

#### 回顾

本节中，我们对Worker Role进行了以下配置。

在Configuration进行了如下的配置：

1. Worker Role需要横向扩展，一共10个节点
2. 每个节点的硬件配置为A2 (2Core / 3.5GB)
3. Enable Diagnostics，表示在Azure Cloud Service启用诊断功能
4. Specify the storage account credentials for the Diagnostics result，表示我们需要把诊断的结果数据，保存到Azure Storage Table中

在Worker Role的Endpoints配置页面里，没有任何的Endpoint，也没有Public Port。Internet用户无法直接访问到Azure Worker Role

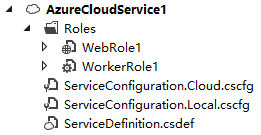
## 查看csdef和cscfg(非常重要)

我们在前几节内容中已经介绍了，Azure Cloud Service的cscfg和csdef有如下关系：

1. csdef，全称是Cloud Service Definition，是定义了Cloud Service的参数，概念上类似于接口
2. cscfg，全称是Cloud Service Configuration，是配置了Cloud Service的参数，概念上类似于类

### 查看ServiceDefinition.csdef

我们通过Visual Studio 2013双击打开ServiceDefinition.csdef文件，如下图：



打开后的显示结果如下：



上图中，我们定义了以下内容：

项目包含了Web Role，计算节点大小为Medium (A2, 2Core / 3.5GB)

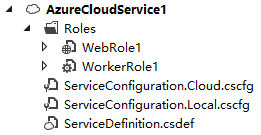
1. Web Role项目包含了访问节点Endpoint1
2. Web Role项目包含了配置信息，参数名为ConnectionString和Setting1
3. Endpoint1访问的协议是http，访问的端口是80

项目包含了Worker Role，计算节点大小为Medium (A2, 2Core / 3.5GB)

1. Worker Role只包含了配置信息，参数名为ConnectionString

### 查看ServiceConfiguration.Cloud.cscfg(非常重要)

我们通过Visual Studio 2013双击打开ServiceConfiguration.Cloud.cscfg文件，如下图：



在 ServiceConfiguration节点，可以看到如下参数:



上图中的参数含义是:

1. osFamily=”4”。表示Cloud Service后台的操作系统是Windows Server 2012 R2。

所以Azure Cloud Service背后运行的操作系统，一定是Windows Server 2012 R2。

有关osFamily的含义请参考MSDN文档：

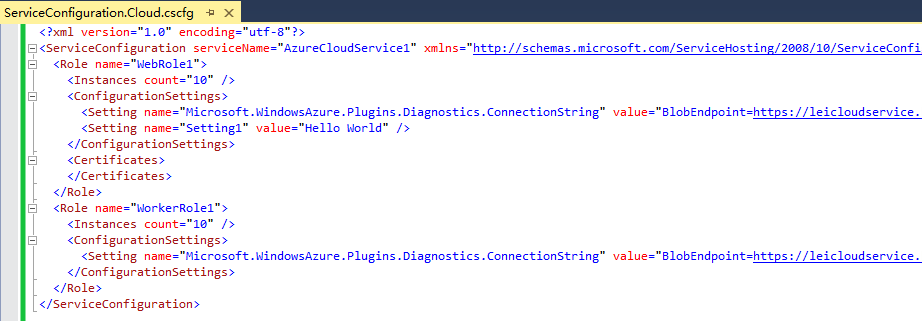
<https://msdn.microsoft.com/zh-cn/library/azure/ee758710.aspx>

1. osVersion=”\*”。表示由微软云Azure来帮助用户升级操作系统补丁，并且保证补丁最新。
2. 如果用户需要手动维护操作系统补丁，请将osVersion手动进行设置。

请参考MSDN文档：

<https://azure.microsoft.com/zh-cn/documentation/articles/cloud-services-guestos-update-matrix/>

我们还可以查看ServiceConfiguration.Cloud.cscfg，如下图：



上图中，我们定义了Web Role信息

1. 项目文件包含了WebRole1，实例数量为10个
2. 定义了参数名ConnectionString和参数值
3. 定义了参数名Setting1和参数值Hello World

我们还定义了Worker Role信息

1. 项目文件包含了WorkerRole1，实例数量为10个
2. 定义了参数名ConnectionString和参数值

## 使用模拟器调试(非常重要)

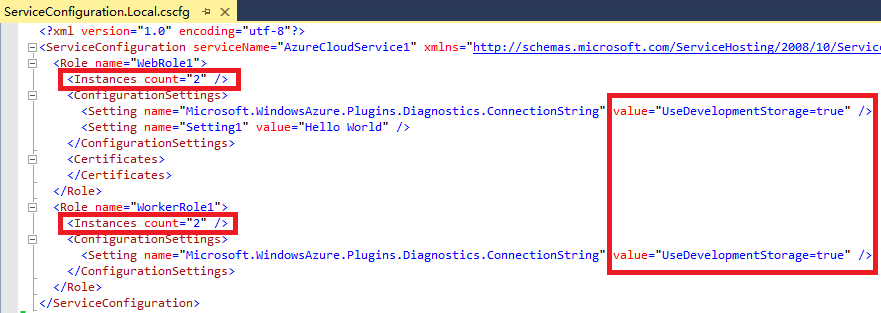
### 修改配置

注意：

在使用本地模拟器调试的时候，请配置文件ServiceConfiguration.Local.cscfg修改：

1. 把WebRole1和Worker Role1的Instance Count都修改为2。上几节内容我们把Instance Count修改为10，但是在本地调试的话会耗费过多的资源。
2. 将ConnectionString配置信息修改为UseDevelopmentStorage=true

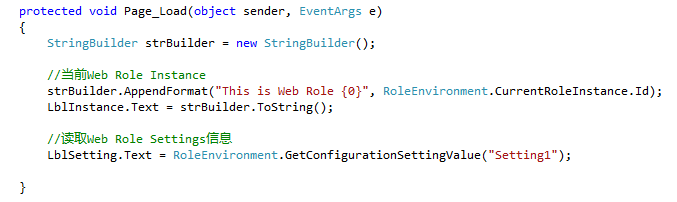
如下图红色部分所示：



### 修改代码(非常重要)

同时我们还要修改以下内容：

1. 在WebRole1的Default.aspx页面，增加2个Label：LblInstance和LblSetting
2. 在Default.aspx.cs的Page\_Load方法，增加以下代码：



上面的代码有2点非常重要的地方：

1. lblInstance表示是由哪个Web Role Instance响应客户端请求的
2. LblSetting的值，是读取Web Role Settings1参数值

关于读取Web Role Setting1参数值：

1. 一般情况下，我们把ASP.NET Web Form的配置信息保存到Web.config文件里。但是如果多台Web Role在横向扩展的时候，我们需要手工一台一台更新每个Web.config文件，费时费力。
2. 读取Web.config的.NET API是：ConfigurationManager.AppSettings
3. 笔者建议对于Web Role的配置信息，以前是保存到Web.config文件里。现在可以保存到Web Role的Setting页面里。这样修改了配置文件，可以在多台的Web Role实例同时生效。我会在后面有DEMO演示
4. 读取ServiceConfiguration.Cloud.cscfg配置文件的.NET API是

RoleEnvironment.GetConfigurationSettingValue

1. 修改WebRole1的WebRole.cs，增加以下代码：



上面的代码表示Configuration的变更和通知机制，我会在后面几章做详细介绍。

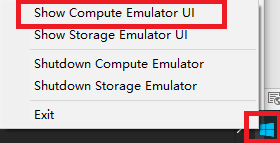
### 观察模拟器

现在我们可以通过本地的Azure Storage Emulator和Azure Storage Emulator，来进行调试。

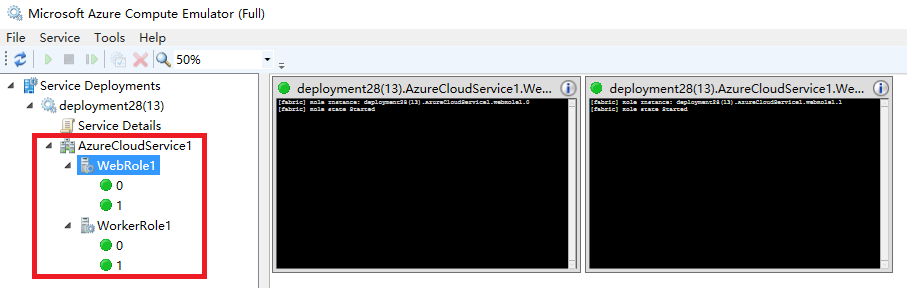
1. 我们点击Visual Studio 2013，启动项目
2. Azure Compute Emulator和Azure Storage Emulator会启动，在Windows右下角会出现模拟器图标，如下图：



1. 我们点击上图的蓝色图标，点击右键

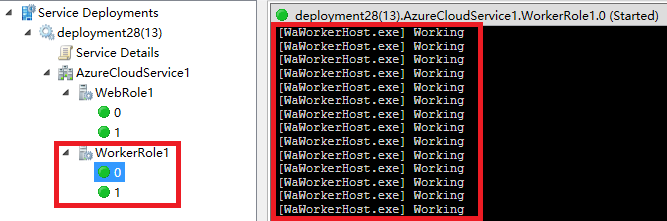


点击上图中的Show Compute Emulator UI，可以查看到Azure Cloud Service有2个Web Role和2个Worker Role，如下图：

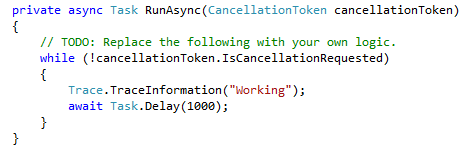


上图的Instance Count就是我们在4.10.1中，修改的Instance count=2，Web Role和Worker Role分为是2个实例。

1. 我们点击Worker Role的绿色图标，如下图：



可以查看到Work Role的打印输出，这个输出其实就是我们在WorkerRole1的WorkerRole.cs中RunAsync()的输出：

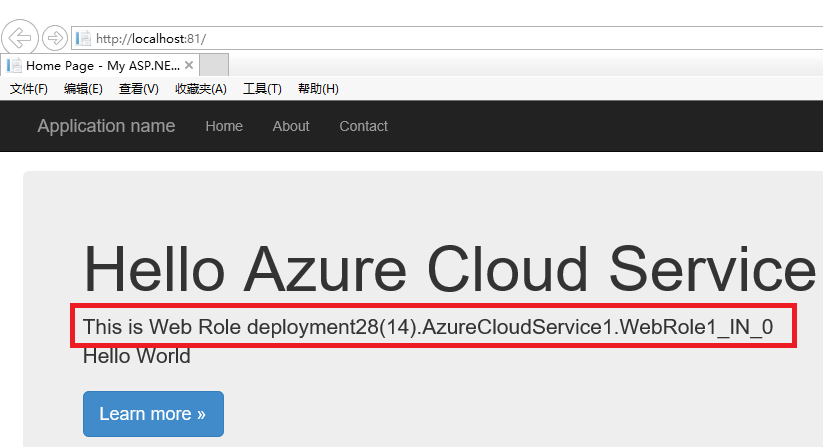


上面的代码片段里await Task.Delay(1000)表示每1000毫秒(1秒)执行打印输出。

我们发现其实Worker Role就像是Windows Service一样，没有界面，在后台默默地执行。当然你也可以改变Task.Delay(1000)，修改每次执行的时间。

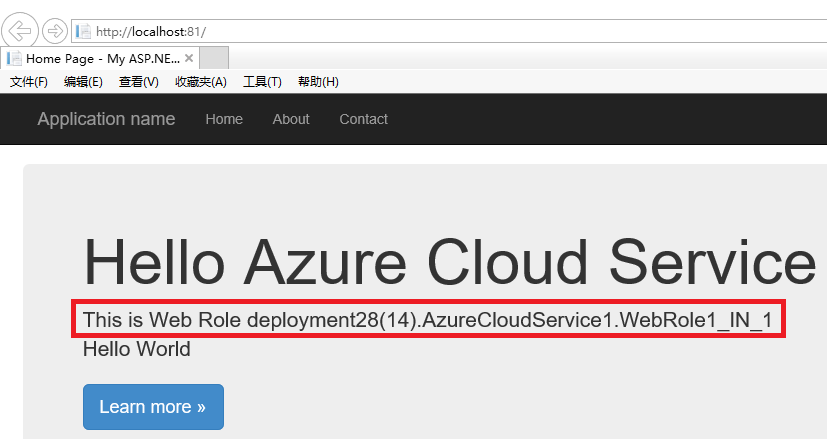
### 代码运行结果(非常重要)

在启动模拟器的时候，IE会打开Web Role的页面内容，如下图：



上图中，我们可以看到红色的部门。当前的客户端请求，是由服务器端的WebRole1\_IN\_0实例来处理的。

我们不停的按IE的F5，对当前页面刷新。会观察到如下图所示：



上图中，我们又观察到红色的部分。客户端的请求，是由服务器端的WebRole1\_IN\_1实例来处理的。

所以默认情况下，Azure Cloud Service是绝对负载均衡。无法保留Session(会话)的。

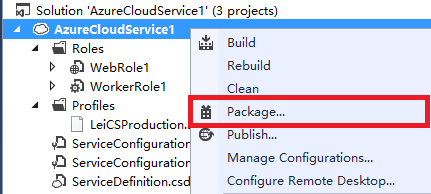
## Azure Cloud Service打包(非常重要)

上面一节中，笔者介绍了如果通过本地Azure Emulator模拟器，模拟Azure Cloud Service运行情况。

接下来，我们需要把代码通过Visual Studio打包，观察打包的情况。

### Cloud Service打包

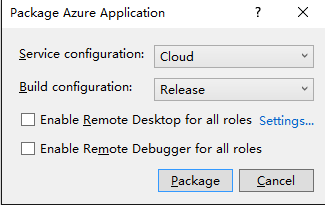
1. 我们点击Azure Cloud Service，右键Package，如下图：



1. 在弹出的窗口中，Service Configuration选择Cloud，

即使用ServiceConfiguration.Cloud.cscfg的配置信息

Build Configuration选择Release。如下图:



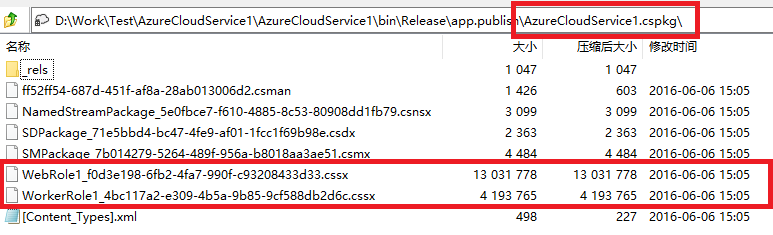
1. 点击上图中的Package，Visual Studio会对整个Azure Cloud Project的Web Role和Worker Role进行打包。打包结束后，会弹出新的文件夹窗口，如下图：



1. 上图中，会有2个项目文件cspkg和cscfg。

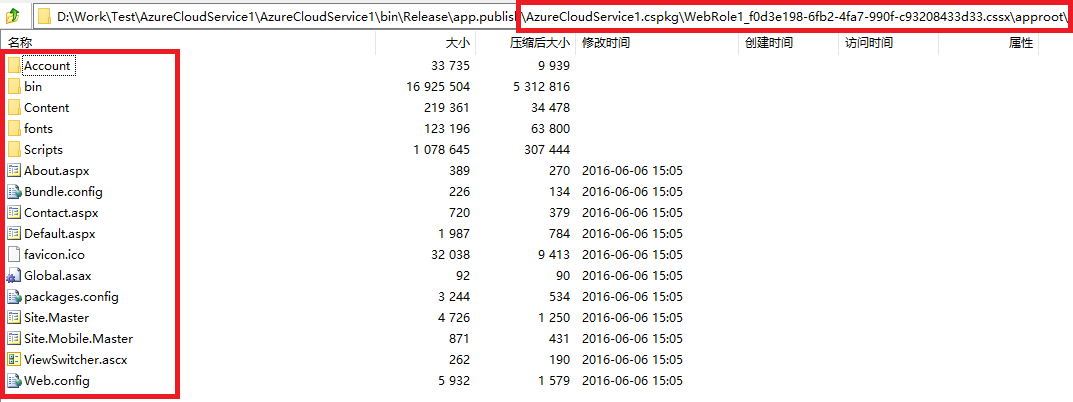
### CSPKG

1. cspkg全称是Cloud Service Package，即Cloud Service中Web Role和Worker Role，进行压缩后打包的项目。我们可以用7Zip打开cspkg文件。如下图：



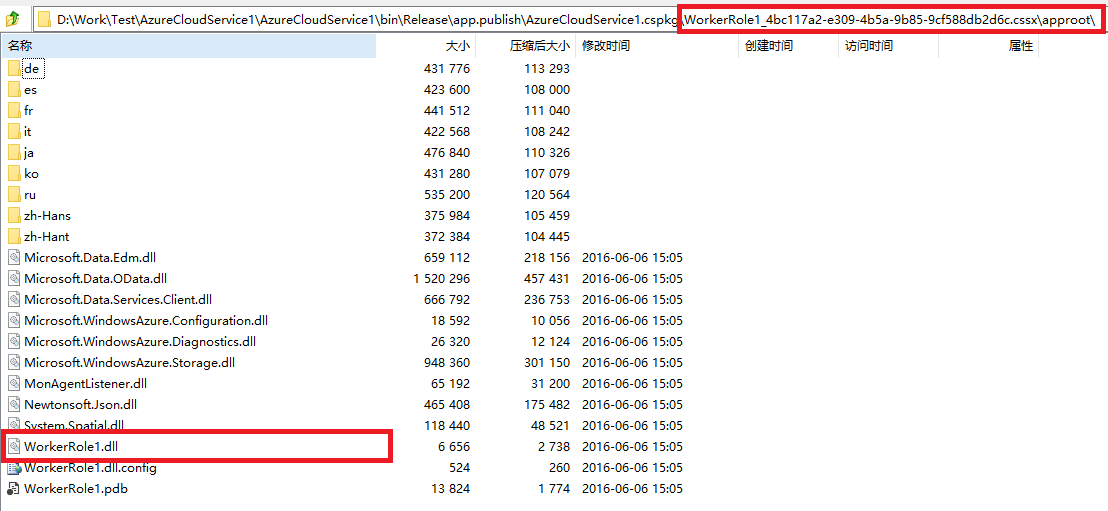
可以看到在这个cspkg文件中，包含了很多项目文件。而且我们从文件名中可以看到WebRole和WorkerRole。如上图红色部分。

1. 我们再用7Zip打开上图的Web Role文件里的approot文件夹。显示结果如下图：



可以看到上图中包含了About.aspx，Default.aspx, Global.asax还有Web.config文件。这个文件夹包含的就是我们在Visual Studio中，Web Role项目的源代码。

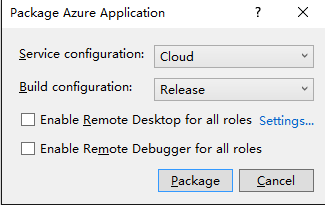
1. 我们再用7Zip，打开WorkerRole文件夹里的approot文件夹。显示结果如下图：



可以看到，上图中的WorkerRole1.dll，就是我们在Visual Studio中，Worker Role项目的编译后的类库。

### CSCFG

我们在Cloud Service打包的时候，Service Configuration选择Cloud，即使用ServiceConfiguration.Cloud.cscfg的配置信息。如下图:



笔者在这里就不再详细说明了。

## Azure Cloud Service发布

在之前几章中，笔者介绍了如何通过模拟器，在本机调试Cloud Service。现在笔者将介绍，如何将Azure Cloud Service发布到Azure云端。

因为国内由世纪互联运维的Azure China，Cloud Service的服务端点(Service Endpoint)和海外的Azure不同，所以笔者在这里详细介绍一下。

### 使用Azure PowerShell下载发布凭据

通过Visual Studio发布Cloud Service，需要借助Azure PowerShell下载发布凭据(publish certification)。

1. 首先我们下载Windows Azure PowerShell，下载地址：

<https://www.azure.cn/downloads/>。我们选择命令行工具，点击安装。如下图：



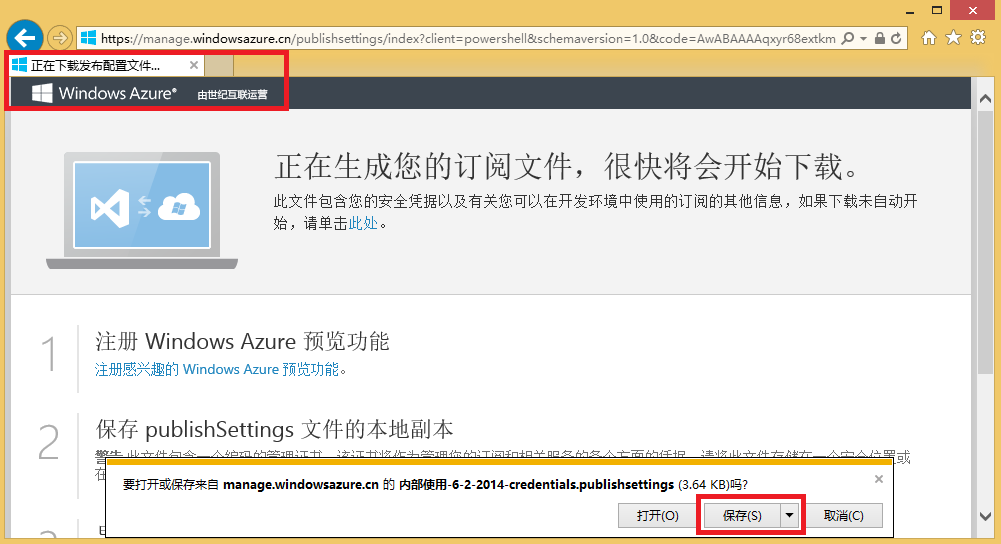
1. 我们以管理员身份，运行Azure PowerShell，执行以下命令:

Get-AzurePublishSettingsFile **-Environment AzureChinaCloud**

请注意上图中红色部分的命令，是与Azure Global不同，是专为Azure China的PowerShell做的特殊的参数。

输入命令后，计算机会弹出新的IE窗口，导航至Azure China网站，并要求我们输入Org ID和密码进行登陆。

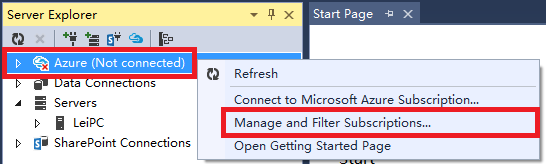
1. 登陆完毕后，系统会要求我们保存扩展名为publishsettings的文件，我们保存至本地计算机的D盘上。如下图:



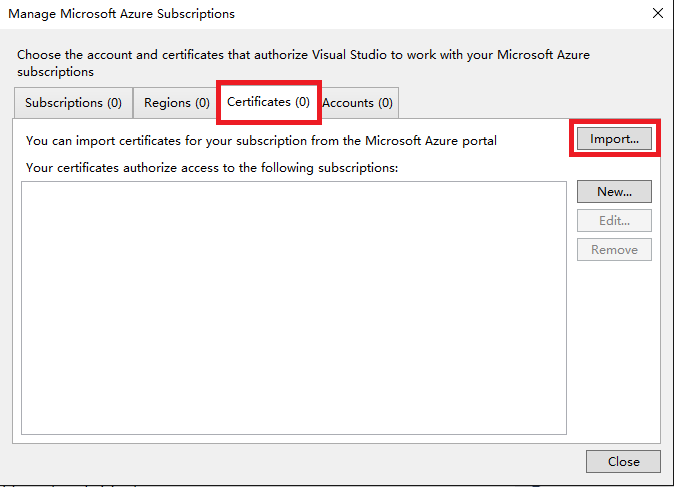
这个publishsettings的文件，就是我们使用Visual Studio发布Cloud Service需要的发布凭据。

### 使用Visual Studio导入发布凭据

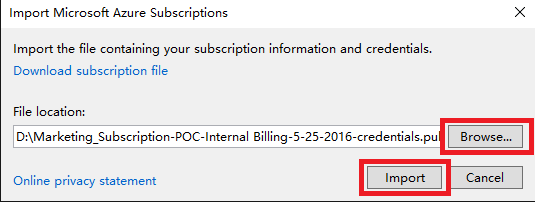
1. 我们回到Visual Studio 2013，点击Server Explorer，选择Azure，右键，选择Manage and Filter Subscriptions。如下图：



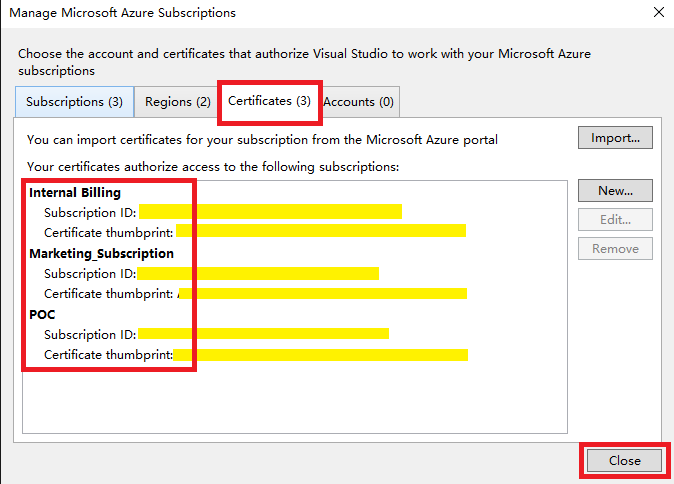
1. 在弹出的窗口中，选择Certificates，点击Import，如下图：



1. 导入我们在上一节中下载的publishsettings文件，如下图：

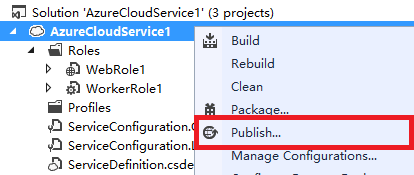


1. 导入成功以后，我们就可以查看到对应的订阅信息，如下图：

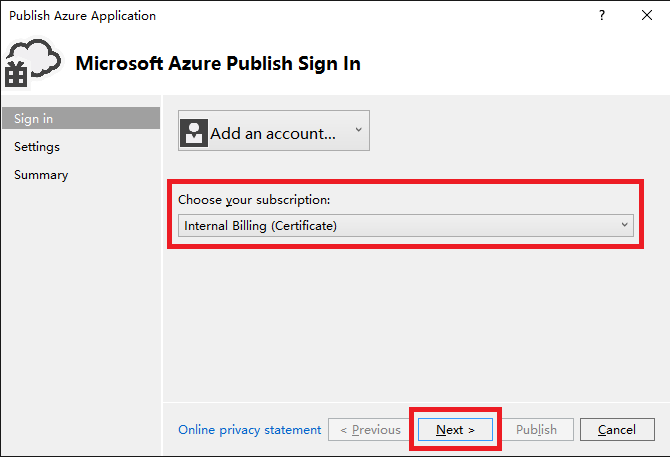


### 发布Cloud Service并配置远程桌面连接

1. 我们回到Visual Studio，点击项目文件Azure Cloud Service，右键，点击Publish，如下图：



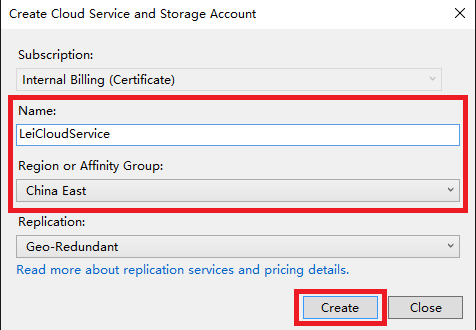
1. 首先，我们选择相应的订阅。如下图：



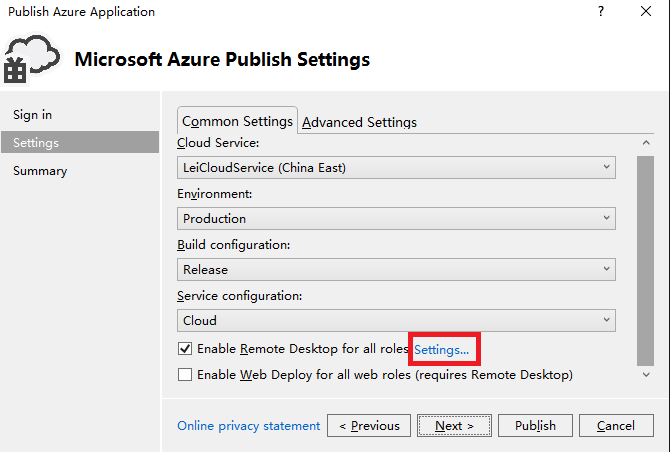
1. 因为是第一次发布，我们需要指定这个Cloud Service的DNS Name和所在的数据中心。

如下图所示，我们指定DNS Name为Lei Cloud Service

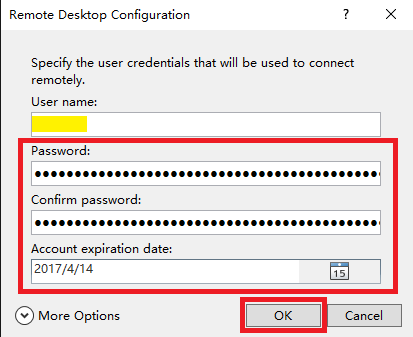
所在数据中心，我们选择China East中国东部。最后点击Create



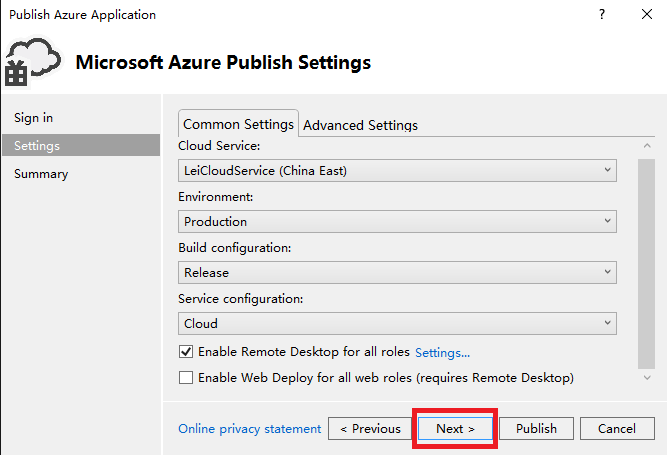
1. 上图Create完毕后，进入下一步。我们还可以配置远程桌面连接。我们点击下图的Settings。



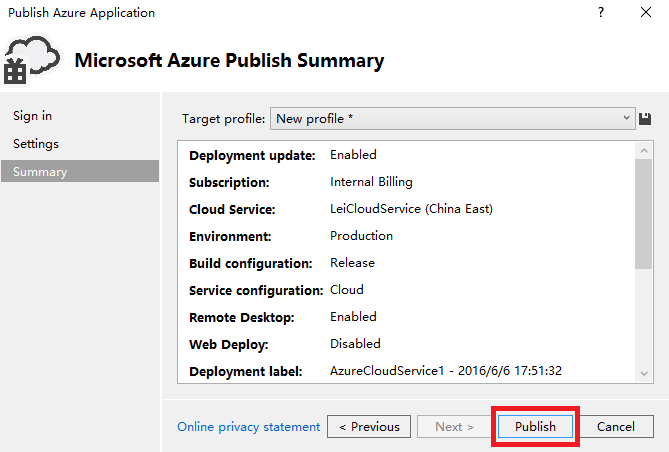
1. 在弹出的窗口中，输入远程桌面连接的用户，密码，确认密码和RDP账户过期时间。如下图：



1. 点击上图的OK后，弹出窗口关闭。然后我们点击下图的Next按钮



1. 最后我们点击Publish，直接将Azure Cloud Service发布到Azure云端

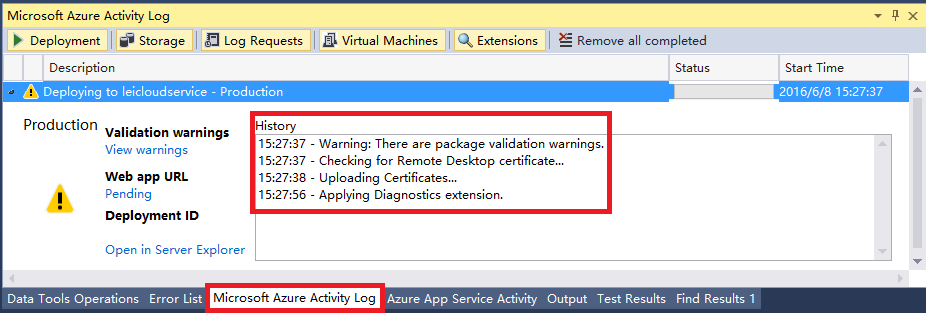


### 观察发布过程和发布结果

#### 发布过程

我们点击上一节图片中的Publish，发布Azure Cloud Service，具体发布的过程我们在下一节中会详细的介绍。本节只需要看发布过程和发布结果。

点击发布以后，Visual Studio会把我们的Cloud Project中的Web Role和Worker Role发布到Azure云端，如下图：



上图红色区域的发布输出，如下：

15:27:37 - Warning: There are package validation warnings.

15:27:37 - Checking for Remote Desktop certificate...

15:27:38 - Uploading Certificates...

15:27:56 - Applying Diagnostics extension.

15:28:35 - Preparing deployment for AzureCloudService1 - 2016/6/8 15:27:12 with Subscription ID 'e2eaa986-29d9-48c9-8302-1e2900a4504b' using Service Management URL 'https://management.core.chinacloudapi.cn/'...

15:28:35 - Connecting...

15:28:35 - Verifying storage account 'leicloudservice'...

15:28:37 - Uploading Package...

15:28:45 - Creating...

15:29:21 - Created Deployment ID: f9f99ee5f486449995afc75178d8faa1.

15:29:21 - Instance 0 of role WebRole1 is stopped

15:29:21 - Instance 1 of role WebRole1 is creating the virtual machine

15:29:21 - Instance 0 of role WorkerRole1 is creating the virtual machine

15:29:21 - Instance 1 of role WorkerRole1 is creating the virtual machine

15:29:21 - Starting...

15:29:42 - Initializing...

15:29:42 - Instance 0 of role WebRole1 is creating the virtual machine

15:29:42 - Instance 1 of role WebRole1 is starting the virtual machine

15:29:42 - Instance 0 of role WorkerRole1 is starting the virtual machine

15:29:42 - Instance 1 of role WorkerRole1 is starting the virtual machine

15:30:14 - Instance 0 of role WebRole1 is starting the virtual machine

15:31:48 - Instance 0 of role WebRole1 is in an unknown state

15:31:48 - Instance 1 of role WebRole1 is busy

Details: Preparing to start role... System is initializing. [2016-06-08T07:31:35Z]

15:31:48 - Instance 0 of role WorkerRole1 is in an unknown state

15:31:48 - Instance 1 of role WorkerRole1 is busy

Details: Preparing to start role... System is initializing. [2016-06-08T07:31:40Z]

15:32:19 - Instance 0 of role WebRole1 is busy

Details: Initializing role... System is initializing. [2016-06-08T07:31:45Z]

15:32:19 - Instance 1 of role WebRole1 is busy

Details: Initializing role... System is initializing. [2016-06-08T07:31:35Z]

15:32:19 - Instance 0 of role WorkerRole1 is busy

Details: Initializing role... System is initializing. [2016-06-08T07:31:43Z]

15:32:19 - Instance 1 of role WorkerRole1 is busy

Details: Initializing role... System is initializing. [2016-06-08T07:31:40Z]

15:33:53 - Instance 0 of role WebRole1 is busy

Details: Bringing role online... Calling OnRoleRun. [2016-06-08T07:33:40Z]

15:33:53 - Instance 1 of role WebRole1 is busy

Details: Bringing role online... Calling OnRoleRun. [2016-06-08T07:33:31Z]

15:33:53 - Instance 0 of role WorkerRole1 is busy

Details: Bringing role online... Calling OnRoleRun. [2016-06-08T07:33:46Z]

15:33:53 - Instance 1 of role WorkerRole1 is busy

Details: Waiting for role to start... Calling OnRoleRun. [2016-06-08T07:33:48Z]

15:35:58 - Instance 0 of role WebRole1 is ready

15:35:58 - Instance 1 of role WebRole1 is ready

15:35:58 - Instance 0 of role WorkerRole1 is ready

15:35:58 - Instance 1 of role WorkerRole1 is ready

15:35:58 - Created web app URL: <http://leicloudservice.chinacloudapp.cn/>

15:35:58 - Complete.

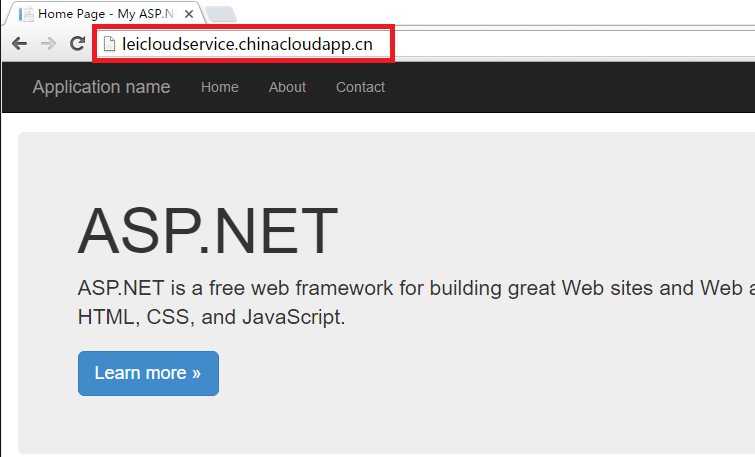
#### 发布结果

1. 我们登陆Azure Management Portal <https://manage.windowsazure.cn>
2. 点击云服务(Cloud Service)，点击我们之前设置的DNS Name：LeiCloudService
3. 页面跳转，点击，实例，生产。如下图：



1. 我们可以看到Web Role有2个实例，每个实例的大小为A2
2. 2个Web Role在不同的更新域(Upgrade Domain)和故障域(Fault Domain)
3. Worker Role有2个实例，每个实例的大小为A1
4. 2个Worker Role在不同的更新域(Upgrade Domain)和故障域(Fault Domain)
5. 因为我们在Visual Studio Cloud Service项目中，分别配置了Web Role和Worker Role横向扩展的数量，和实例大小
6. 微软云Cloud Service自动把我的Visual Studio项目中，Web Role和Worker Role的源代码打包，分发到若干个Instance实例上，并且自动设置了高可用

我们可以打开浏览器，查看发布结果：<http://leicloudservice.chinacloudapp.cn/>



这个页面，就是我们Web Role项目的Default.aspx页面，显示的是Web Form的Template。

### Cloud Service部署和运行过程(非常重要)

本节我们会对Azure Cloud Service Role的生命周期进行进一步的了解。首先，让我们走到幕后看看一个Role Instance是怎么被发布到虚拟机上并启动起来的。

#### Role在虚拟机上部署和运行的过程

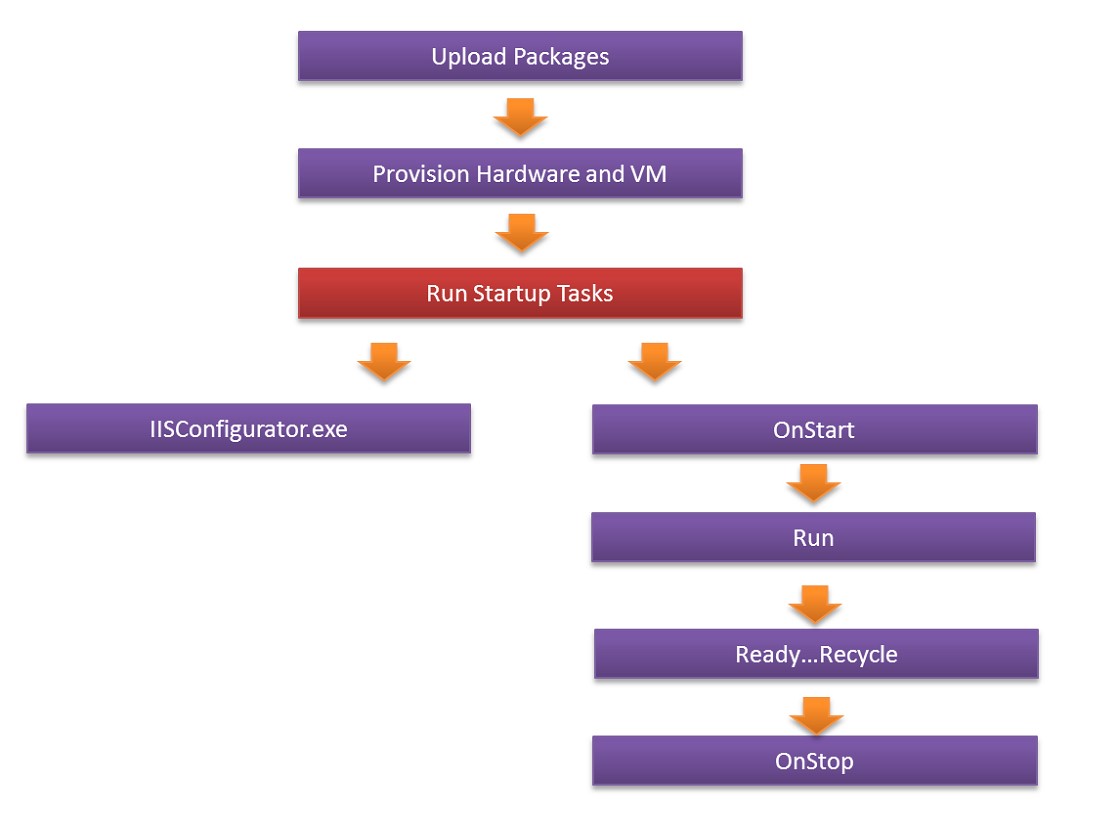
以下是角色实例(Role Instance)发布和启动的一个简要过程。

1. Windows Azure在服务器池中选择一个有足够CPU内核数的宿主服务器，或者启动一个新的满足需求的宿主服务器
2. Windows Azure将云服务包(CSPKG)和配置文件(CSCFG)复制到宿主服务器上。宿主服务器上的一个宿主代理程序(Host Agent)会启动虚拟操作系统。
3. 在虚拟操作系统上有一个名为WaAppAgent的代理程序。这个代理程序负责配置虚拟操作系统并启动一个名为WaHostBootstrapper的进程。
4. 如果角色上定义了启动任务，WaHostBootstrapper会运行这些启动任务并等待所有标记为Simple类型的启动任务正确执行
5. 如果角色是Web 角色(Web Role)，WaHostBootstrapper会启动IISConfigurator来配置IIS
6. 如果角色是Worker Role，WaHostBootstrapper启动一个名为WaWorkerHost的进程；如果角色是Web Role，WaHostBootstrapper启动一个名为WaIISHost的进程
7. 在上述进程中，载入角色的程序集并搜索其实施的RoleEntryPoint子类
8. 调用OnStart()方法
9. 调用Run()方法。同时，该实例被标记为"Ready"并加入负载平衡器
10. 如果Run()方法退出，OnStop()方法被调用。WaWorkerHost/WaIISHost结束运行，实例重启
11. WaHostBootstrapper开始循环监测实例的运行状态

Web Role不一定需要实施RoleEntryPoint类，这是因为Web Role最终是部署在IIS上，是IIS在接受和转发用户的请求。

这也是为什么我们如果把WebRole.cs从web项目中删除不会影响到网站运行的原因。

但是，如果您的Web Role实施了RoleEntryPoint类，您要确保Run()方法不退出，否则实例会重启。



#### Role Instance的状态

Role Instance可以处于不同的状态：Busy或者Ready。

只有在Role处于Ready状态时，才参与负载平衡任务的分配。

如果您想在代码中临时改变Role Instance的状态，可以响应RoleEnvironment类所定义的StatusCheck(状态检查)事件，并且通过事件参数中RoleInstanceStatusCheckEventArgs的Setbusy()方法将实例标记为Busy。这样，负载均衡器就不会把任务分配到当前实例，直至下一次WaHostBootstrapper对实例状态进行检查。

如果在运行过程中您的Role Instance代码退出了Run()方法，或者抛出了未处理的异常(Unhandledexceptions)，WaHostBootstrapper会重新启动您的Role Instance。这种自动恢复的机制能够帮助提高系统的可用性。当然，在重新启动的过程中，这个实例在非Ready状态下是不能接受用户请求的。如果WaHostBootstrapper自身奔溃了或者虚拟机宕机了，Windows Azure会重启虚拟机以及Role Instance。如果宿主服务器宕机了，Windows Azure在若干恢复重试后将Role Instance重新部署到一台健康的服务器上。

在Windows Azure关闭Role Instance的过程中，会触发Stopping事件，并调用Role的OnStop()方法。您可以在这里实现实例关闭过程中需要进行的处理，例如释放Role使用的资源等。

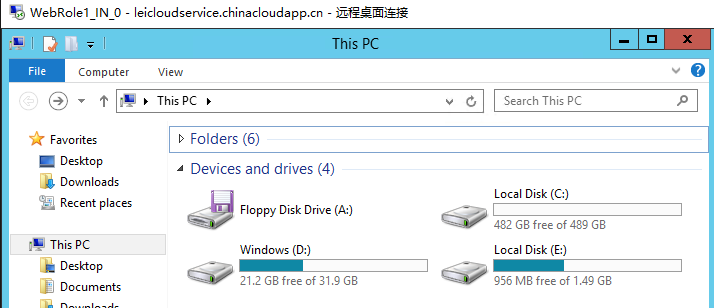
最后，如果想强制重新启动您的实例，可以用RoleEnvironment.RequestRecycle()方法来通知Windows Azure进行实例的重新启动。

### 远程连接Cloud Service

1. 我们点击云服务Cloud Service，页面跳转，选择相应的实例，点击连接



1. 在弹出的窗口，输入我们在Visual Studio Project配置的远程桌面连接的用户名和密码。我们就可以登陆到这台Cloud Service背后的虚拟机



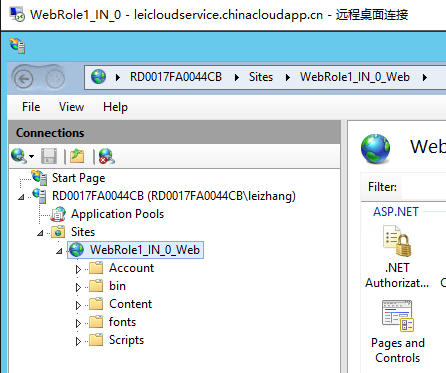
1. 我们可以观察到这个虚拟机与传统的Windows Server 2012虚拟机的磁盘结构不一样。

C盘是模板文件

D盘是Windows操作系统

E盘approot目录下，是项目文件

1. 我们通过远程桌面，运行inetmgr，打开IIS。运行结果如下：



1. 我们可以看到，这台特殊版本的Windows Server 2012已经帮我们配置好了IIS

## Azure Cloud Service是非持久化虚拟机(非常重要)

在本章内容，笔者花费了很大的篇幅介绍了Azure Cloud Service是非持久化虚拟机，所有的部署都是通过CSPKG和CSCFG文件来提交。

这个概念非常非常重要，笔者举一个例子，供大家参考和理解。

1. 假设我们开发一个会员系统，通过Azure Cloud Service的Web Role部署到微软云平台。
2. Web Role通过CSPKG和CSCFG部署到Azure Cloud Service
3. 第一次部署的时间为2016年1月1日，Web Role的ASP.NET项目，一共包含3个文件夹。
4. 项目部署完毕后，Internet用户开始使用这个会员系统，用户把头像、照片上传到云端虚拟机的本地磁盘，ASP.NET项目变为5个文件夹。**请注意，新增加的2个文件夹，并没有包含在CSPKG文件里。**
5. 2016年5月1日，Web Role所在的虚拟机宕机，Web Role通过CSPKG和CSCFG文件，漂移到同一个数据中心的其他节点上。
6. **请注意，因为新增加的2个文件夹，并没有包含在CSPKG文件里。所以在Web Role漂移以后，项目文件还原到2016年1月1日。**
7. **在2016年1月1日至2016年5月1日，这期间的增量文件，即用户的头像、照片等，都丢失了！！！**
8. 我们需要把不包含在CSPKG文件里的，应用系统运行期间产生的增量文件，请保存到Azure Storage云存储里。切记切记！

## 总结(非常重要)

1. Azure Cloud Service是非持久化虚拟机，应用系统运行期间产生的增量文件，请保存到Azure Storage存储账号里
2. Azure Cloud Service，可以同时包含Web Role和Worker Role，或者只包含两者其中的一种
3. Web Role响应前端用户请求
4. Worker Role在后端做任务处理
5. 我们在Visual Studio项目中，配置了Web Role的相关参数
   1. Configuration，配置了Instance Count，VM Size和Diagnostic
   2. Settings，配置连接字符串，功能类似于Web.config
   3. Endpoint，配置了打开的端口，和端口映射，类似Azure Virtual Machine的Public Port和Private Port
   4. Local Storage，使用默认的设置即可
   5. Certificates，设置HTTPS访问需要的证书
   6. Caching，使用默认的设置，如果客户想使用Cache缓存服务，请使用Azure Redis Cache。

参考资料：<http://www.cnblogs.com/threestone/category/741409.html>

1. 我们还配置了Worker Role的相关参数

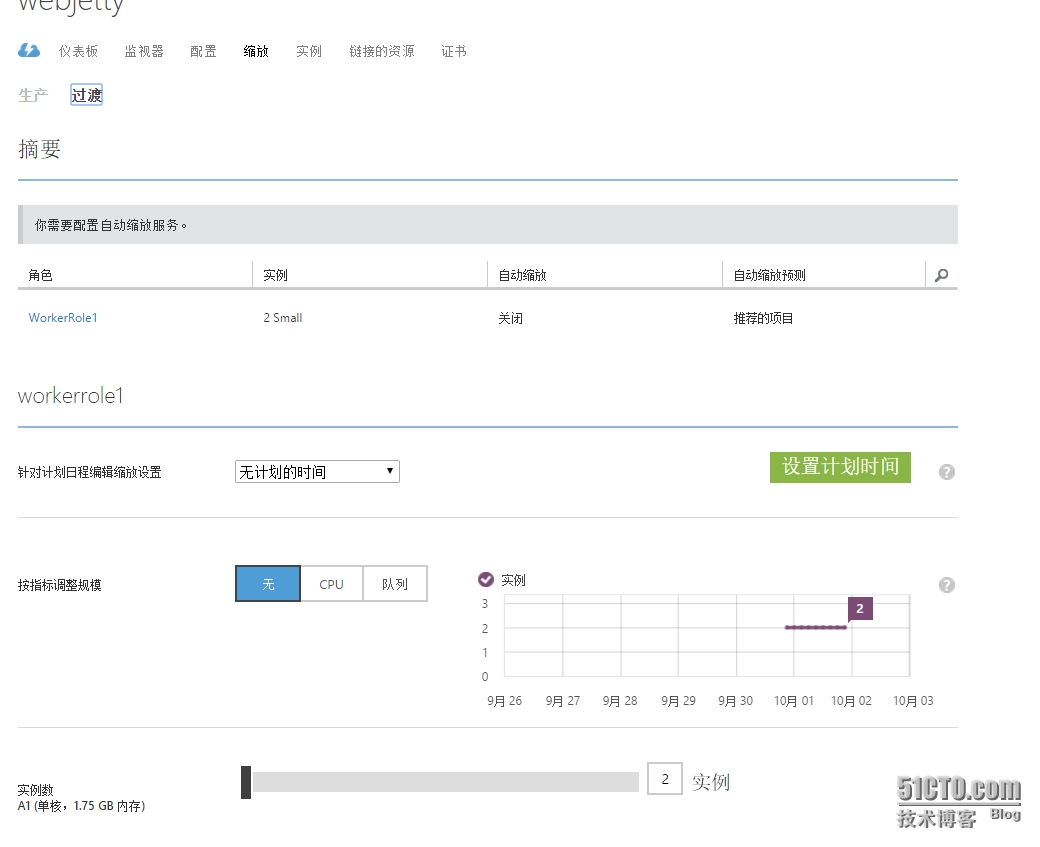
# Java开发者必读

## 第一部分

Microsoft Azure是一个开放的，灵活的云平台，除了对自家的.Net平台有良好的支持外，对于各种开源的软件，语言，工具，框架都有着良好的支持，比如Java，Php，Python等等，你可以使用自己喜欢的语言开发任何应用部署在Azure的web site或者云服务中。

[](http://s3.51cto.com/wyfs02/M00/74/02/wKioL1YNV7fzv4suAABsEtY1XTw035.jpg)

Azure的云服务是Azure的一个PAAS平台，同样支持多种不同的语言和框架，并且可以基于多种不同的阈值如CPU负载，队列，定时等等实现Auto scaling等高级功能，如下图所示：

[](http://s3.51cto.com/wyfs02/M01/74/02/wKioL1YNWbLiBMn4AAHuUo7STZY957.jpg)

本文简单介绍如何使用Azure所提供的Java的Eclipse插件，快速部署你的Java Web应用到Azure的Cloud Service，在你正式使用这项功能之前，希望你能了解以下一些限制，以便于你能更好的设计你的云端体系架构：

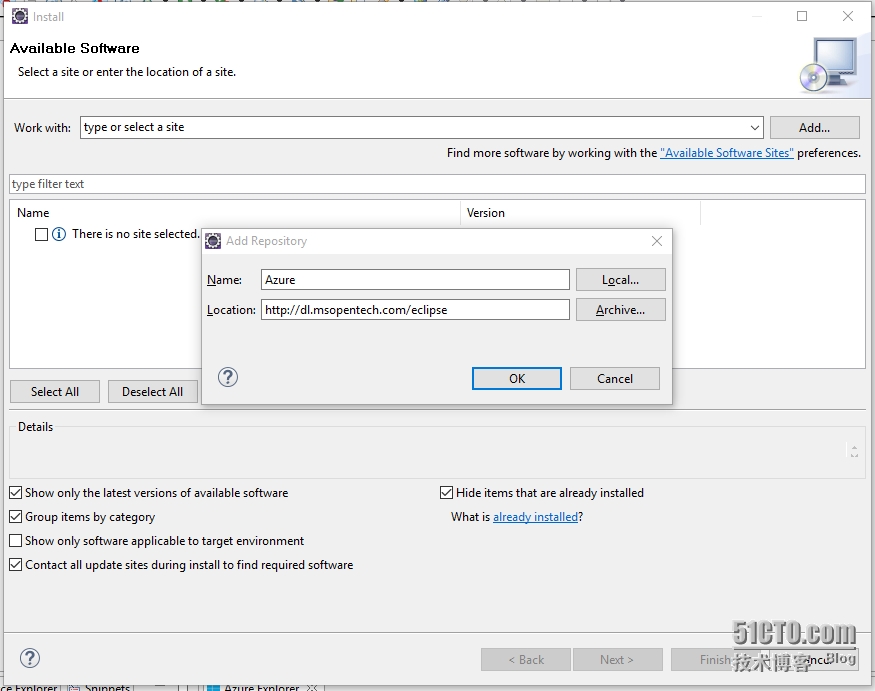
1. 目前Azure的cloud service底层的虚拟机是Windows 2008/2012，如果你有一些应用是跑在Linux上需要迁移到Azure Cloud Service，并且依赖于一些Linux的系统调用，那么你需要做写一些代码修改

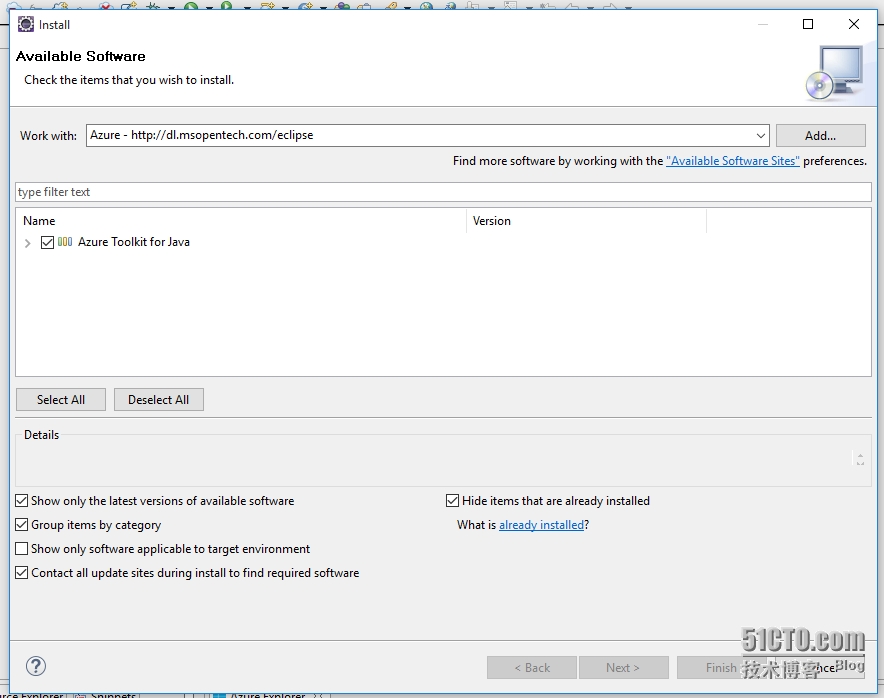
2. Java应用在Cloud Service目前只能是workrole，没有.Net中web role和work role的定义和机制，但对于各项云服务如队列，存储，数据库等的使用并不受限制，你可以通过开发实现

回到正文，我们开始我们的Java On Azure之旅，首先你需要安装JDK，Eclipse，也必须要有一个Azure的账号，这些基础的部分就不赘述了，关于Azure账号申请的部分请登录官方网页或者联系你的IT管理人员（如果你们公司已经购买了Azure服务）

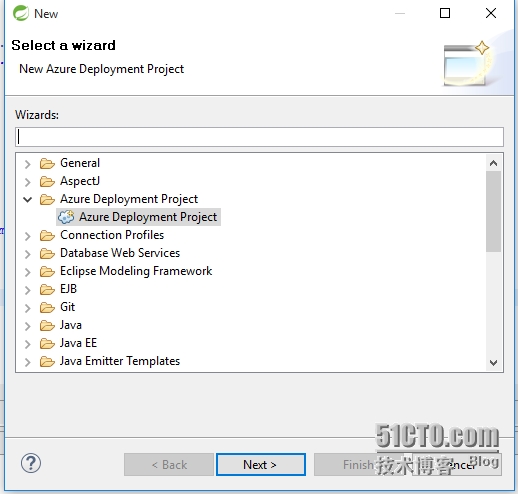
1. 打开Eclipse，安装Azure for Eclipse插件。你或许使用各个Eclipse的不同版本或发行版本，比如Spring Tool Suite， JBOSS IDE， Oracle IDE， IBM IDE等等，只要是Eclipse based，安装过程都大同小异。

2. 选择“Install New Software", 输入Azure插件的安装地址：http://dl.msopentech.com/eclipse，选择确定，在出现软件选择的时候选择全部安装，接受全部条款，然后一直next，知道安装完成，你的Eclipse会重启一次。

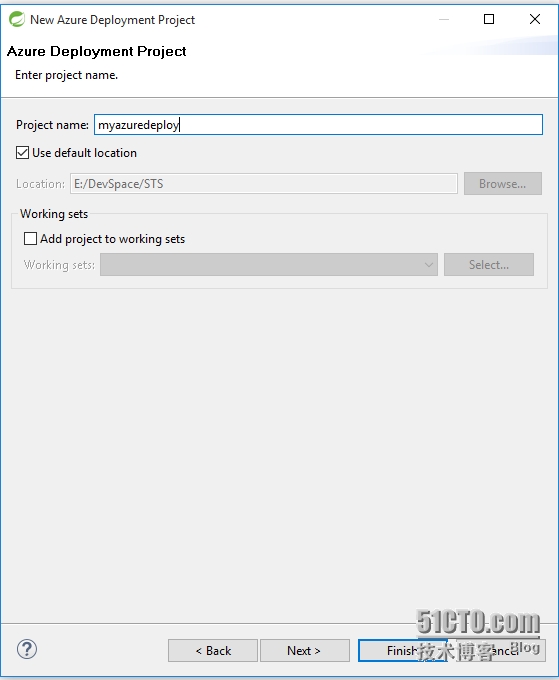
[](http://s3.51cto.com/wyfs02/M00/74/02/wKioL1YNXZ3gooRsAALuskElcWk234.jpg)

[](http://s3.51cto.com/wyfs02/M01/74/05/wKiom1YNXZTBP3gNAALBFsBomjU880.jpg)

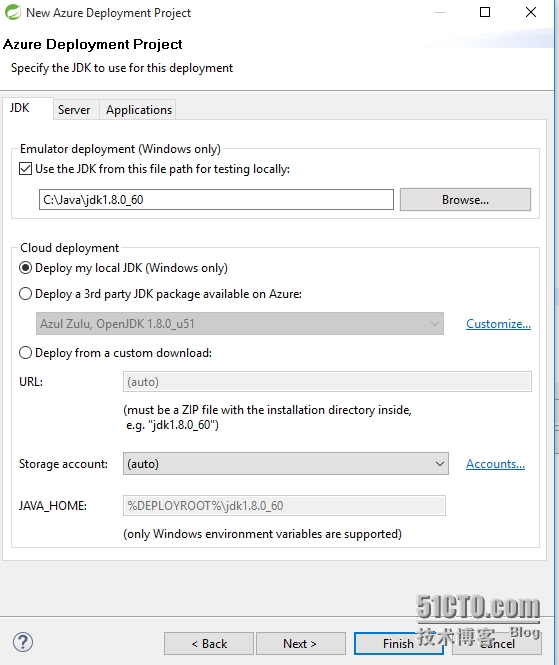
3. 完成安装后，我们就来创建我们第一个Azure Java的项目，因为本次主要是展示如何部署Java应用程序部署到Cloud Service，所以你需要准备一个一个你自己的WAR包，如果没有的话也ok，Azure自带了一个测试的helloworld.war,来进行简单的测试。打开Eclipse，选择New project，找到Azure deployment project，选择新建项目。

[](http://s3.51cto.com/wyfs02/M00/74/05/wKiom1YNYVGT5au4AAFsbl19CaM345.jpg)

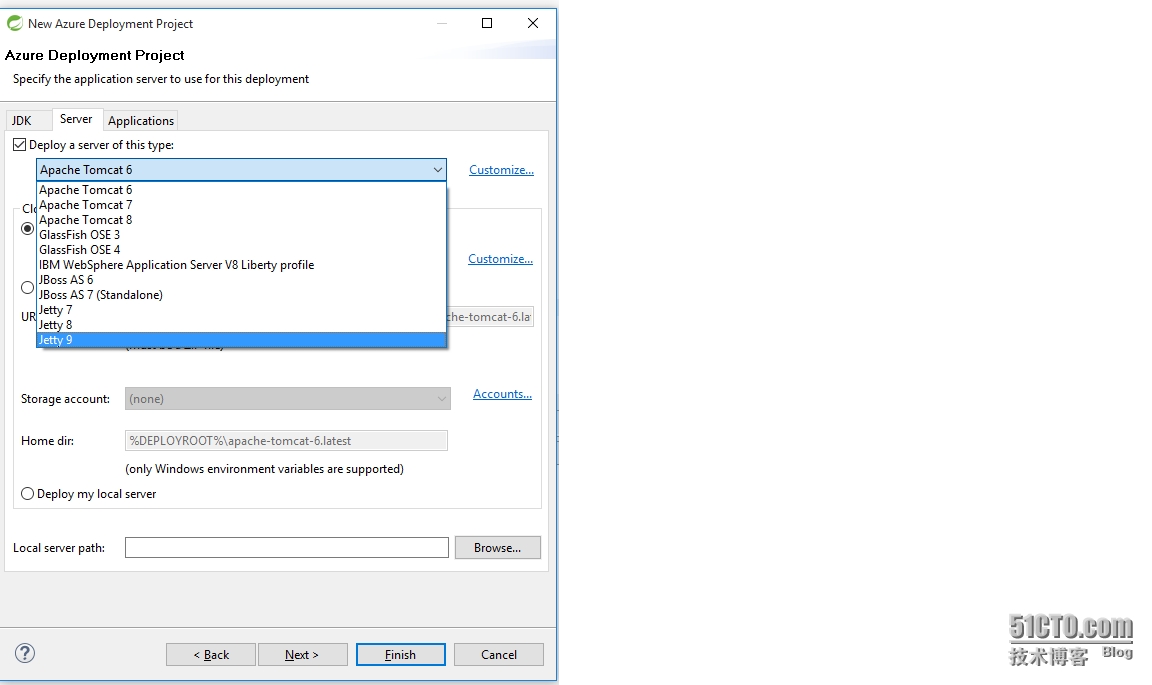
4.给你的项目写个名称，然后选择下一步：

[](http://s3.51cto.com/wyfs02/M00/74/02/wKioL1YNYV-yq0IPAAFLi_MWW7k393.jpg)

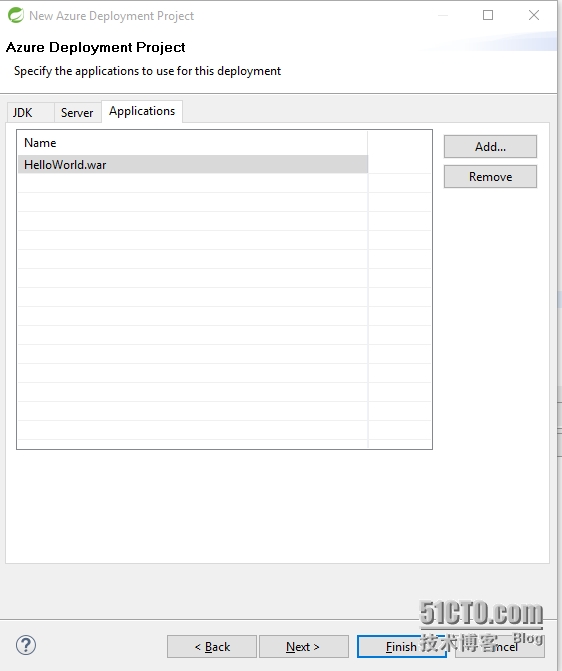
5.在这个页面，你需要指定你部署到云服务中的JDK，应用服务器，以及应用。那么我先来看一下JDK的部分，有几种不同的选项，第一种是你可以部署你本地的JDK到云端，比如Oracle的JDK 1.8，你可以部署第三方的，从云端直接下载的JDK，目前只支持OpenJDK，你可以制定一个远端的站点进行下载，但必须要注意的是，远端的JDK必须是zip包，因为不属实脚本只负责将JDK解开，在本示例中，我选择部署我本地的Oracle JDK 1.8.0-60到云端：

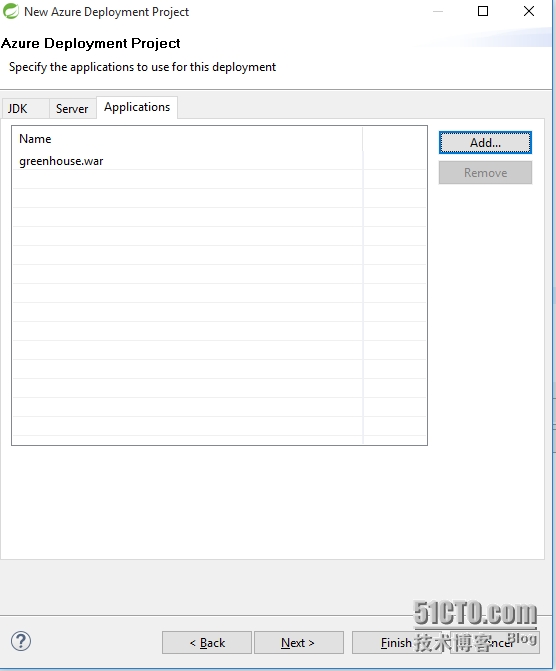
[](http://s3.51cto.com/wyfs02/M02/74/02/wKioL1YN3dWAp71jAAJfNokTQLY834.jpg)

6.第二个页面是需要选择你要部署的应用服务器，目前有多种服务器可供选择比，常见的Tomcat，Jetty，JBoss，GlassFish等都在列表中，本例中选择Jetty 9作为Java应用服务器；同样的，如果你有一些特殊的设置，你可以选择将本地的应用服务器上传到云端，只需要你指定本地服务器的路径即可。

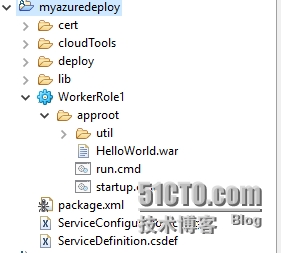
[](http://s3.51cto.com/wyfs02/M00/74/05/wKiom1YN3syzzUsjAAKmSQ_CbU8756.jpg)

7.最后一部分，是选择你要部署的Java Web应用程序，是一个标准的war包，默认情况下，Azure的应用程序会提供一个非常简单的HelloWorld的war包，他的基本功能就是输出经典的Hello Wolrd，如果你只想测试一下部署过程，可以选择该部署包，在本例中我会部署一个我的测试包Greenhouse.war：

[](http://s3.51cto.com/wyfs02/M00/74/02/wKioL1YN3-PCGNKJAAFP4-Op2NY440.jpg)

[](http://s3.51cto.com/wyfs02/M01/74/02/wKioL1YN3-OQ2gkKAAFM6AsCfN8725.jpg)

8.点击完成，创建新的项目，创建完成后，你会看到如下图所示的项目结构。cert目录会存放一些项目需要的证书，比如远程桌面连接的证书，cloudtools是一些发布和构建工具，deploy里面是打包完成需要发布的包，workrole1是一些启动运行脚本和示例HelloWorld包，另外三个文件是包的定义，云端服务配置文件和定义文件。

[](http://s3.51cto.com/wyfs02/M01/74/05/wKiom1YN4Qajg5oRAACx4C8Eoes648.jpg)

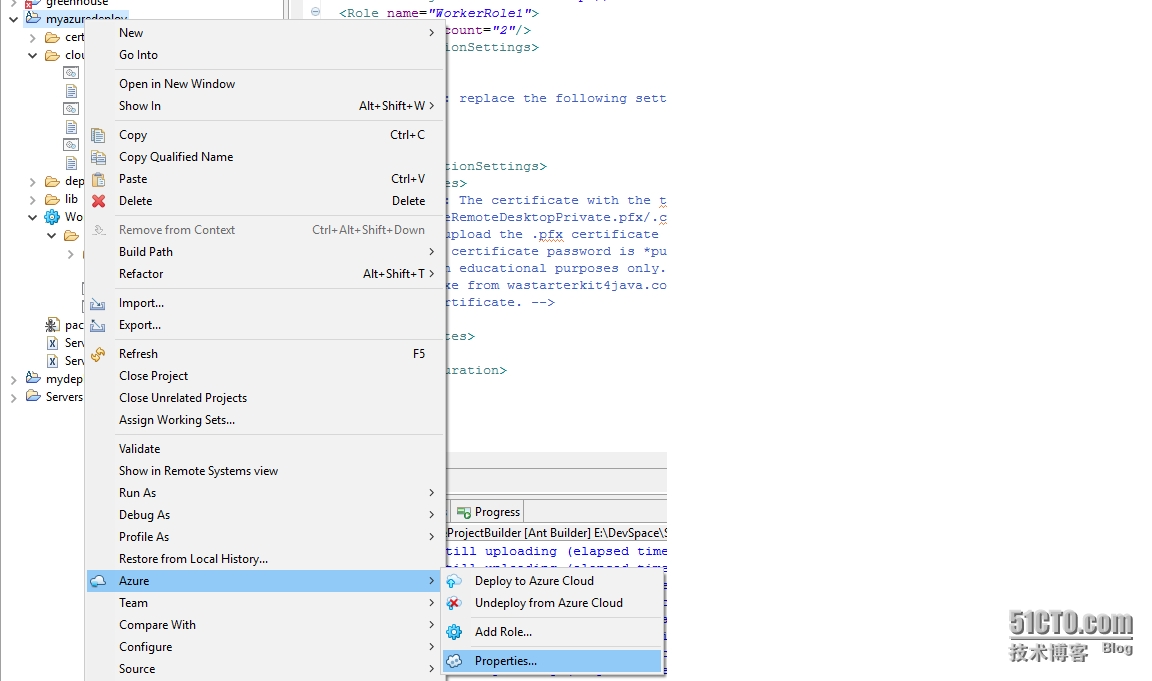
## 第二部分

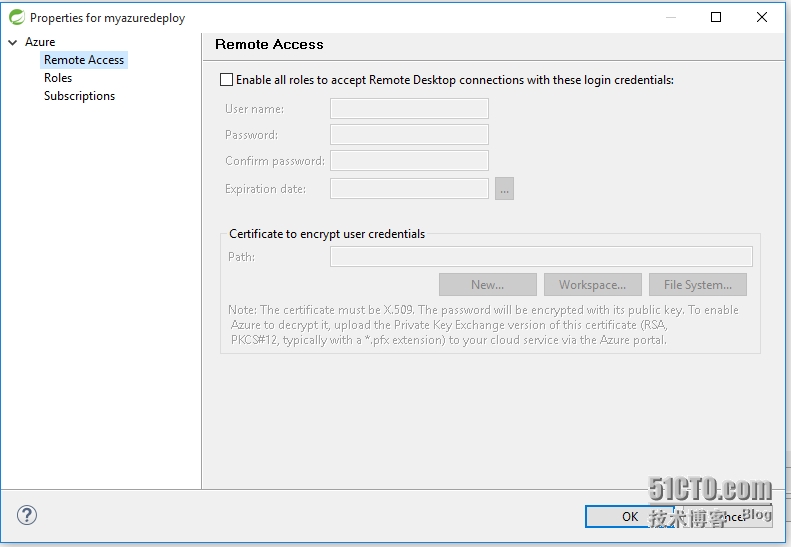
接上文

9.在进行发布之前，需要对我们的订阅做一些设置，因为默认情况下，Azure的service end指向的是Azure global的站点，如果我们要将服务发布在Azure的中国站点，需要做下简单的设置，在Eclipse中，打开偏好设置（preference），找到Azure，在service endpoint页面中，选择”windowsazure。才能（China)，选择确定：

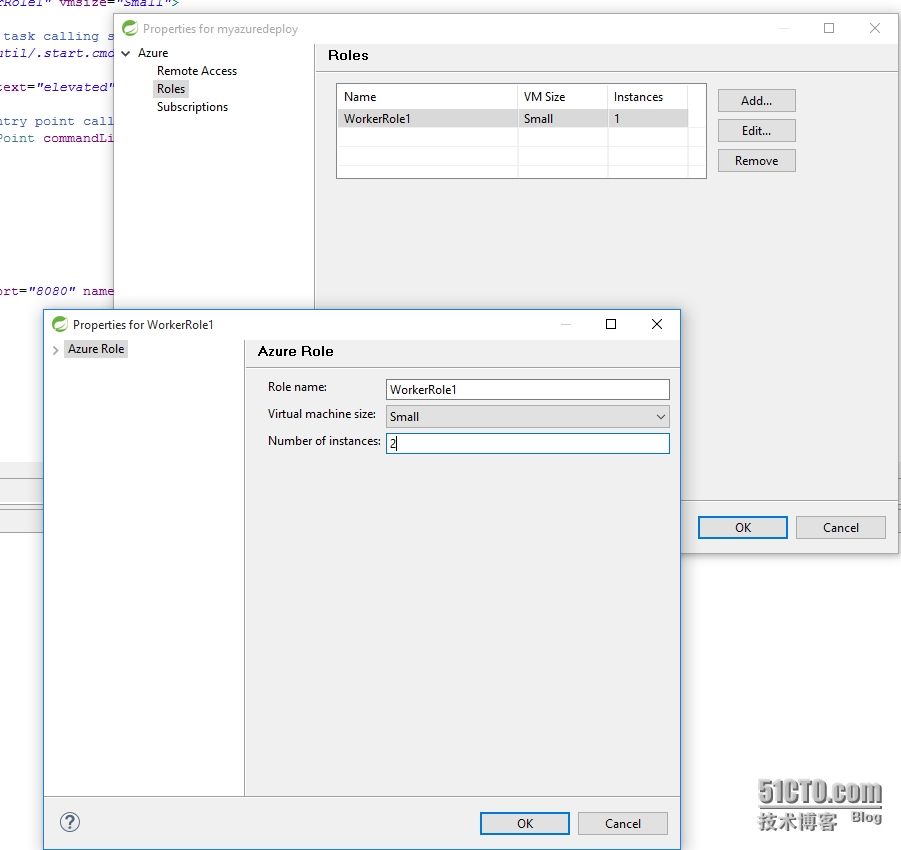
[](http://s3.51cto.com/wyfs02/M00/74/02/wKioL1YN4uqDO3rfAAJ4zHMjQes072.jpg)

10. 回到项目，选择myazuredeploy并单击右键，选择Azure，properties,第一项是选择是否配置远程访问，因为云服务底层实际上是Windows Server，所以本处实际是配置RDP访问，你可以可以在Azure portal直接配置，本例选择不配置

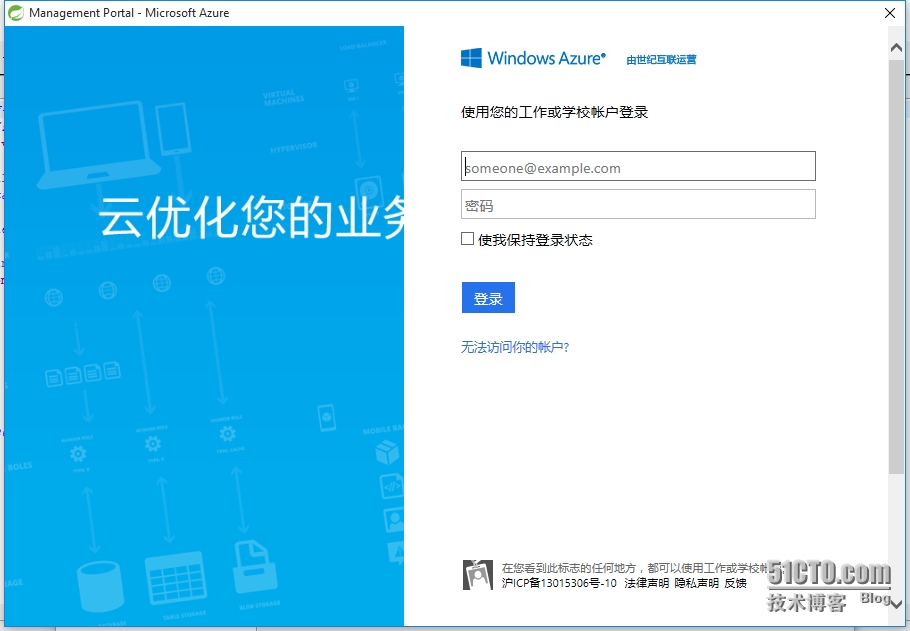
[](http://s3.51cto.com/wyfs02/M02/74/05/wKiom1YN5CKAyo86AAOhi5hBHlY706.jpg)

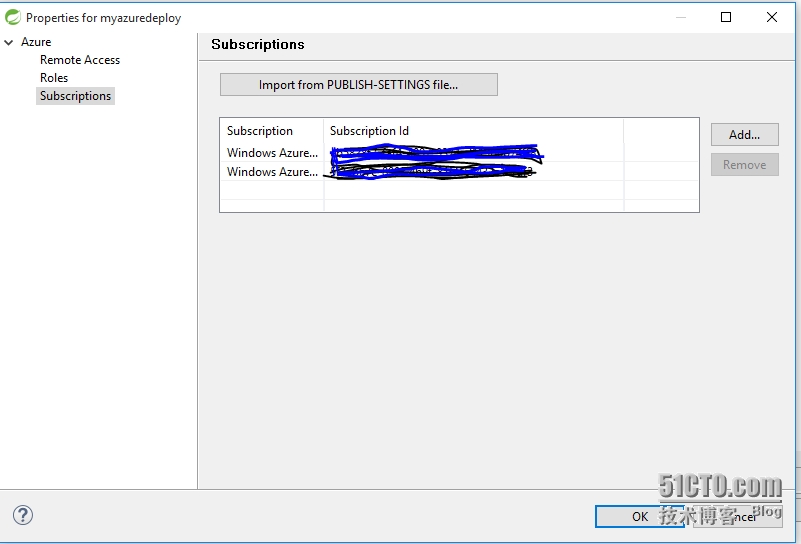
[](http://s3.51cto.com/wyfs02/M02/74/02/wKioL1YN5DDwM4SIAAHF2A31qkk760.jpg)

11. 第二项是role的定义，在本处你可以选择VM虚拟机的大小，以及你要在云服务中需要启动的实例个数，点击修改，修改实例个数为2，云服务中实例2个级以上才有SLA保障哦:)

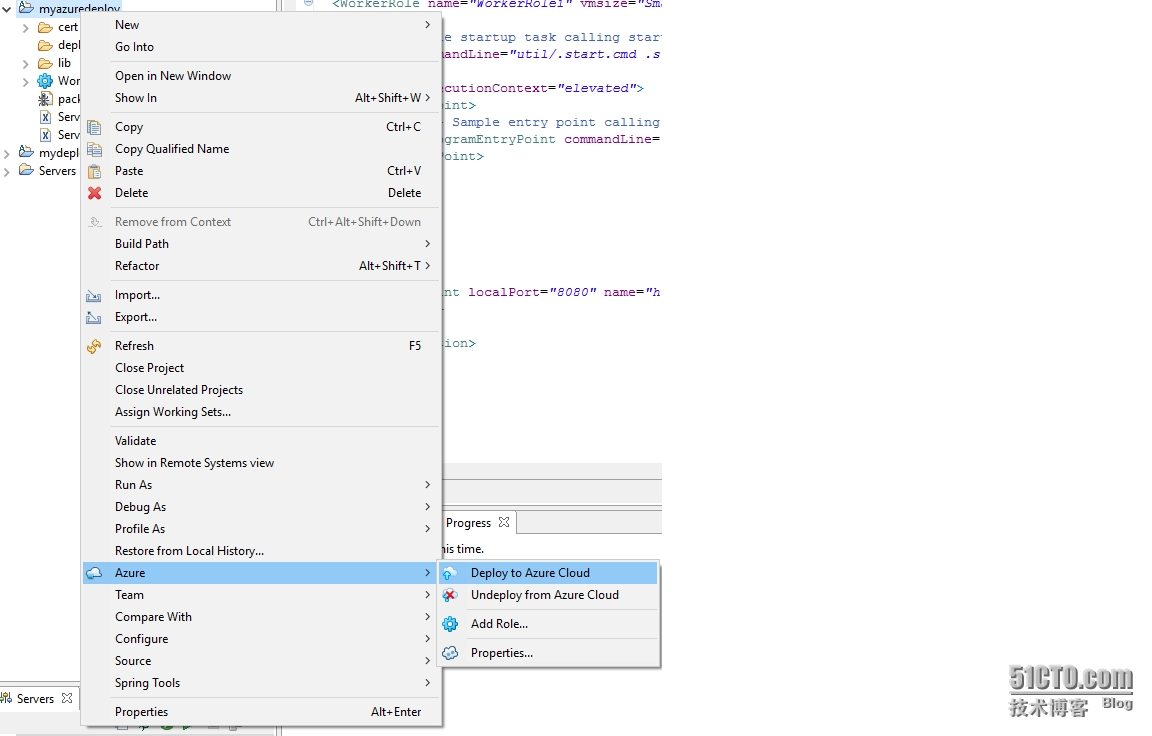
[](http://s3.51cto.com/wyfs02/M00/74/02/wKioL1YN5KnwthAJAAI7JAVK4B0495.jpg)

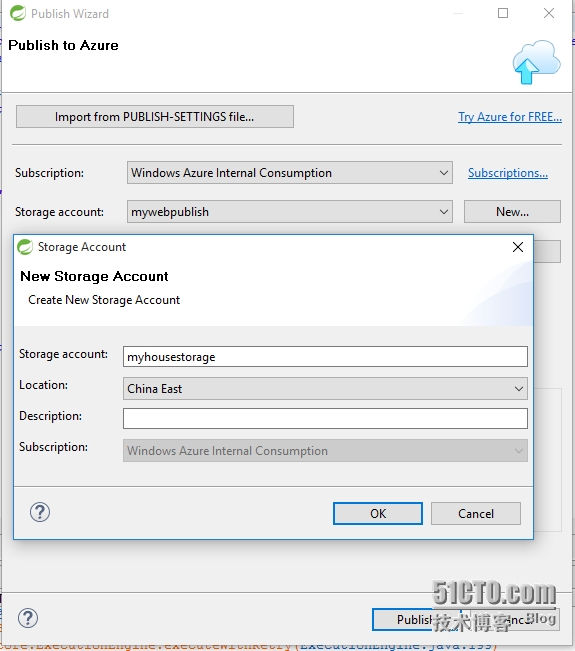
12.最后你需要添加你的订阅，以让部署脚本知道，你要将该应用部署在哪个订阅下.单击按钮“import from PUBLISH-SETTING file”会自动跳转到Azure China的登录界面，输入你的Azure帐号密码，会自动下载和导入setting文件，如下图所示，完成后，点击OK按钮退出

[](http://s3.51cto.com/wyfs02/M01/74/05/wKiom1YN6hzA2J_3AAJjtcd_Qqw781.jpg)

[](http://s3.51cto.com/wyfs02/M02/74/02/wKioL1YN5ZGCsonhAAFfDjkEDAw846.jpg)

13. 回到项目，单击右键选择Azure，选择第一项”Deploy to Azure Cloud“，你可以看到在该界面中，已经列出了你的订阅，你要部署到云端订阅的默认的存储帐号，云服务等等，由于本次是新部署，所以我们选择新建存储，将该部署所有的实例，应用存放到一个存储帐号下，选择”New“按钮，在弹出的界面中输入存储账号名称，选择Location，需要注意如果你希望你的应用部署在East或者North，那么对应的后续配置都需要选择同样的地区：

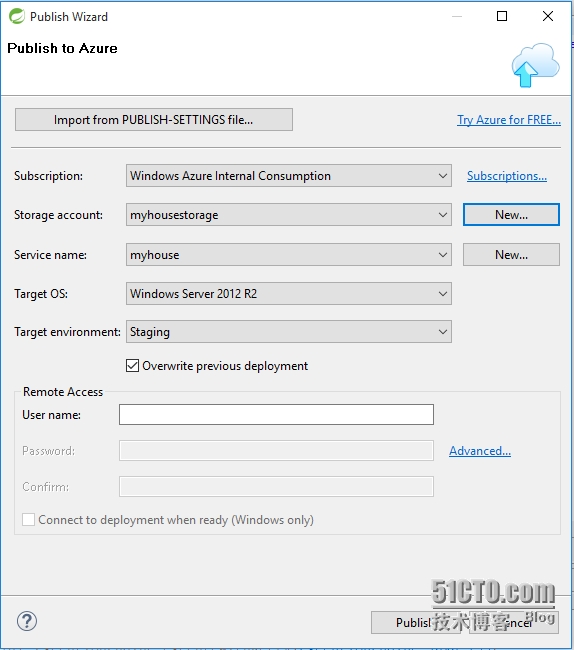
[](http://s3.51cto.com/wyfs02/M02/74/02/wKioL1YN6ULiCaFPAAMe2-Ko_bY214.jpg)

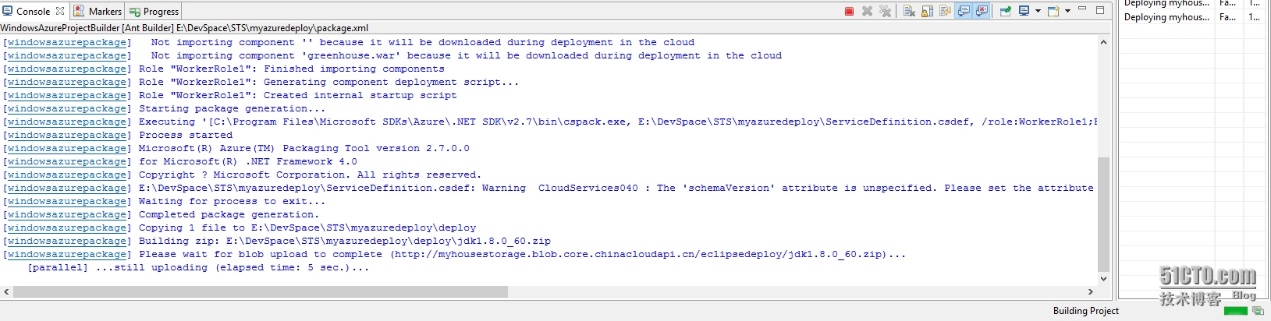
[](http://s3.51cto.com/wyfs02/M01/74/02/wKioL1YN6CvQvbgEAAIYu0zJ_P0897.jpg)

14.同样的道理，选择新建云服务，例子中名称为myhouse，同样选择China East作为地区：

[](http://s3.51cto.com/wyfs02/M02/74/05/wKiom1YN6SCAsurFAAH-9pgVvzQ980.jpg)

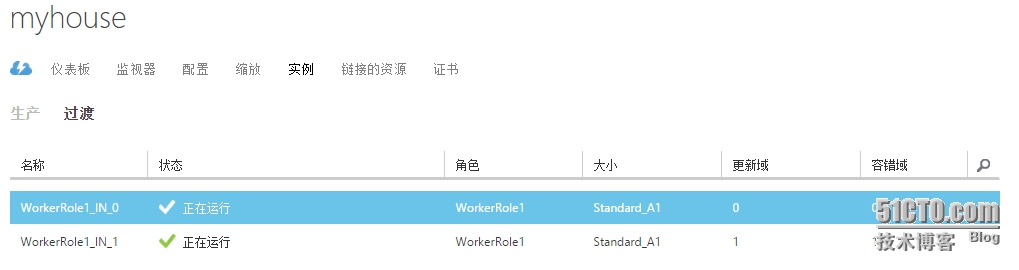
15.配置完成后如下图所示，点击发布，那么部署程序自动帮你创建存储账号，云服务，创建虚拟机，发布应用：

[](http://s3.51cto.com/wyfs02/M00/74/05/wKiom1YN6e-D8-glAAITjwRRkx8547.jpg)

[](http://s3.51cto.com/wyfs02/M00/74/02/wKioL1YN6f7z0H5gAAbI1mSjwT4823.jpg)

16.显示部署完成后，你可以登陆到你的Azure管理门户，查看部署的云服务和实例情况，在云服务的仪表板上，你可以找打站点的URL，选择实例页面，你也可以看到按照我们的定义，已经为改云服务创建了2个实例，并部分在不同的容错域：

[](http://s3.51cto.com/wyfs02/M02/74/02/wKioL1YN6yaBpGoLAAIeBV5ph6U884.jpg)

[](http://s3.51cto.com/wyfs02/M00/74/05/wKiom1YN6xnzQVJtAAD7yJAat7o050.jpg)

17.最后，我们可以测试一下我们的发布成果，在你的浏览器中输入站点名称，和应用名称，例如:http://XXXXX.chinacloudapp.cn/greenhouse/,你就可以看到你的Java web服务正常工作如下：

[](http://s3.51cto.com/wyfs02/M00/74/02/wKioL1YN67rzsC5rAAHVI4ZLIVM604.jpg)

# 高级内容

## Cloud Service是非持久化虚拟机

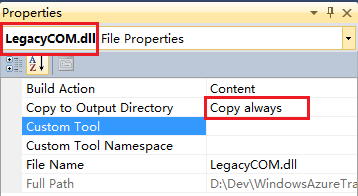
在笔者之前的文章中已经介绍了，Azure Cloud Service是非持久化虚拟机，应用程序时通过CSPKG和CSCFG文件，打包上传到Azure云端。

任何针对Azure Cloud Service本地磁盘的操作，都是无效的。我们需要把增量文件，保存到Azure Storage云存储里。

## 添加外部组件

我们在开发Azure Cloud Service的时候，会调用第三方外部类库，我们需要把这些类库在编译、部署的时候，把这些类库编译到CSPKG文件里。

我们打开Visual Studio，将需要的dll类库包含在项目里(including in Project)，并且将属性中的”Copy to Output Directory”设置成”Copy Always”



这样这个LegacyCom.dll类库就会编译到CSPKG文件里。

## 注册第三方组件

我们在做Web应用部署的时候，经常会遇到需要额外安装的第三方软件，比如安装Office，安装FTP Server。而我们在Azure Cloud Service是无法安装软件的，

[Windows Azure Cloud Service (24) 使用Startup注册COM组件(上)](http://www.cnblogs.com/threestone/archive/2012/03/13/2394484.html)

[Windows Azure Cloud Service (25) 使用Startup注册COM组件(下)](http://www.cnblogs.com/threestone/archive/2012/04/02/2429761.html)

## 修改时区

<http://www.cnblogs.com/threestone/p/4845520.html>

## 修改IIS托管管道模式为4.0经典模式

<http://www.cnblogs.com/threestone/p/4351945.html>

## Cloud Service虚拟目录

<http://www.cnblogs.com/threestone/archive/2012/03/10/2389094.html>

## 加快部署速度

<http://www.cnblogs.com/threestone/p/4486392.html>

## 配置SSL证书

<http://www.cnblogs.com/threestone/p/3951227.html>

## 固定Cloud Service内网IP和公网IP

<http://www.cnblogs.com/threestone/p/4360587.html>

## 收集日志记录

<http://www.cnblogs.com/threestone/archive/2012/01/30/2332078.html>

## Web Role和Worker Role交互

[Windows Azure Cloud Service (11) PaaS之Web Role, Worker Role(上)](http://www.cnblogs.com/threestone/p/3410510.html)

[Windows Azure Cloud Service (12) PaaS之Web Role, Worker Role, Azure Storage Queue(下)](http://www.cnblogs.com/threestone/p/4201065.html)