工资计算器应用

部署及使用手册

June 2018

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

The names of manufacturers, products, or URLs are provided for informational purposes only, and Microsoft makes no representations and warranties, either expressed, implied, or statutory, regarding these manufacturers or the use of the products with any Microsoft technologies. The inclusion of a manufacturer or product does not imply endorsement of Microsoft of the manufacturer or product. Links may be provided to third-party sites. Such sites are not under the control of Microsoft and Microsoft is not responsible for the contents of any linked site or any link contained in a linked site, or any changes or updates to such sites. Microsoft is not responsible for webcasting or any other form of transmission received from any linked site. Microsoft is providing these links to you only as a convenience, and the inclusion of any link does not imply endorsement of Microsoft of the site or the products contained therein.

© 2018 Microsoft Corporation. All rights reserved.

Microsoft and the trademarks listed at <https://www.microsoft.com/en-us/legal/intellectualproperty/Trademarks/Usage/General.aspx> are trademarks of the Microsoft group of companies. All other trademarks are the property of their respective owners.

Contents

[工资计算器应用 1](#_Toc517962277)

[设计目标 1](#_Toc517962278)

[前提条件 3](#_Toc517962279)

[第一章：编写工资计算器应用 3](#_Toc517962280)

[Task 1: 创建ASP.NET Web Application 项目 3](#_Toc517962281)

[Task 2: 运行项目 9](#_Toc517962282)

[Task 3: 部署应用到Azure 10](#_Toc517962283)

[第二章：利用GitHub和VSTS配置CI/CD 14](#_Toc517962284)

[Task 1: 配置GitHub代码仓库 14](#_Toc517962285)

[Task 2: 配置VSTS中CI/CD Pipeline. 21](#_Toc517962286)

[Task 3: 在VSTS中查看CI/CD Pipeline的相关设置. 24](#_Toc517962287)

[第三章: 项目测试 28](#_Toc517962288)

[Task 1: 提交代码更改 28](#_Toc517962289)

[Task 2: 验证CI/CD 32](#_Toc517962290)

[小结 35](#_Toc517962291)

工资计算器应用

## 设计目标

工资单计算器程序

当输入员工的详细信息：名字，姓氏，年薪（正整数），工资起始日期时，计算器程序应产生姓名，工资支付期，总收入，所得税，净收入, 养老金，医疗保险，失业金。

您的解决方案需满足以下条件：

* 列出您在编写这个程序时候所作的假设
* 编写程序（编程语言不限，技术栈不限）
* 编写文档来说明如何运行您的程序
* 编写测试方案来说明如何验证您的程序
* 编写把你的程序部署到azure上的CICD pipeline方案（需要包含infra as code，技术栈不限 ）
* 文档和代码需要上传至github

#计算规则如下#

工资支付期=每个日历月

总收入=年薪/ 12个月

养老金 = 以税前工资为基数按下表计算

医疗保险 = 以税前工资为基数按下表计算

失业金 = 以税前工资为基数按下表计算

所得税=以（总收入-养老金-医疗保险-失业金）为基数按税表计算

净收入=总收入 - 所得税 – 养老金 – 医疗保险 – 失业金

注：所有计算结果应四舍五入到元

三金缴费额

|  |  |
| --- | --- |
| 缴费基数 | 4279-21396 元 |
| 养老金 | 8% |
| 医疗保险 | 2% |
| 失业金 | 0.5% |

应纳税所得额

|  |  |  |
| --- | --- | --- |
| 级数 | 全月应纳税所得额（含税级距） | 税率(%) |
| 1 | 不超过1,500元 | 3 |
| 2 | 超过1,500元至4,500元的部分 | 10 |
| 3 | 超过4,500元至9,000元的部分 | 20 |
| 4 | 超过9,000元至35,000元的部分 | 25 |
| 5 | 超过35,000元至55,000元的部分 | 30 |
| 6 | 超过55,000元至80,000元的部分 | 35 |
| 7 | 超过80,000元的部分 | 45 |

＃示例数据＃

员工年薪为60,050，该员工在3月份的工资收入多少钱？

（3月1日至3月31日）

总收入= 60,050 / 12 = 5,004.16666667（舍​​入）= 5,004

养老金 = 5004\*8% = 400

医疗保险 = 5004\*2% = 100

失业金 = 5004\*0.5% = 25

所得税= （（5004-400-100-25）-1500）\*10%+1500\*3%= 343

净收入= 5,004 – 400-100-25-343 = 4136

以下是我们提供的csv输入和输出格式。 （这里的csv格式仅供参考，可以使用任意格式）

输入（名字，姓氏，年薪，工资起始日期）：

三，张，60050，3月1日 - 3月31日

产出（名称，工资期，总收入，所得税，净收入，养老金，医疗保险，失业金）：

三 张，3月1日至3月31日，5004,343,4136,400,100,25

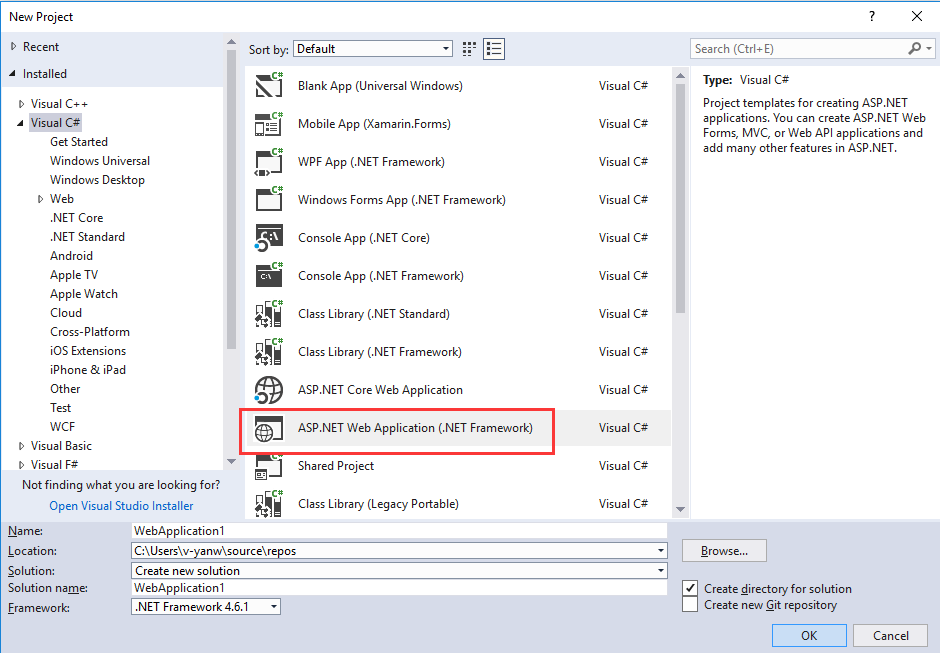
## 前提条件

* 拥有一个Microsoft Azure 订阅并具有至少 co-administrator权限
* 开发机配置
  + Visual Studio Community 2017 版本 15.4 或更高版本
    - <https://www.visualstudio.com/vs/>
  + Azure development workload for Visual Studio 2017
    - <https://docs.microsoft.com/azure/azure-functions/functions-develop-vs#prerequisites>
  + ASP.NET and web development workload for Visual Studio 2017

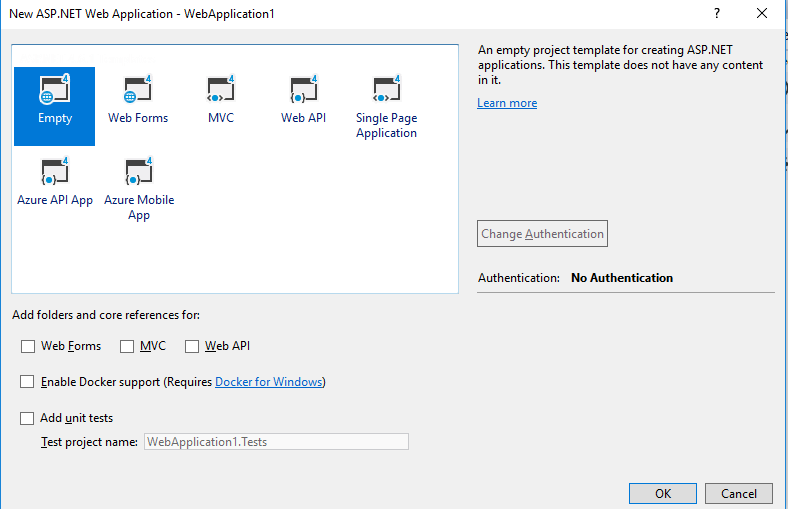
## 第一章：编写工资计算器应用

#### Task 1: 创建ASP.NET Web Application 项目

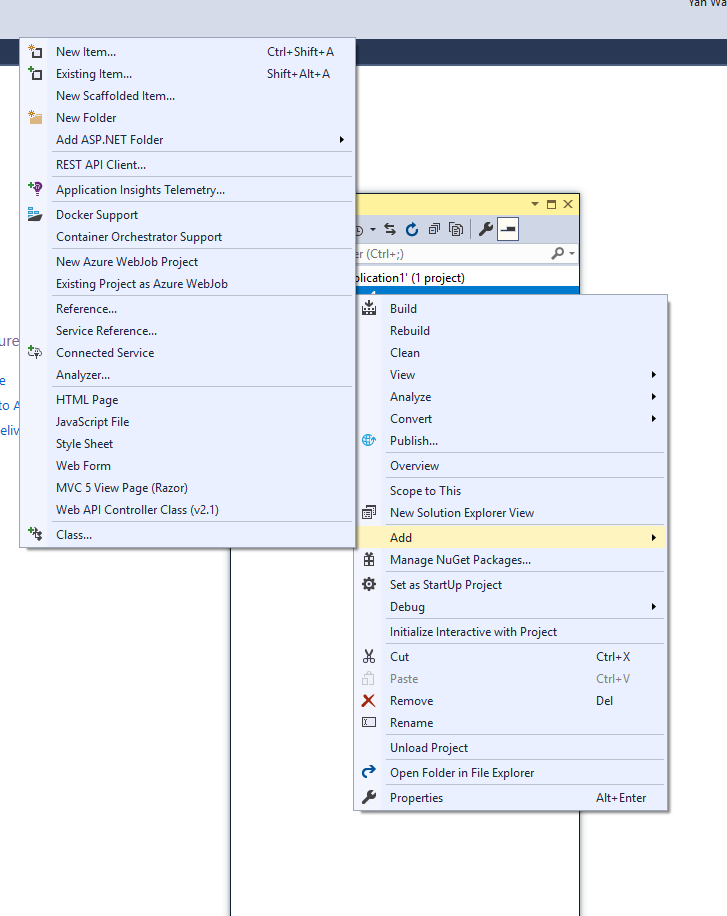
1. 以管理员模式打开Visual Studio 2017 ， 创建一个ASP.Net Web application 项目。

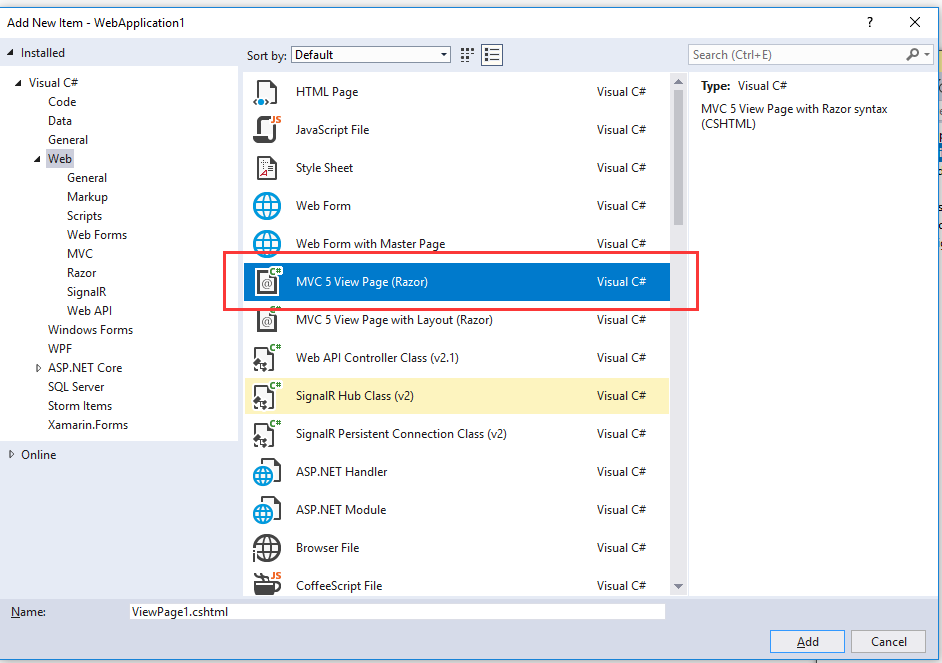


1. 选择Empty 模板。



1. 在Solution Explorer里选择添加一个New Item, item类型选择“ MVC5 View Page (Razor)”,命名为ViewPage1.cshtml.





1. 编辑ViewPage1.cshtml, 替换为如下代码：

<!DOCTYPE html>

<html>

<body>

@{

if (IsPost)

{

string firstname = Request["firstname"];

string lastname = Request["lastname"];

int salary = Request["annualpay"].AsInt();

string salarydate = Request["period"];

int total = salary / 12;

var pension = Math.Round(total \* 0.08);

var medical = Math.Round(total \* 0.02);

var dismiss = Math.Round(total \* 0.005);

var taxamount = Math.Round(total - pension - medical - dismiss);

double tax = 0;

if (taxamount >= 80000)

{ tax = Math.Round((taxamount - 80000) \* 0.45 + 25000 \* 0.35 + 20000 \* 0.3 + 29000 \* 0.25 + 4500 \* 0.2 + 3000 \* 0.1 + 1500 \* 0.03); }

else if (taxamount >= 55000)

{ tax = Math.Round((taxamount - 55000) \* 0.35 + 20000 \* 0.3 + 29000 \* 0.25 + 4500 \* 0.2 + 3000 \* 0.1 + 1500 \* 0.03); }

else

if (taxamount >= 35000)

{ tax = Math.Round((taxamount - 35000) \* 0.3 + 29000 \* 0.25 + 4500 \* 0.2 + 3000 \* 0.1 + 1500 \* 0.03); }

else if (taxamount >= 9000)

{ tax = Math.Round((taxamount - 9000) \* 0.25 + 4500 \* 0.2 + 3000 \* 0.1 + 1500 \* 0.03); }

else if (taxamount >= 4500)

{ tax = Math.Round((taxamount - 4500) \* 0.2 + 3000 \* 0.1 + 1500 \* 0.03); }

else if (taxamount >= 1500)

{ tax = Math.Round((taxamount - 1500) \* 0.1 + 1500 \* 0.03); }

else

{ tax = Math.Round(taxamount \* 0.03); }

var netincome = total - tax - pension - medical - dismiss;

<p>

<h3>you entered:（名字:@firstname，姓氏:@lastname，年薪：@salary，工资起始日期:@salarydate）<br >

Your salary calculation result is :</h3> <br>

<table style="width:100%">

<tr>

<th>名称</th>

<th>工资期</th>

<th>总收入</th>

<th>所得税</th>

<th>净收入</th>

<th>养老金</th>

<th>医疗保险</th>

<th>失业金</th>

</tr>

<tr>

<th>@firstname @lastname</th>

<th>@salarydate</th>

<th>@total</th>

<th>@tax</th>

<th>@netincome</th>

<th>@pension</th>

<th>@medical</th>

<th>@dismiss</th>

</tr>

</table>

</p>

}

else

{

<form method="post" action="">

名字:

<input type="text" name="firstname" value=""><br><br />

姓氏:

<input type="text" name="lastName" value=""><br><br>

年薪:

<input type="number" name="annualpay" value=""><br><br>

工资起始日期： <select name="period">

<option value="1月1日-1月31日">1月1日-1月31日</option>

<option value="2月1日-2月28日">2月1日-2月28日</option>

<option value="3月1日-3月31日">3月1日-3月31日</option>

<option value="4月1日-4月30日">4月1日-4月30日</option>

<option value="5月1日-5月31日">5月1日-5月31日</option>

<option value="6月1日-6月30日">6月1日-6月31日</option>

<option value="7月1日-7月31日">7月1日-7月31日</option>

<option value="8月1日-8月30日">8月1日-8月31日</option>

<option value="9月1日-9月30日">9月1日-9月30日</option>

<option value="10月1日-10月31日">10月1日-10月31日</option>

<option value="11月1日-11月30日">11月1日-11月31日</option>

<option value="12月1日-12月31日">12月1日-12月31日</option>

</select>

<input type="submit" value="Submit" class="submit">

</form>

}

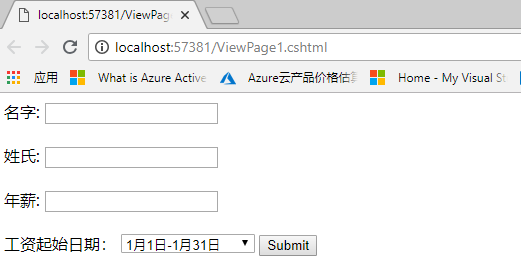
}

</body>

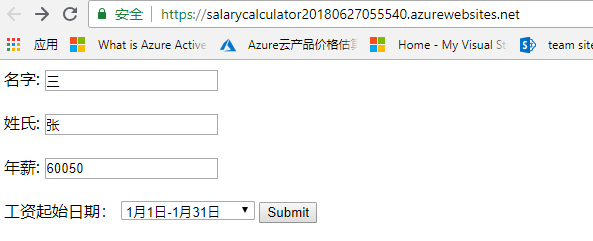
</html>

#### Task 2: 运行项目

1. 按F5启动应用，该项目会在本地浏览器中打开，如下图：



1. 输入名字，姓氏，年薪，工资起始日期后，点击Submit按钮。

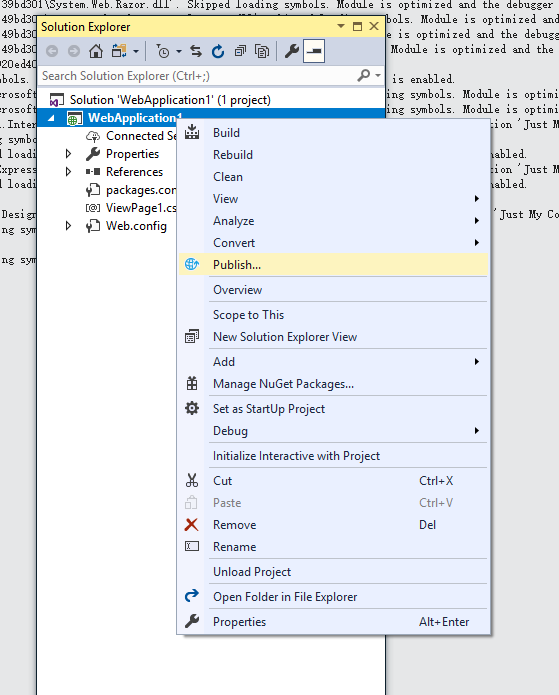


工资计算结果显示如下：

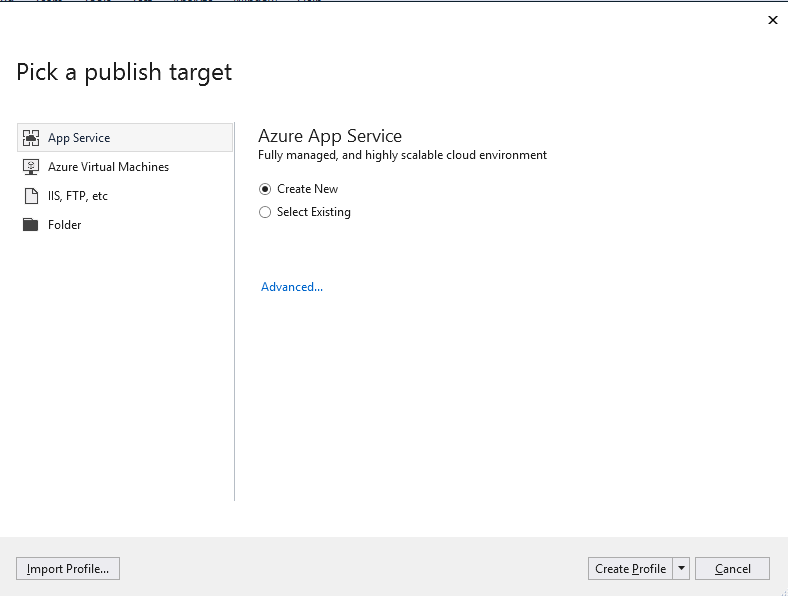


#### Task 3: 部署应用到Azure

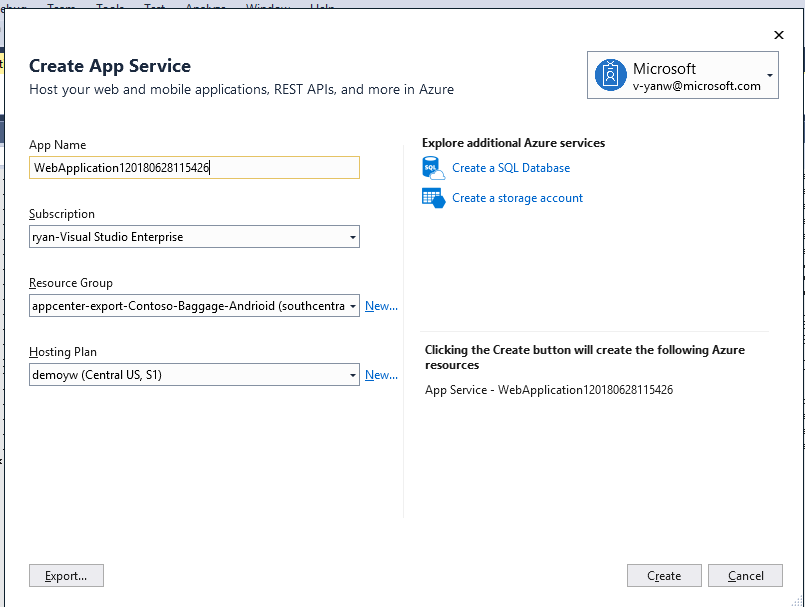
1. 在Solution Explorer里右击项目名称，选择Publish…菜单。



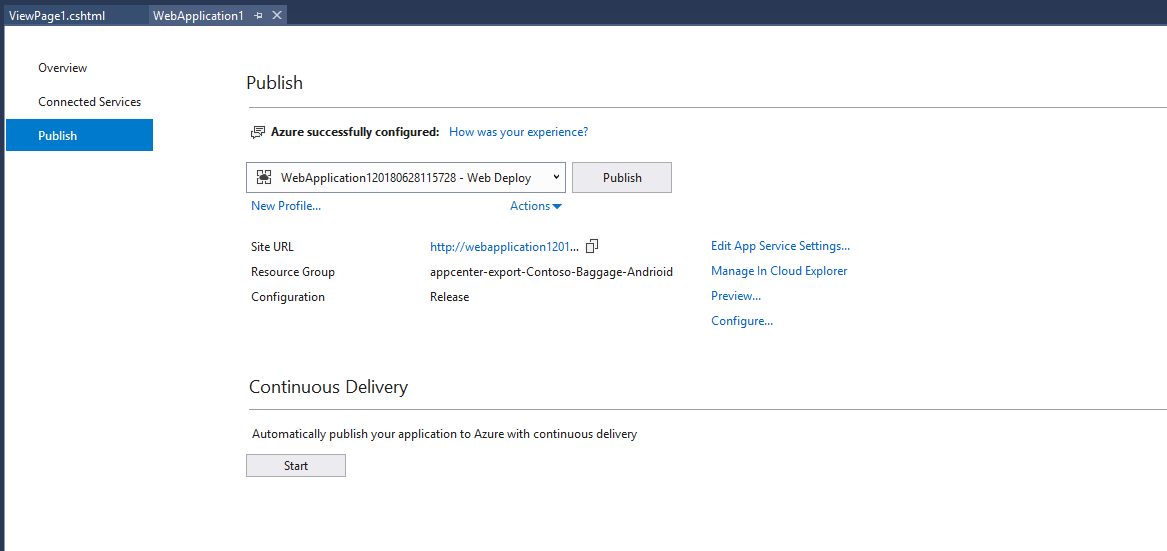
1. 选择App Services.

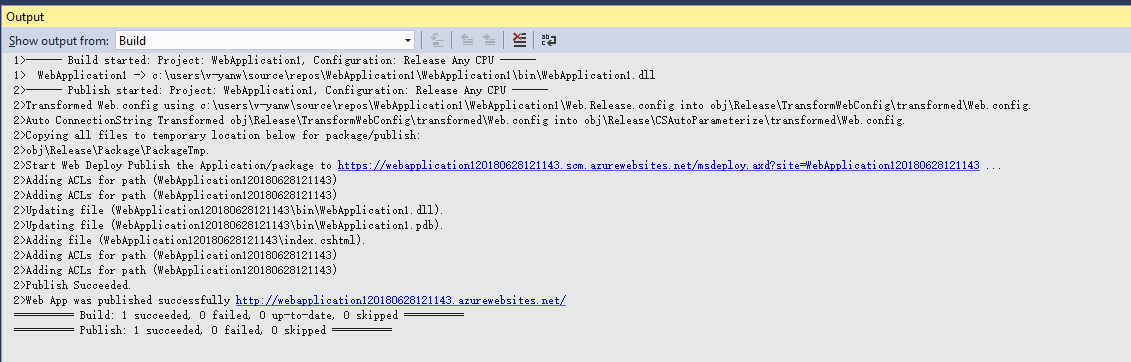


1. 选择 Create Profile按钮，用Azure账号登录，配置App Name, Subscription, Resource Group, Hosting Plan等，按Create按钮创建Profile.



1. 点击Publish按钮发布应用到Azure App Services.





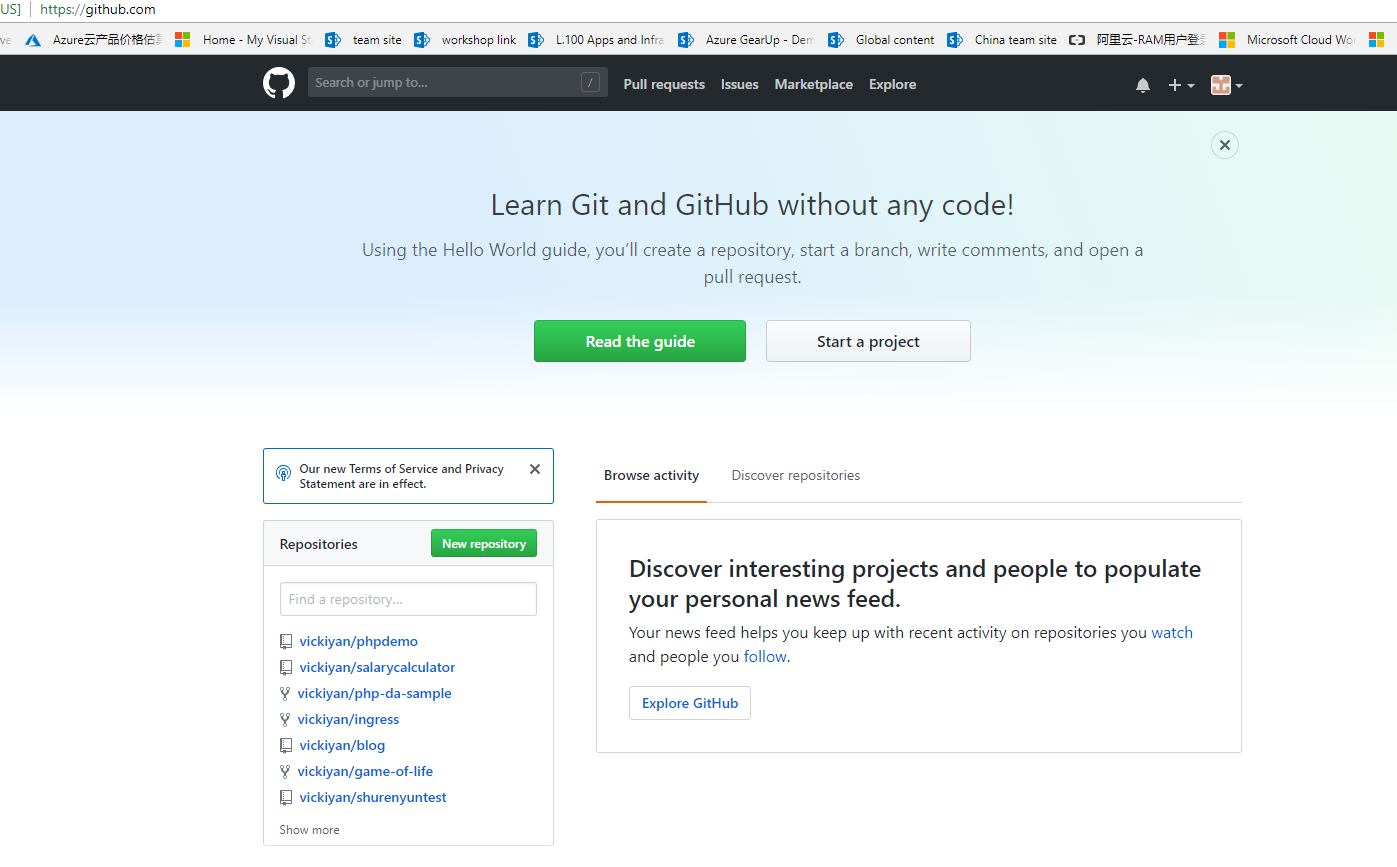
1. 显示发布成功后，在浏览器中打开上图所示的URL ,比如<https://webapplication120180628121143.azurewebsites.net/ViewPage1.cshtml>. 如出现下面界面说明部署成功。



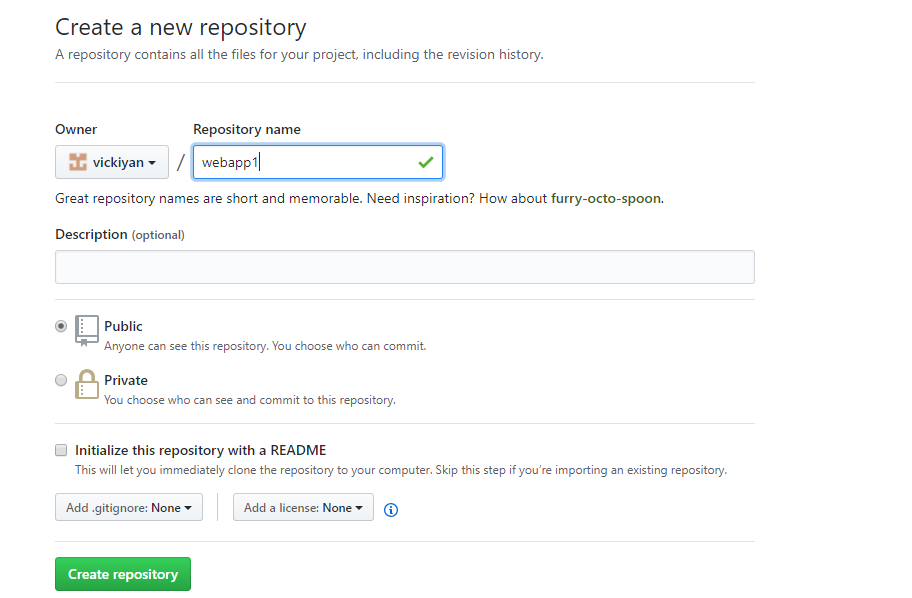
## 第二章：利用GitHub和VSTS配置CI/CD

#### Task 1: 配置GitHub代码仓库

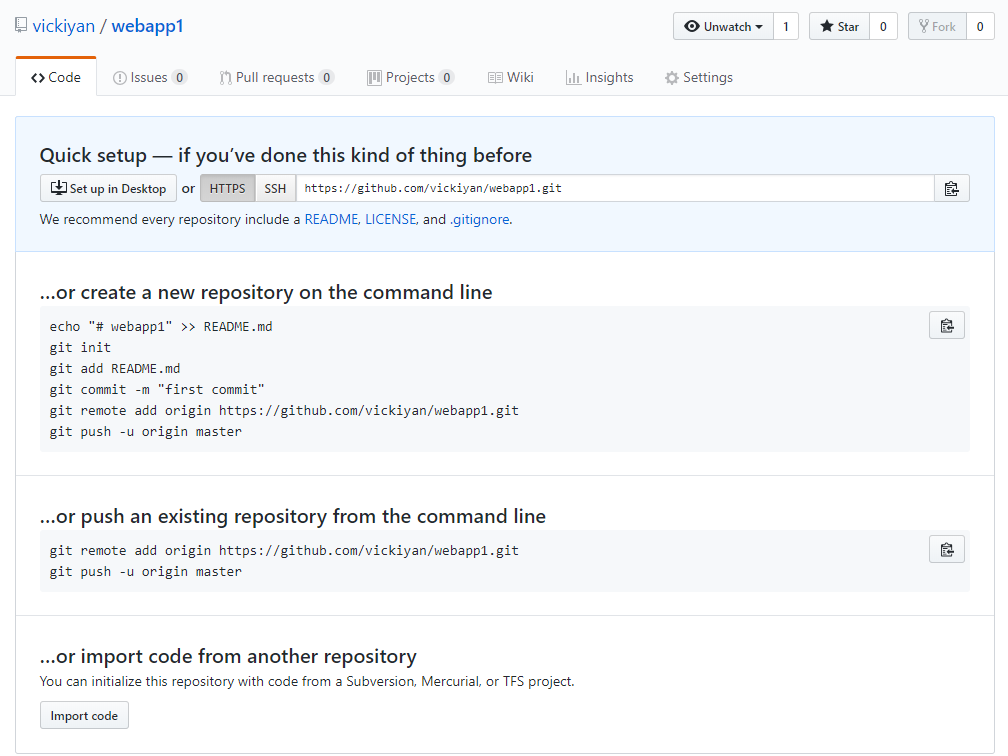
1. 用您的GitHub账号登录GitHub，点击New Repository创建一个新的代码仓库比如WebApp1.

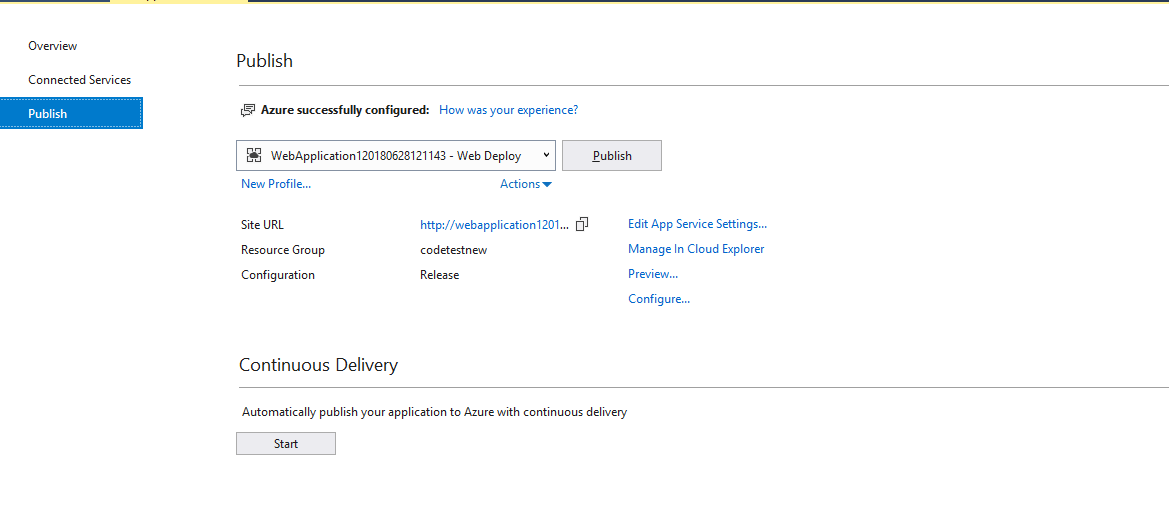


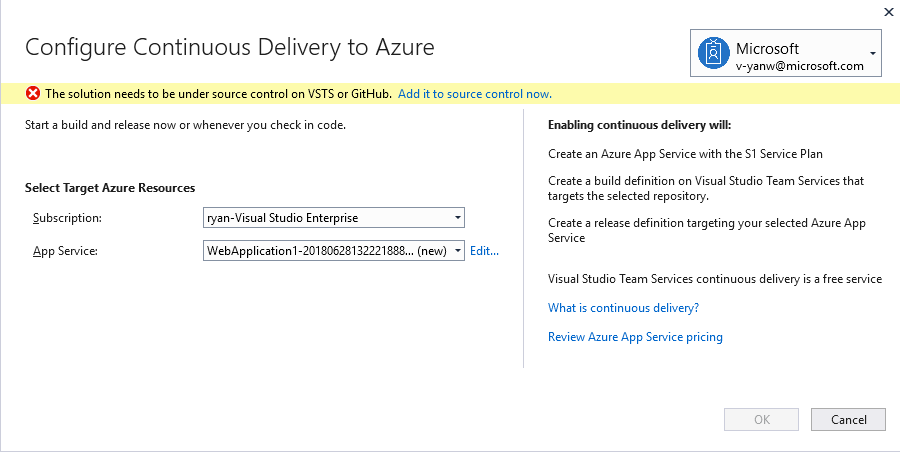
1. 点击Create, 创建。



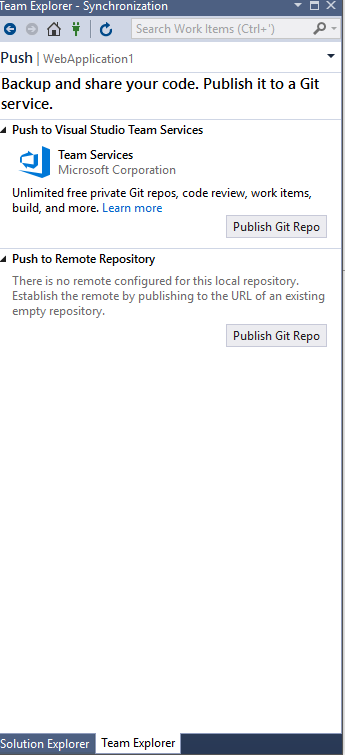
1. 记录下Repository Git 路径“https://github.com/vickiyan/webapp1.git”



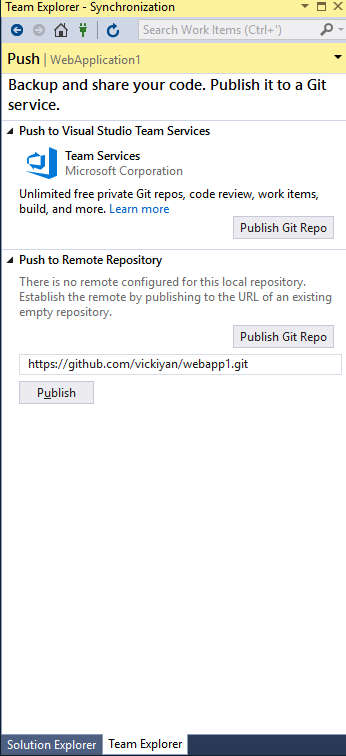
1. 在Solution Explorer中右键点击项目名称，选择Publish…菜单。点击Continuous Delivery项下的Start按钮.  
     
   
2. 点击Add it to source control now.

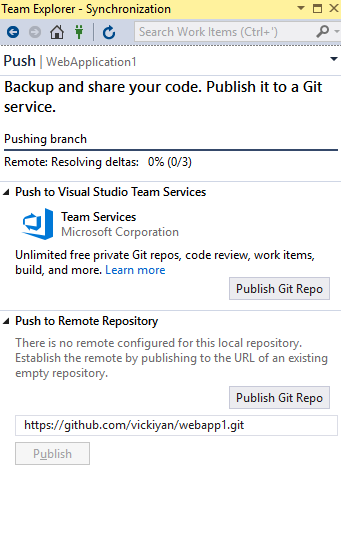


1. 选择 Publish to remote repository 项下的Publish Git Repo。

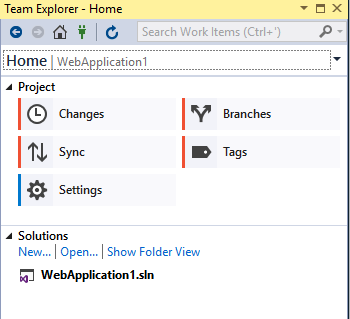


1. 填入第3步中的Git路径，点击Publish按钮。



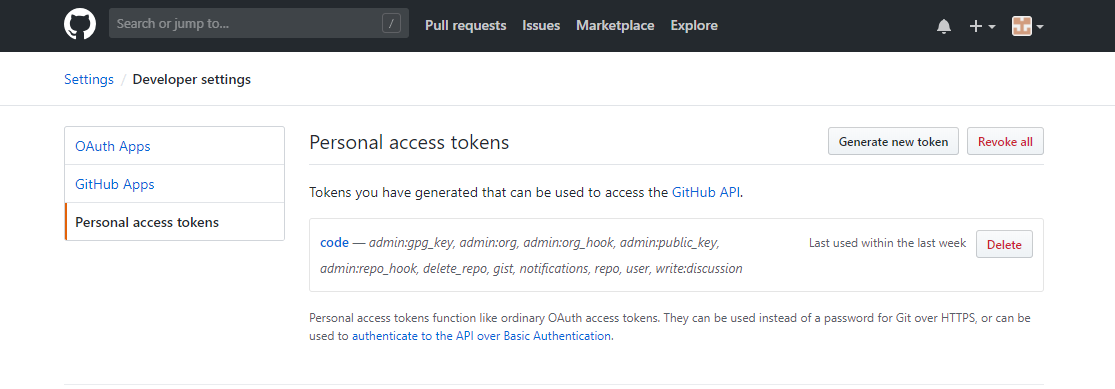


1. Publish成功后会出现下图：

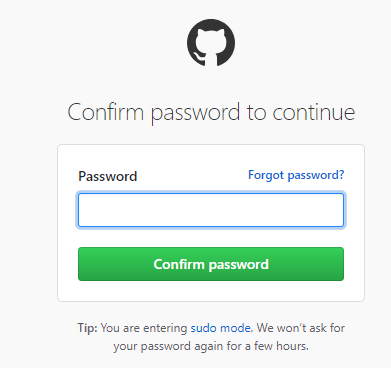


#### Task 2: 配置VSTS中CI/CD Pipeline.

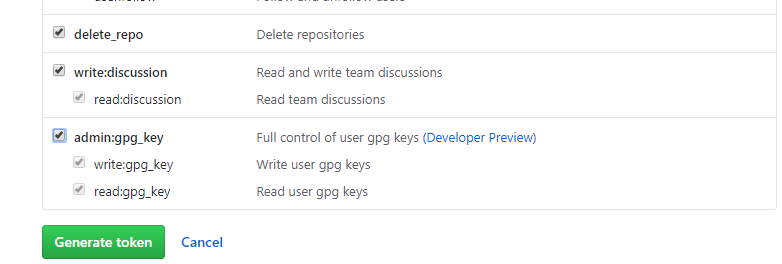
1. 打开https://github.com/settings/tokens获取GitHub PAT(Personal Access Token), 点击Generate New Token。



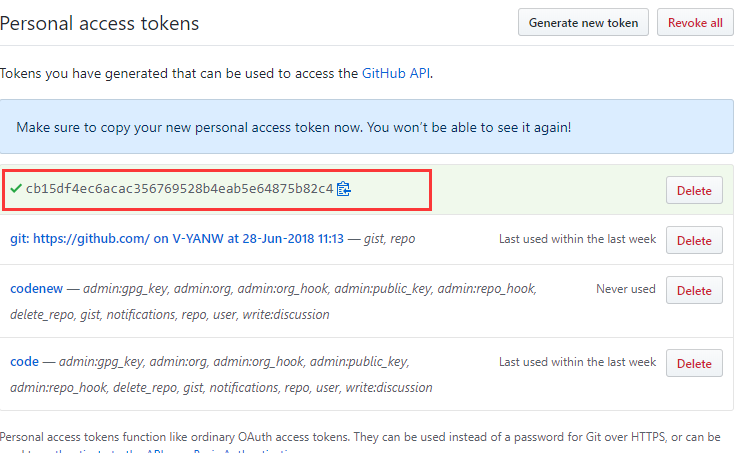
1. 会提示您输入Github密码。



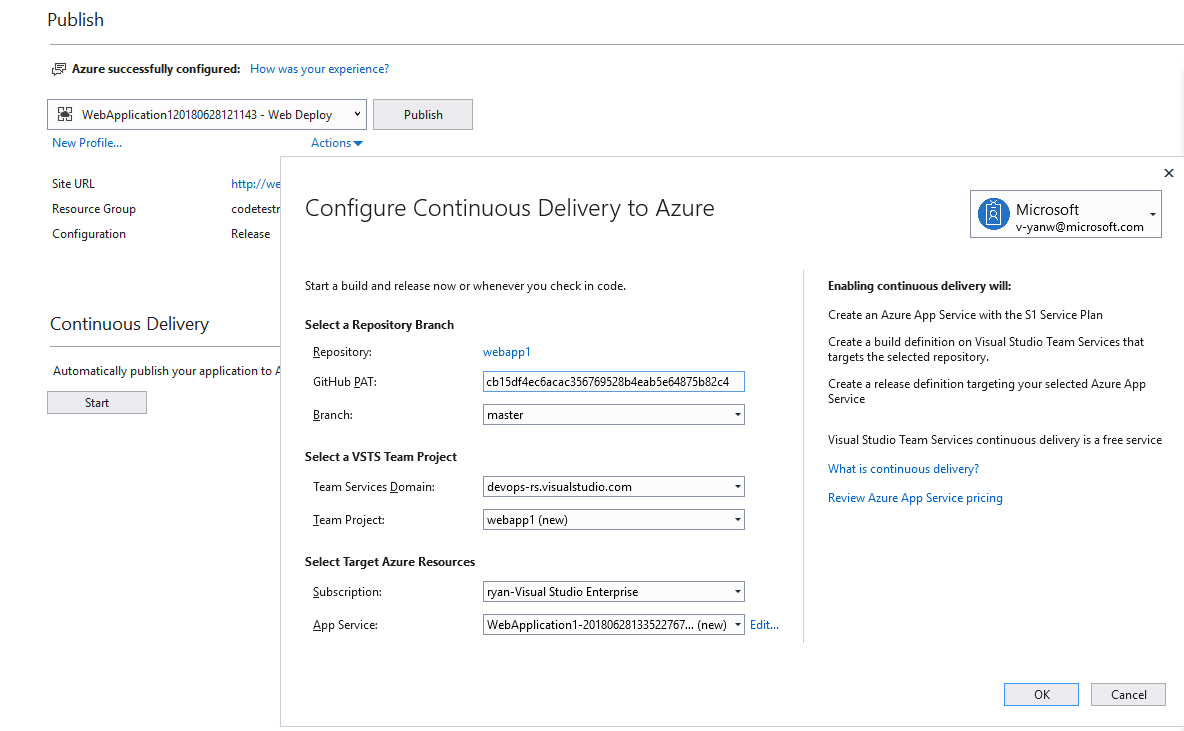
1. 选择相应的权限（为简便起见可以全选），点击Generate Token



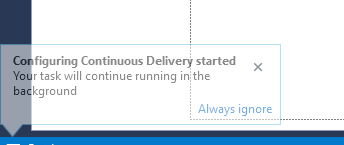
1. 记录下生成的PAT.



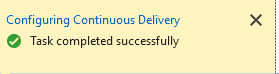
1. 回到Visual Studio应用，重新打开Publish窗口，选择Continuous Delivery 项下的Start按钮，在Configure Continuous Delivery to Azure 窗口，输入上步得到的PAT. 点击OK按钮开始配置。（注意App Service可以选择之前发布的Azure服务）



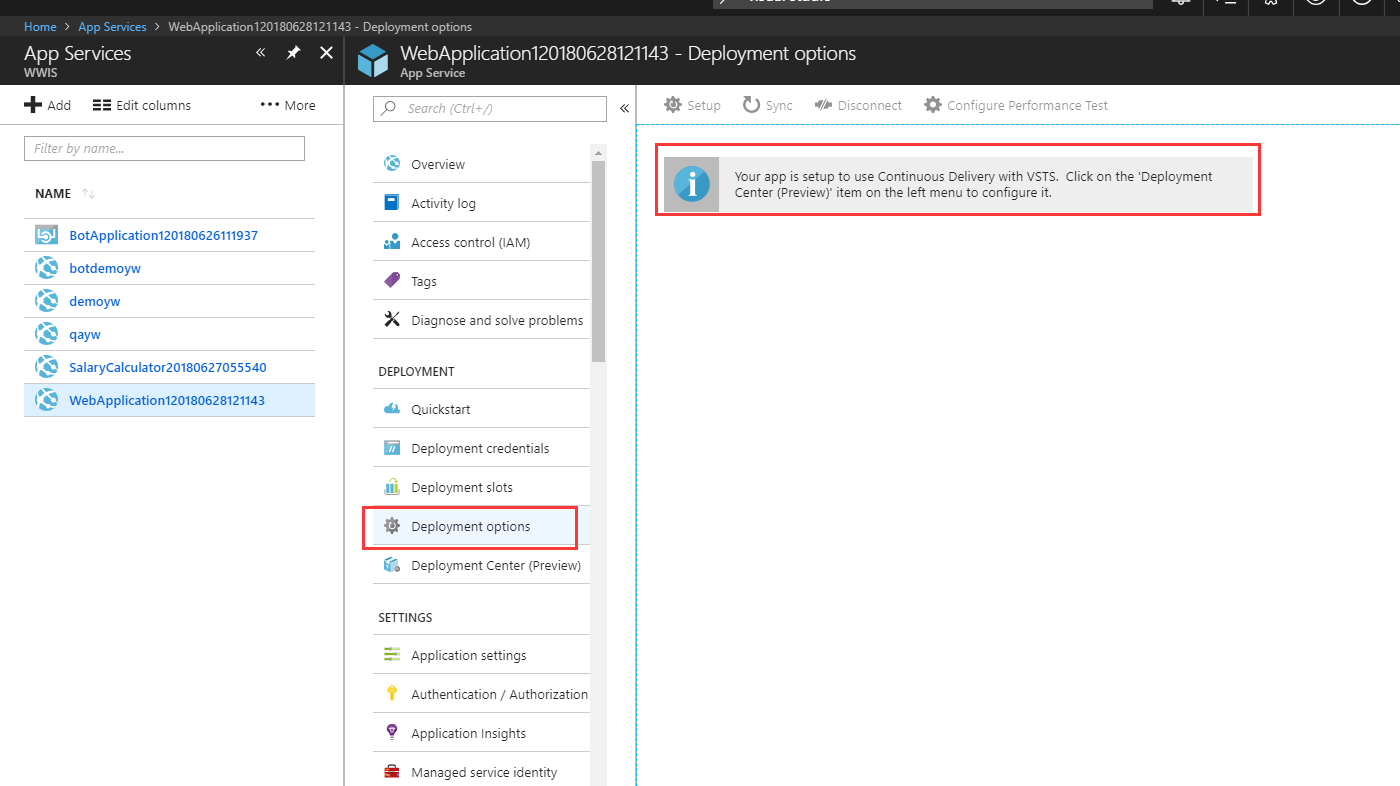
1. Visual Studio左下角会提示Configuring Continuous Delivery started.

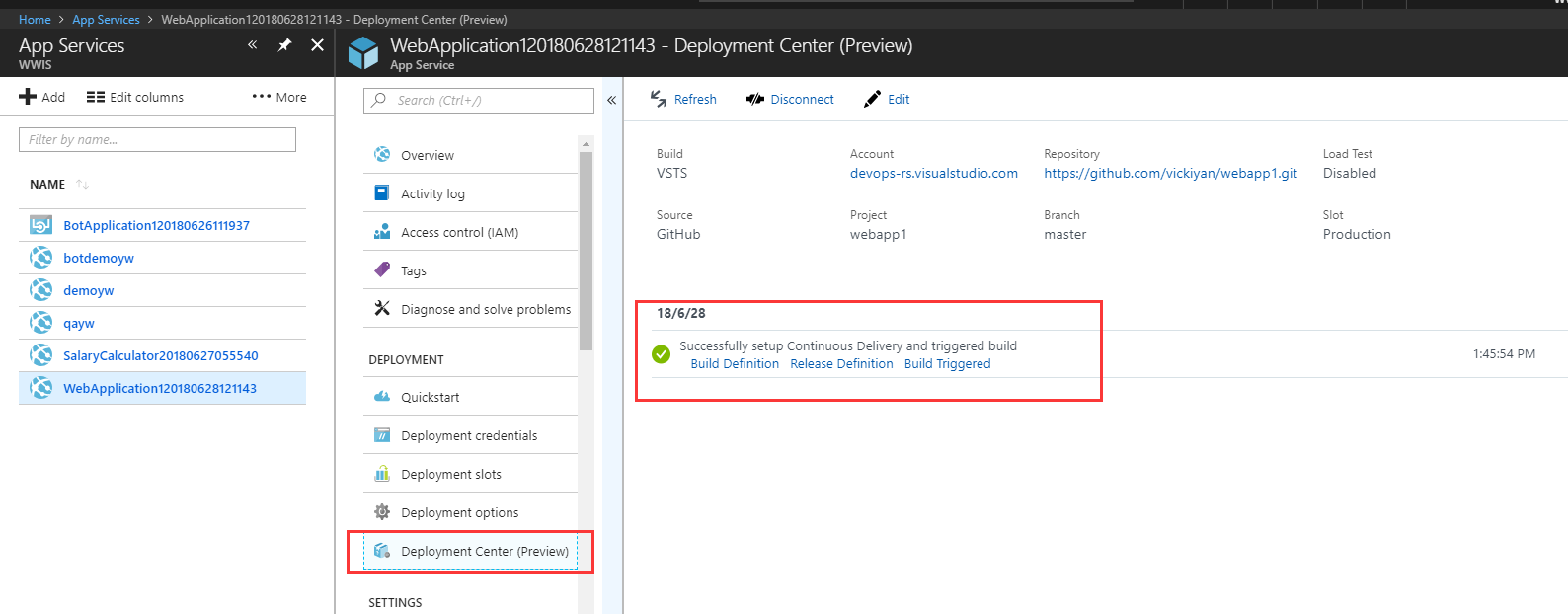


1. 配置结束后会提示成功。



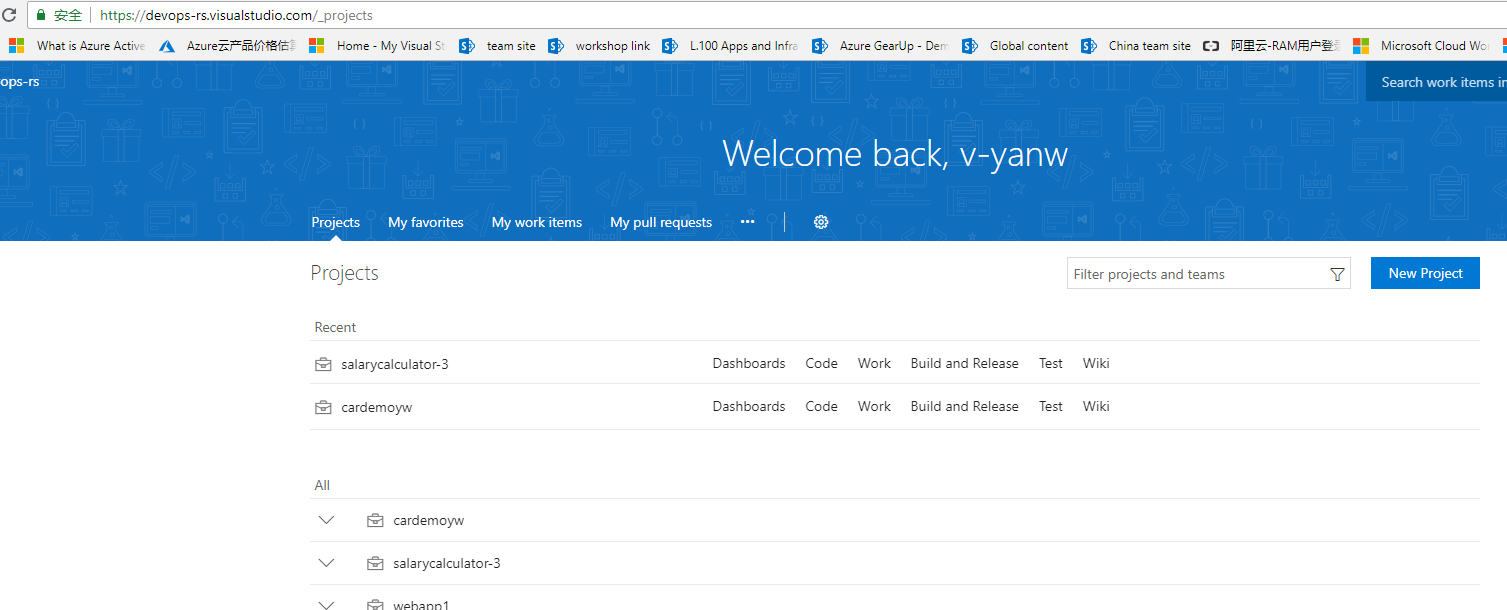
1. 在Azure Portal中打开相应的App Service管理门户，分别选择Deployment Options和Deployment Center(Preview)菜单，可以显示VSTS CI/CD配置成功。



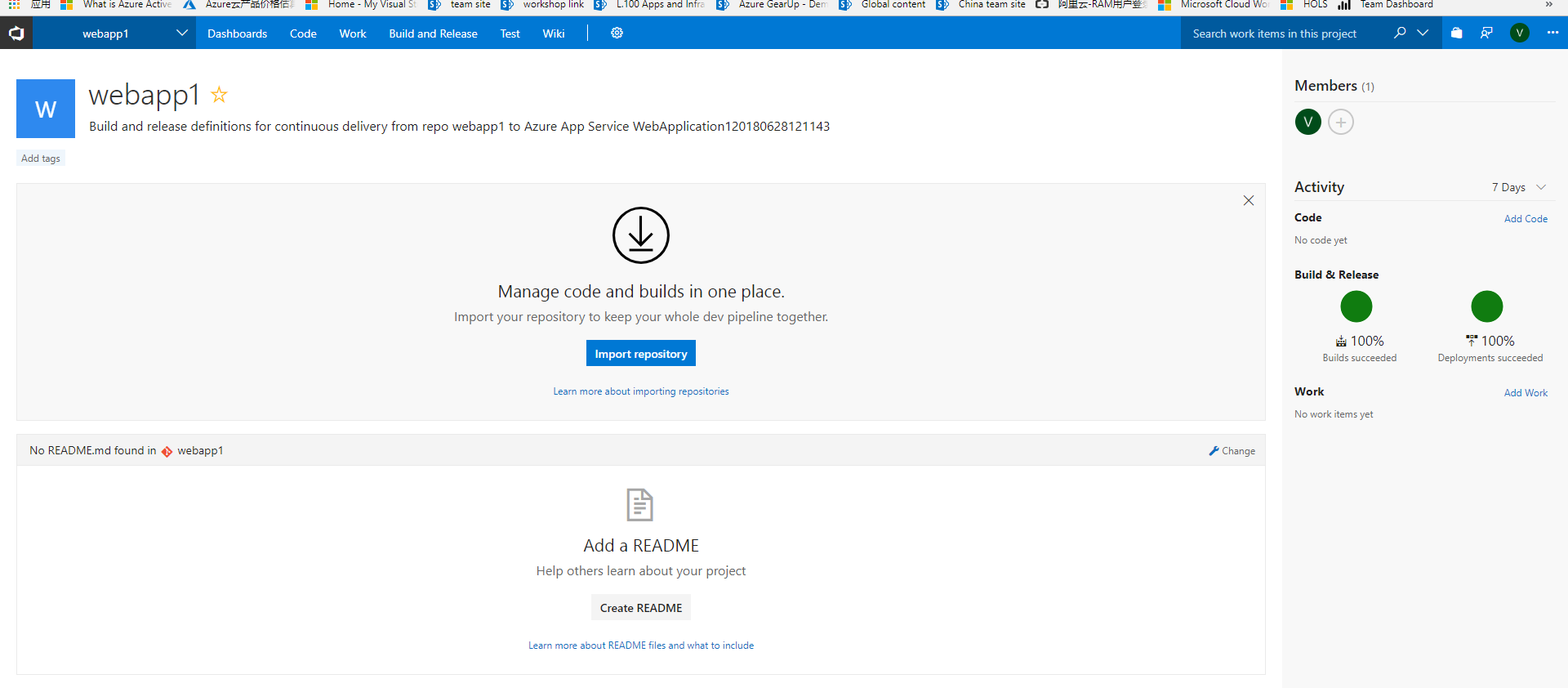


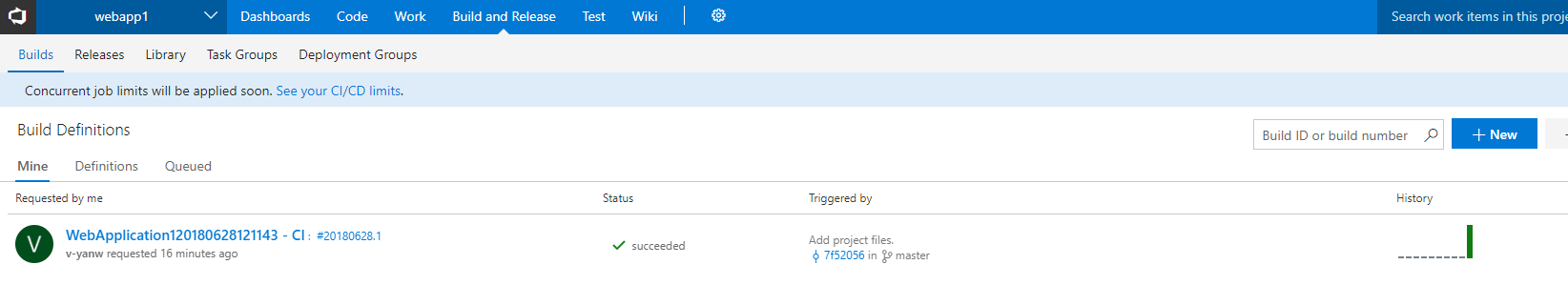
#### Task 3: 在VSTS中查看CI/CD Pipeline的相关设置.

1. 打开VSTS 门户，地址为https://<accountname>.visualstudio.com。选择创建的webapp1项目。

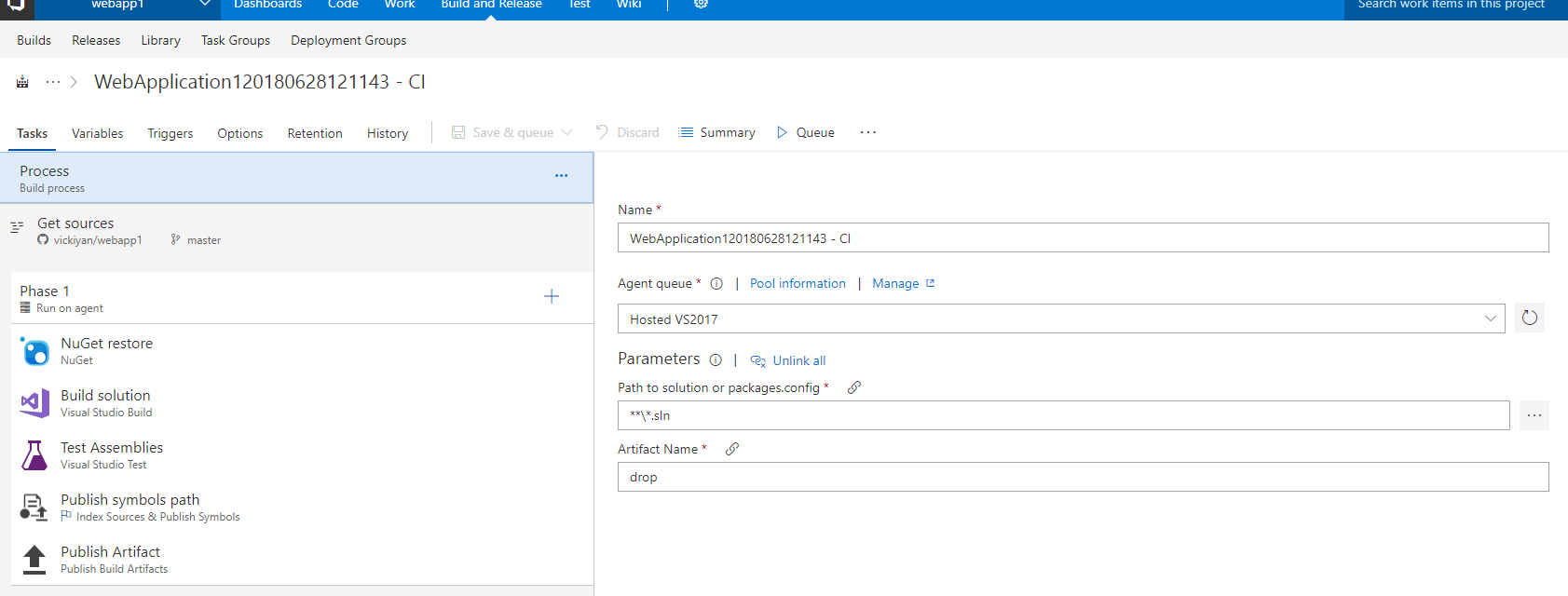


1. 选择Build and Release菜单下的Build按钮。

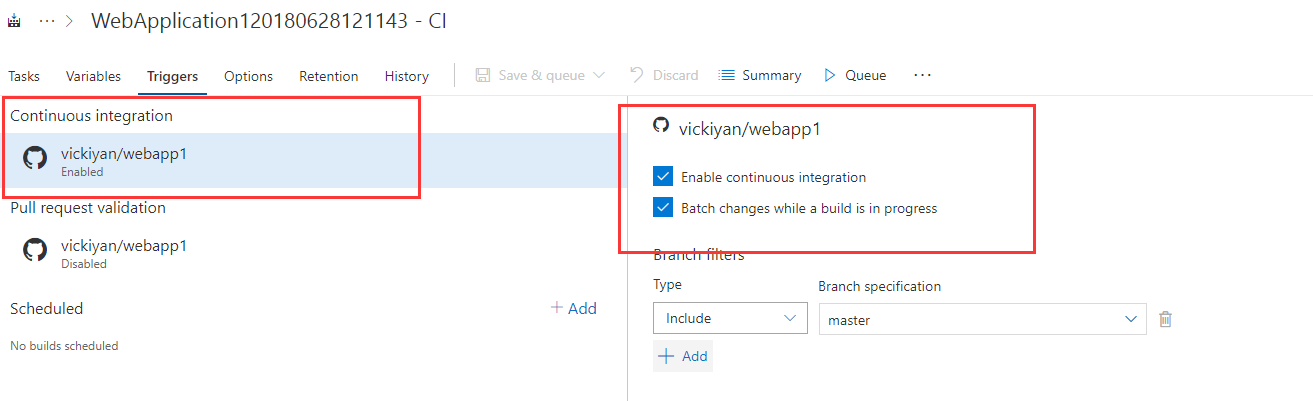




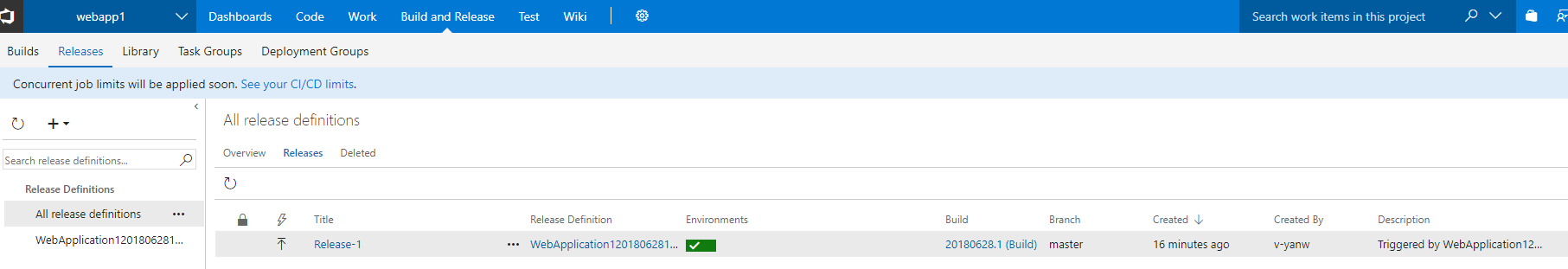
1. 点击省略号处的下拉菜单可以选择Edit Definition,可以查看Build定义。



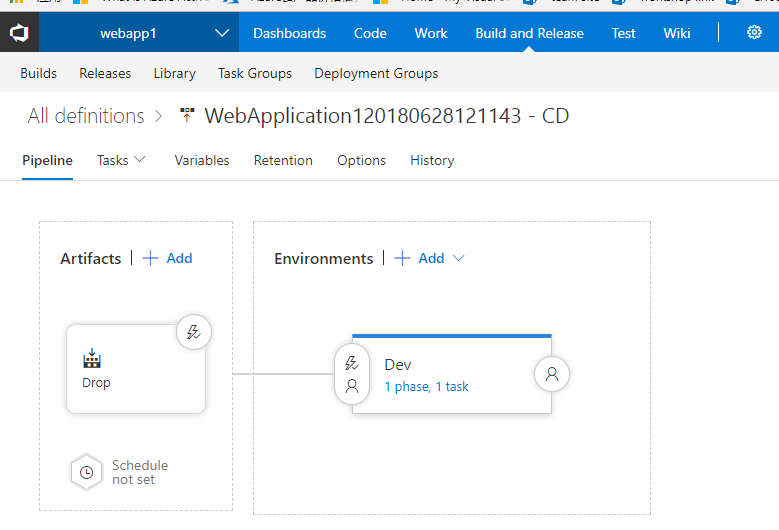
1. 点击Trigger 菜单，可以看到Continuous integration开关已经打开。



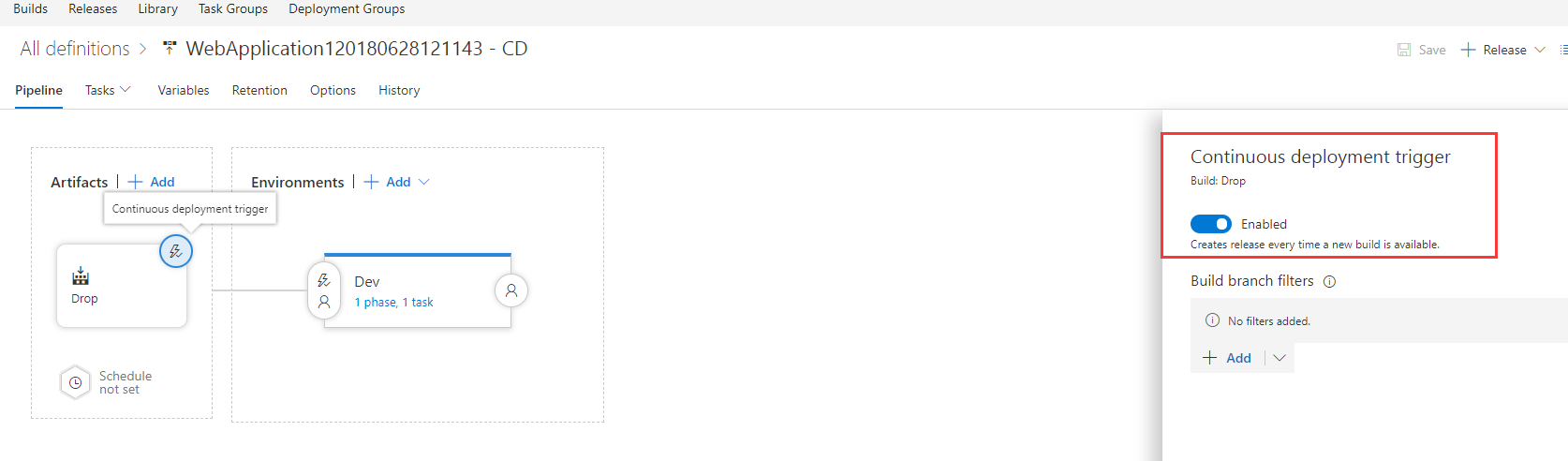
1. 选择Build and Release菜单的Release菜单可以显示Release界面。



1. 点击省略号处的下拉菜单选择Edit，可以显示CI/CD流程。



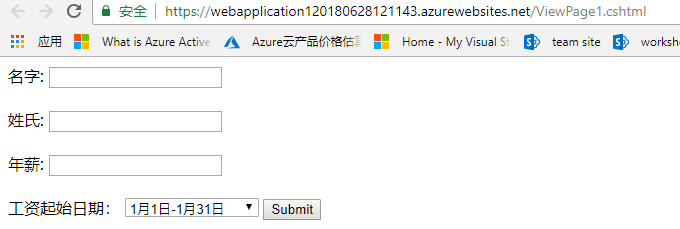
1. 点击闪电符号，可以看到Continuous deployment trigger 中持续部署开关已经打开。



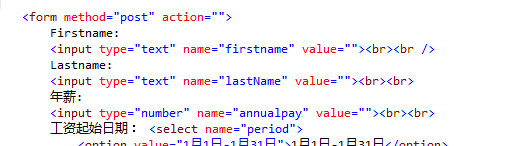
## 第三章: 项目测试

#### Task 1: 提交代码更改

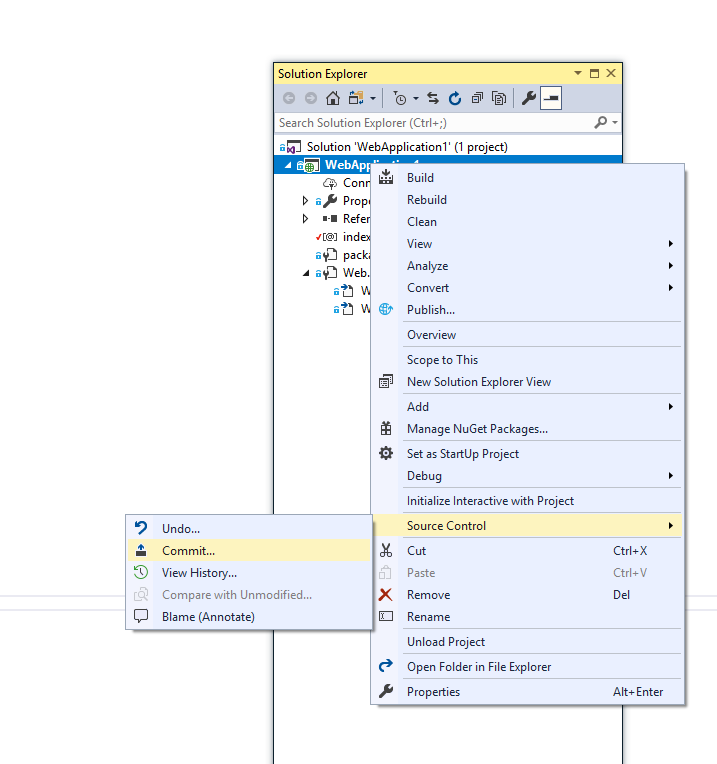
1. 打开之前发布的App Service网页，比如https://webapplication120180628121143.azurewebsites.net/ViewPage1.cshtml，起始页上效果如下：



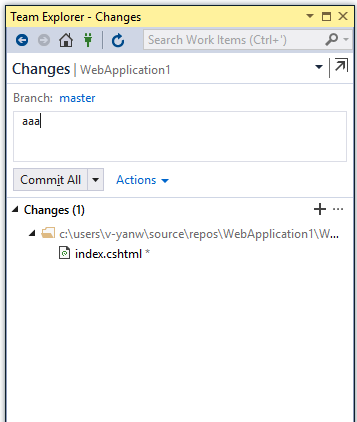
1. 在Visual Studio中编辑该项目，打开ViewPage1.cshtml文件，修改名字为FirstName,姓氏为LastName,如下图：



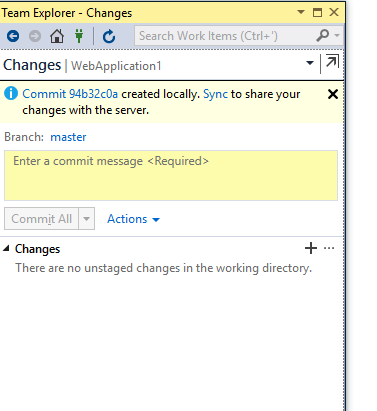
1. 在Solution Explorer中，右击项目名称，选择Source Control-commit



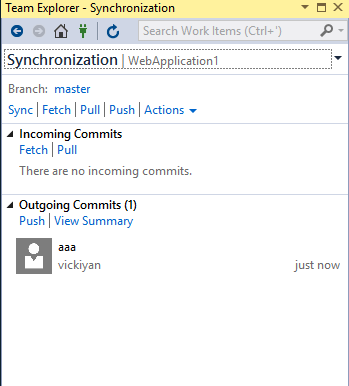
1. 在Team Explorer中，填写Commit comments,选择Commit All.



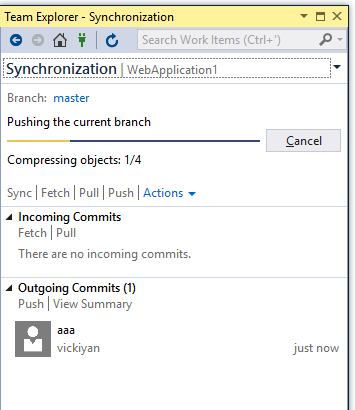
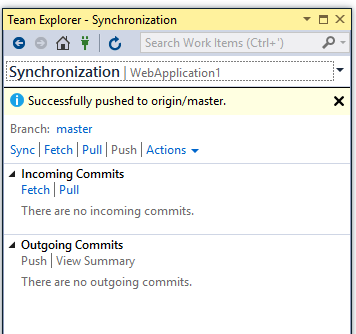
1. 提示需要Sync,点击Sync。



1. Sync完成后，点击Push, push代码更改到GitHub代码库。

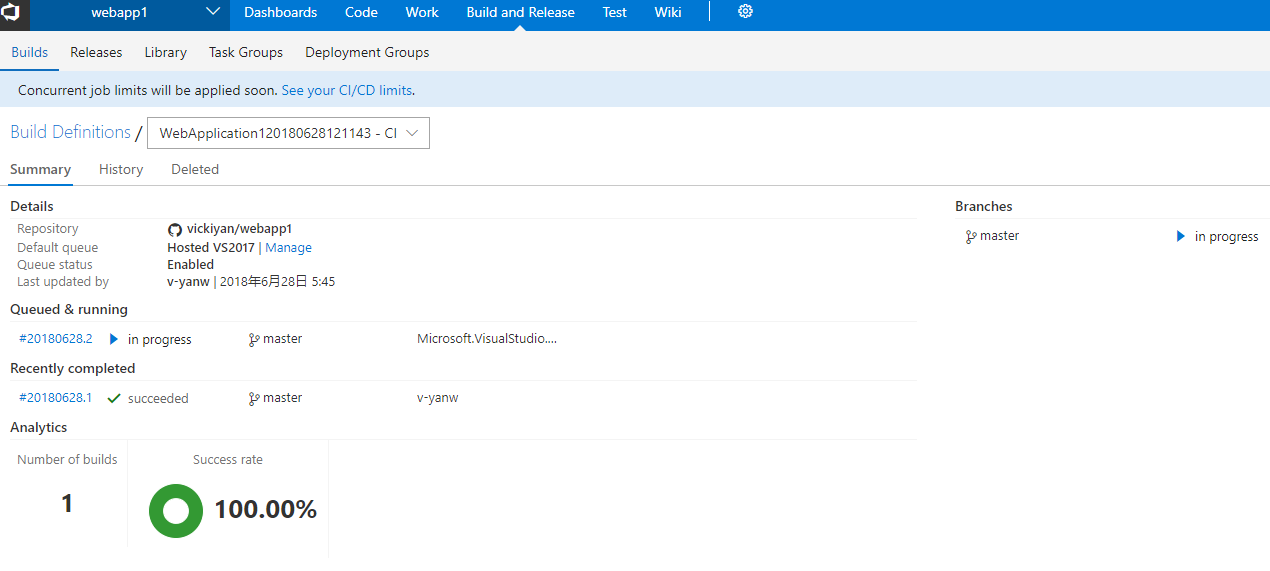


1. 显示正在Push，直到Push成功

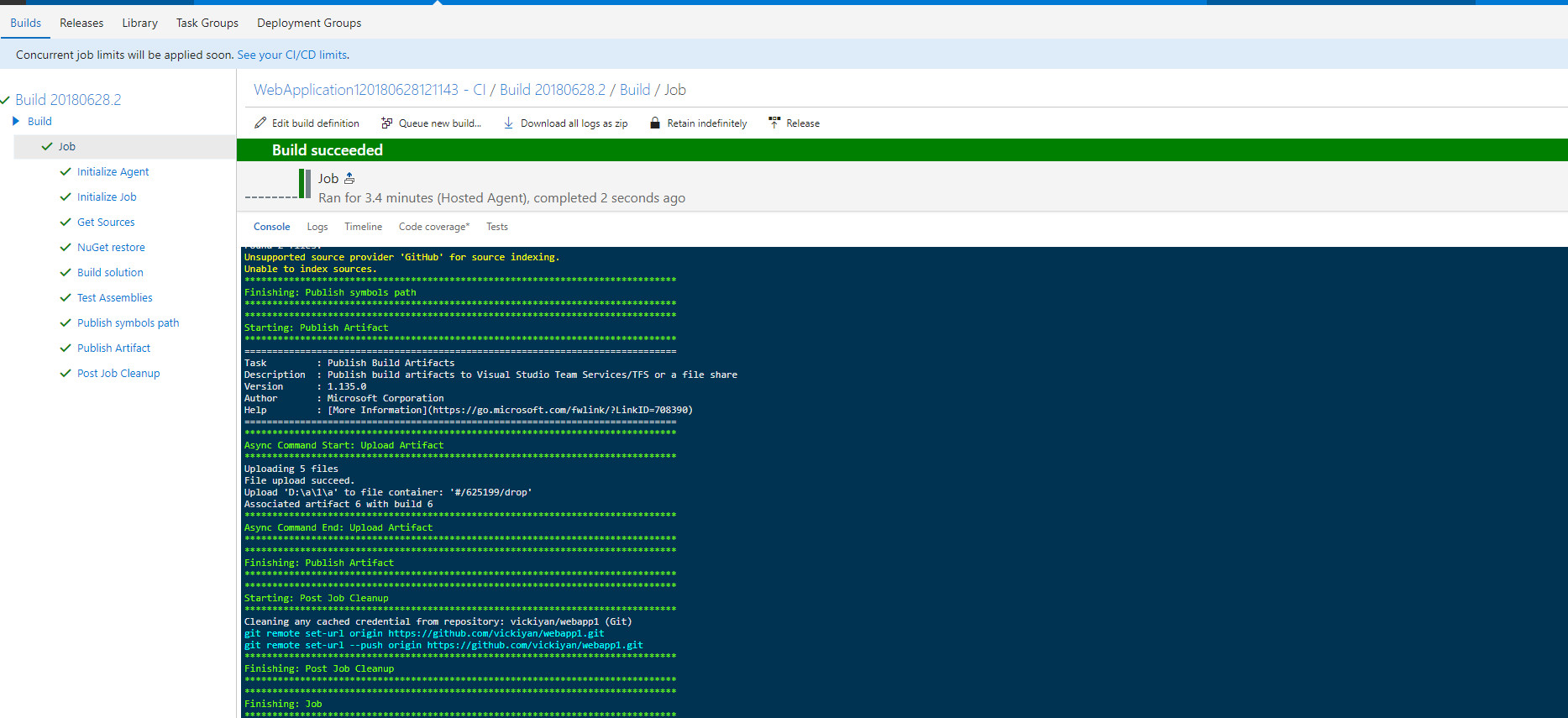
 

#### Task 2: 验证CI/CD

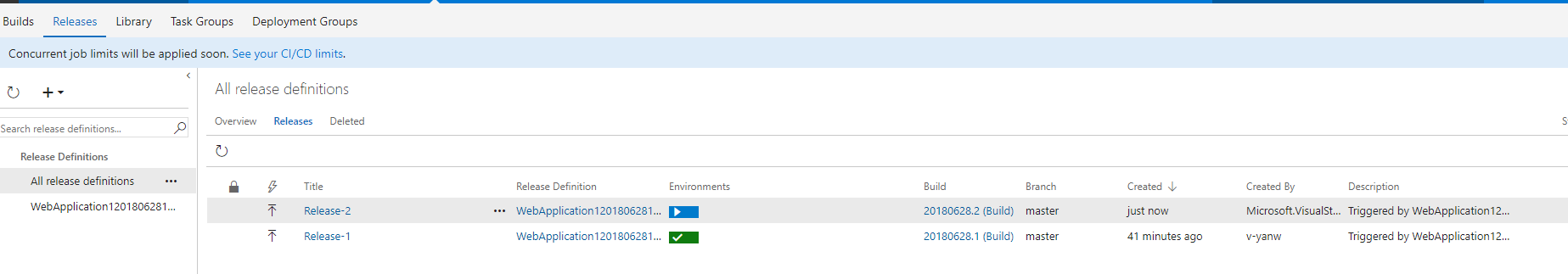
1. 在浏览器中打开VSTS门户，打开webapp1项目的Build Definition界面。可以看到一个新的Build正在进行，说明持续集成已经由代码提交被自动触发。



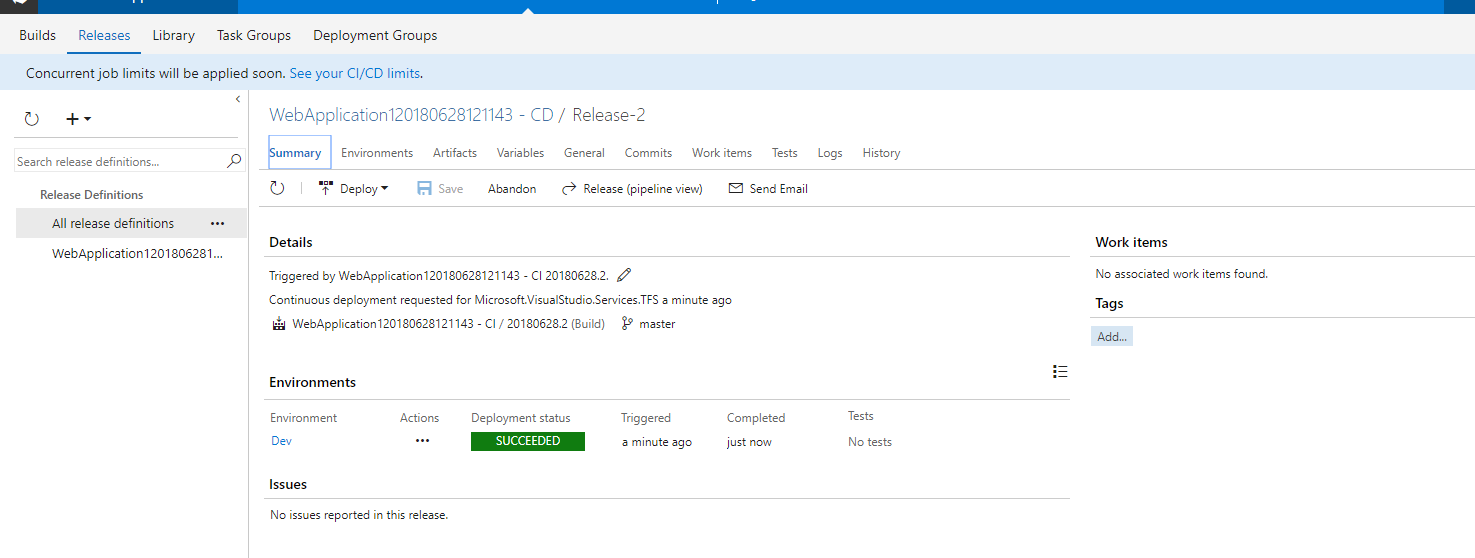
1. 点击Build作业，可以查看日志信息:



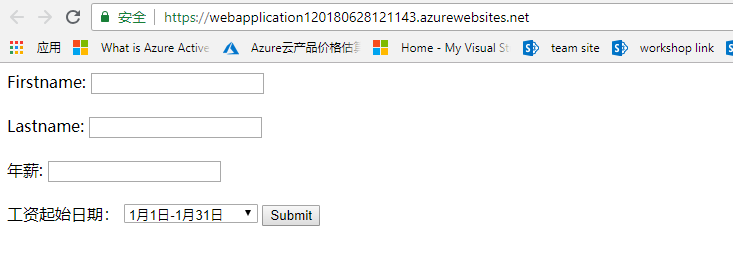
1. Build成功后，点击Release菜单，可以看到创建一个新的Release 作业，表明build成功后，持续部署会自动被触发。



1. 点击Release作业，可以看到部署已成功



1. 刷新或者重新打开网站，可以看到名字和姓氏已经由中文改成英文显示。



## 小结

* 列出您在编写这个程序时候所作的假设：
  + 名字和姓氏为字符类型；年薪为数字类型；工资起始日期为12个固定区间选择。
* 编写程序（编程语言不限，技术栈不限）
  + 该示例利用Visual Studio 2017 作为编程环境，基于C#语言和ASP.Net Razor 框架编写，通过一个CSHTML页面完成代码逻辑和展现，为简便起见，没有用到数据库存储数据。
* 编写文档来说明如何运行您的程序
  + 请参考本文档的第一章《编写工资计算器应用》和第二章《利用GitHub和VSTS配置CI/CD》。
* 编写测试方案来说明如何验证您的程序
  + 请参考本文档第三章《项目测试》。
* 编写把你的程序部署到azure上的CICD pipeline方案（需要包含infra as code，技术栈不限 ）
  + 利用GitHub作为源代码管理工具，VSTS作为CI/CD 的Pipeline引擎，通过Visual Studio 2017自带的发布应用向导轻松完成CI/CD设置。
* 文档和代码需要上传至github
  + GitHub地址为https://github.com/vickiyan/salarycalculator