

实验：通过 Azure IoT Hub Device Twins 监控 IoT DevKit 运行状态

课程目的

本实验将介绍如何使用 IoT DevKit 开发套件连接到 Azure IoT Hub，并通过 Azure IoT Hub Device Twins 监控 IoT DevKit 运行状态，及控制 IoT DevKit User LED。

步骤

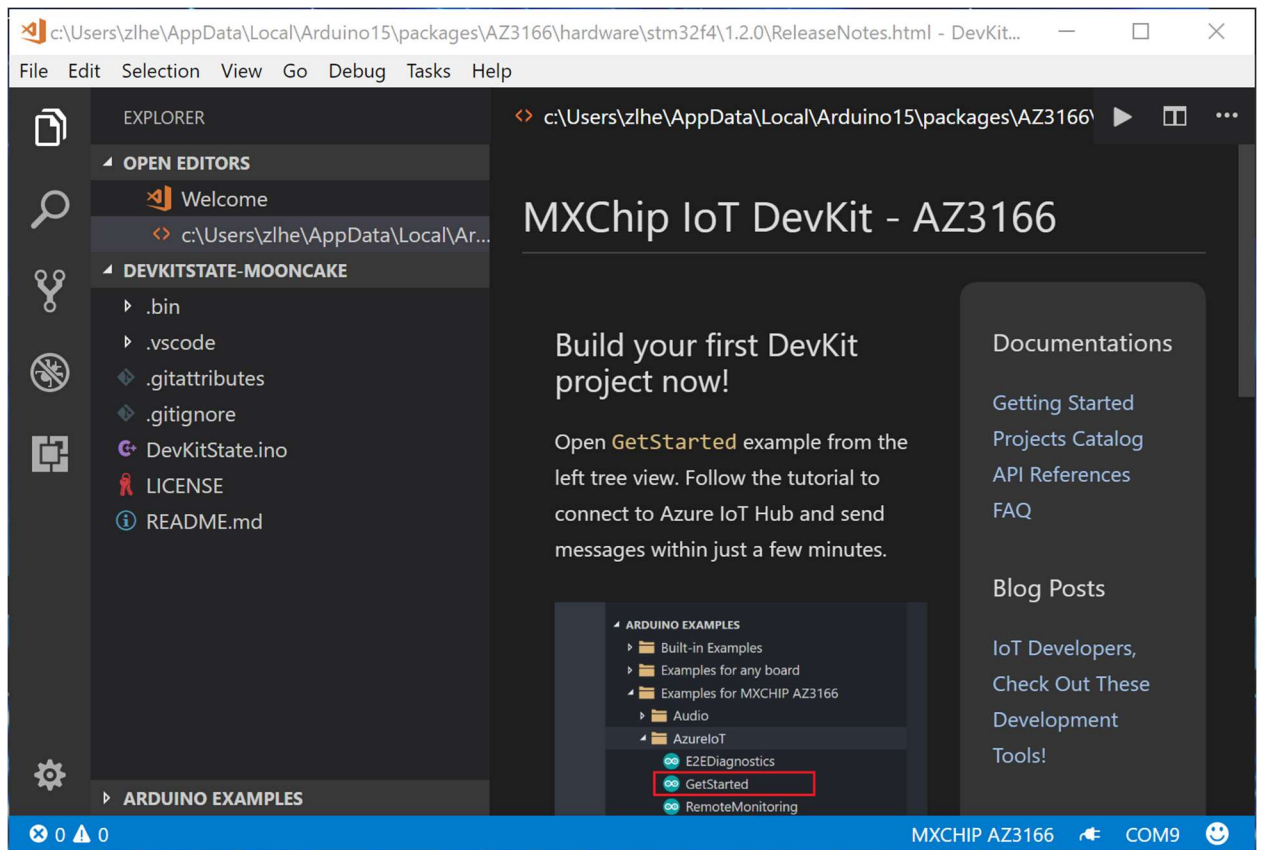
一．下载并打开例程

1. 访问 <https://github.com/DevKitExamples/DevKitState/tree/mooncake>
2. 点击 “Clone or download” 选择 “Download ZIP”
3. 在本地解压缩文件
4. 在命令行中打开解压目录
`cd DevKitState-mooncake`

二．打开 VS Code 连接 IoT DevKit

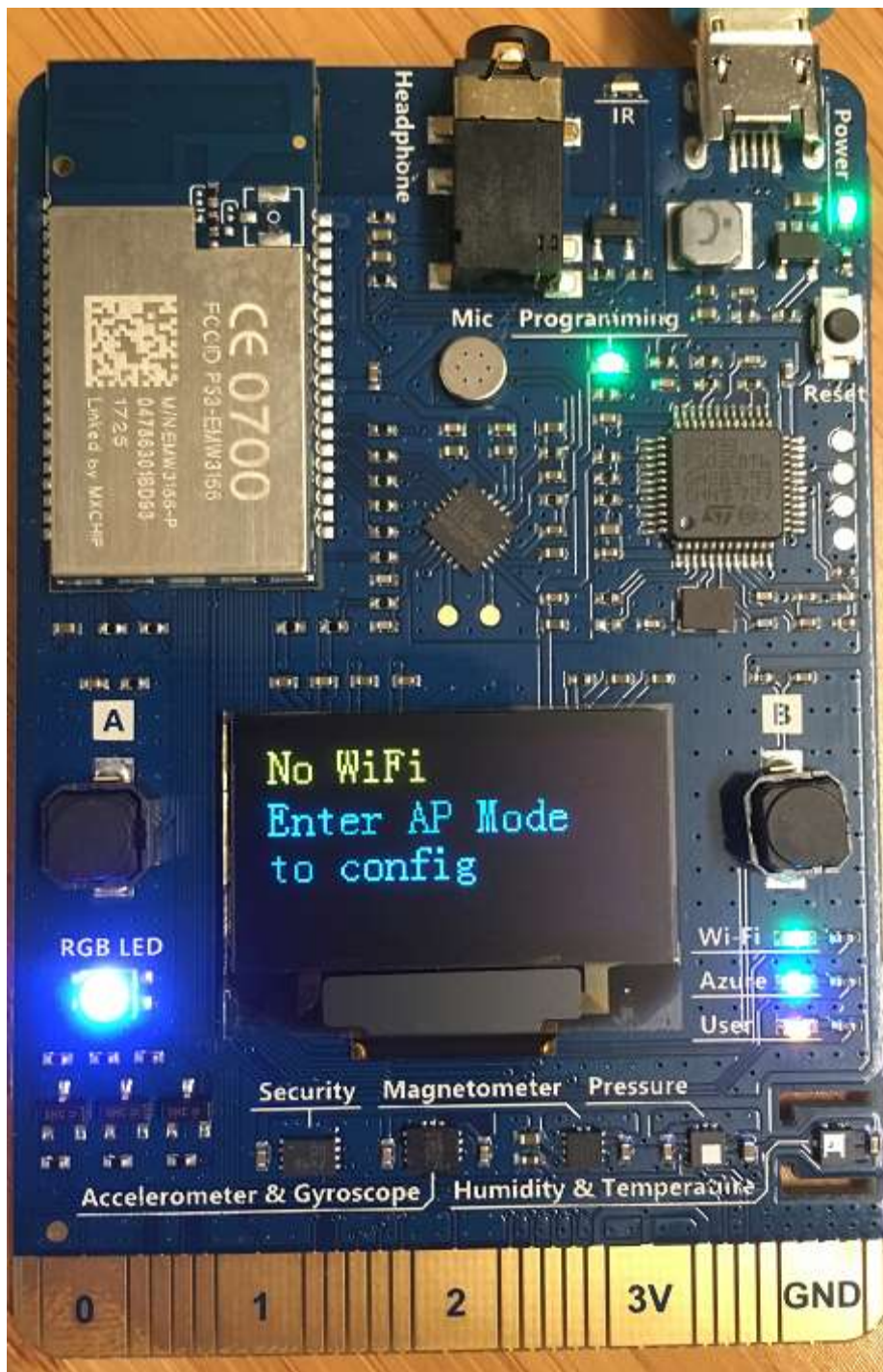
1. 使用 Visual Studio Code 打开当前目录：
`code .`

2. 连接 IoT DevKit 到计算机



三 . 配置 IoT DevKit 连接 WiFi

1. 连接 IoT DevKit 到电脑后，DevKit 屏幕上会先开始自检并尝试连接网络。如果无法连接一个有效的 Wi-Fi 网络，开发版的屏幕上将会显示：



2. 此时你需要将 DevKit 切换到设置模式以配置并连接 Wi-Fi。方法为：首先按住 A 键，然后保持按住并同时按一下 Reset 键后放开 Reset 键。最后再放开 A 键。如果成功，屏幕上将会显示“ Configuration” 字样：

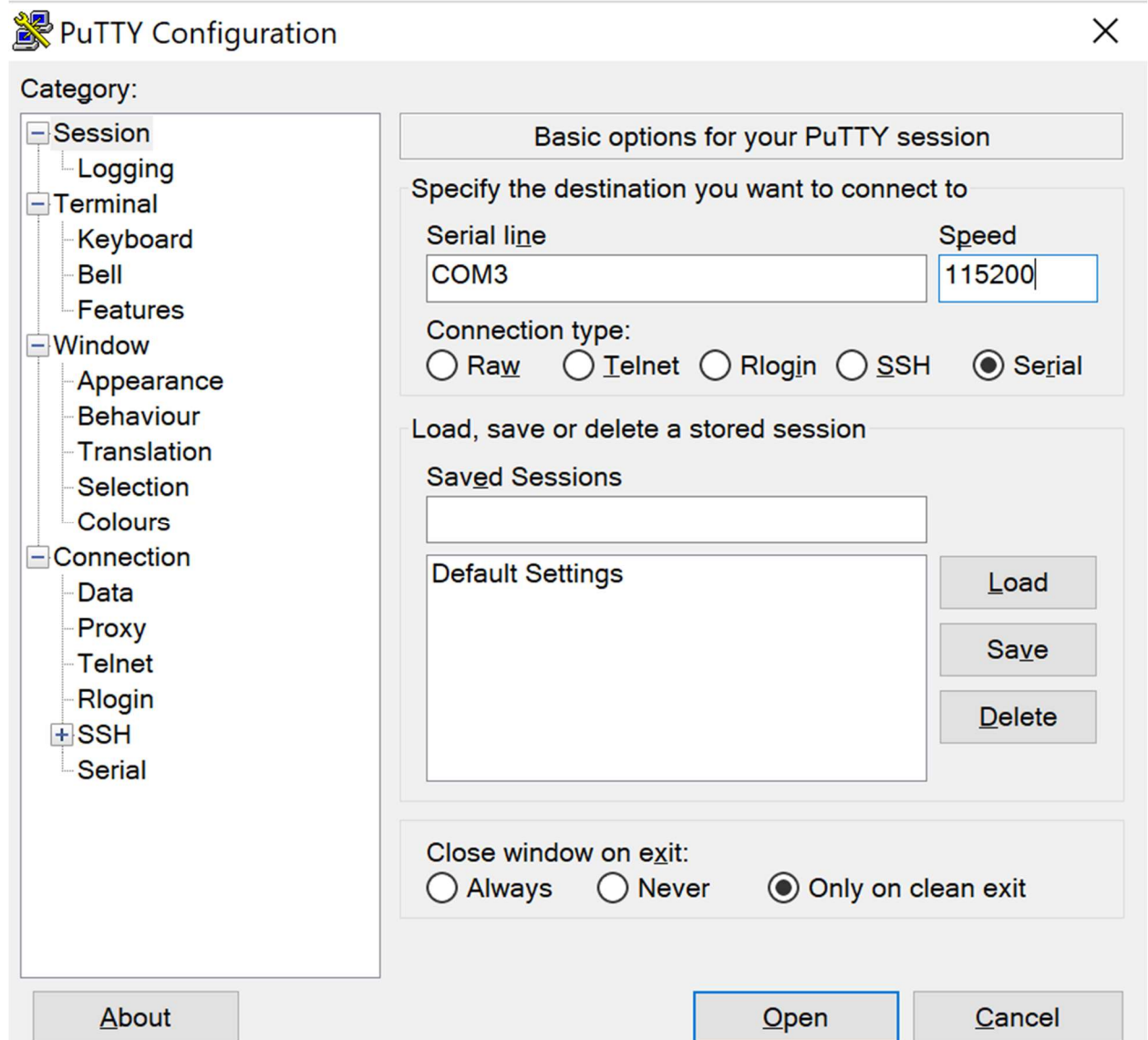


3. 然后在电脑的桌面上找到 Putty 快捷方式，打开程序。
4. 在登录窗口中输入如下配置以进入 DevKit 配置命令行窗口：

Connection Type: **Serial**

Serial line: **COM?** (请依据 VS Code 窗口右下角显示 DevKit 所占用的串口号输入)

Speed: **115200**



The image shows the PuTTY Configuration dialog box. On the left is a tree view under 'Category:' with sub-items: Session, Logging, Terminal, Keyboard, Bell, Features, Window, Appearance, Behaviour, Translation, Selection, Colours, Connection, Data, Proxy, Telnet, Rlogin, SSH, and Serial. The 'Serial' item under 'Connection' is selected. The main area is titled 'Basic options for your PuTTY session'. It contains a section 'Specify the destination you want to connect to' with 'Serial line' set to 'COM3' and 'Speed' set to '115200'. Below this, 'Connection type:' has radio buttons for Raw, Telnet, Rlogin, SSH, and Serial (which is selected). Another section 'Load, save or delete a stored session' has a 'Saved Sessions' list (empty) and a 'Default Settings' list (empty), with 'Load', 'Save', and 'Delete' buttons to the right. At the bottom, 'Close window on exit:' has radio buttons for Always, Never, and Only on clean exit (which is selected). At the very bottom are 'About', 'Open', and 'Cancel' buttons.

Category:

- Session
 - Logging
- Terminal
 - Keyboard
 - Bell
 - Features
- Window
 - Appearance
 - Behaviour
 - Translation
 - Selection
 - Colours
- Connection
 - Data
 - Proxy
 - Telnet
 - Rlogin
 - SSH
 - Serial

Basic options for your PuTTY session

Specify the destination you want to connect to

Serial line: COM3 Speed: 115200

Connection type:
☐ Raw ☐ Telnet ☐ Rlogin ☐ SSH ☒ Serial

Load, save or delete a stored session

Saved Sessions

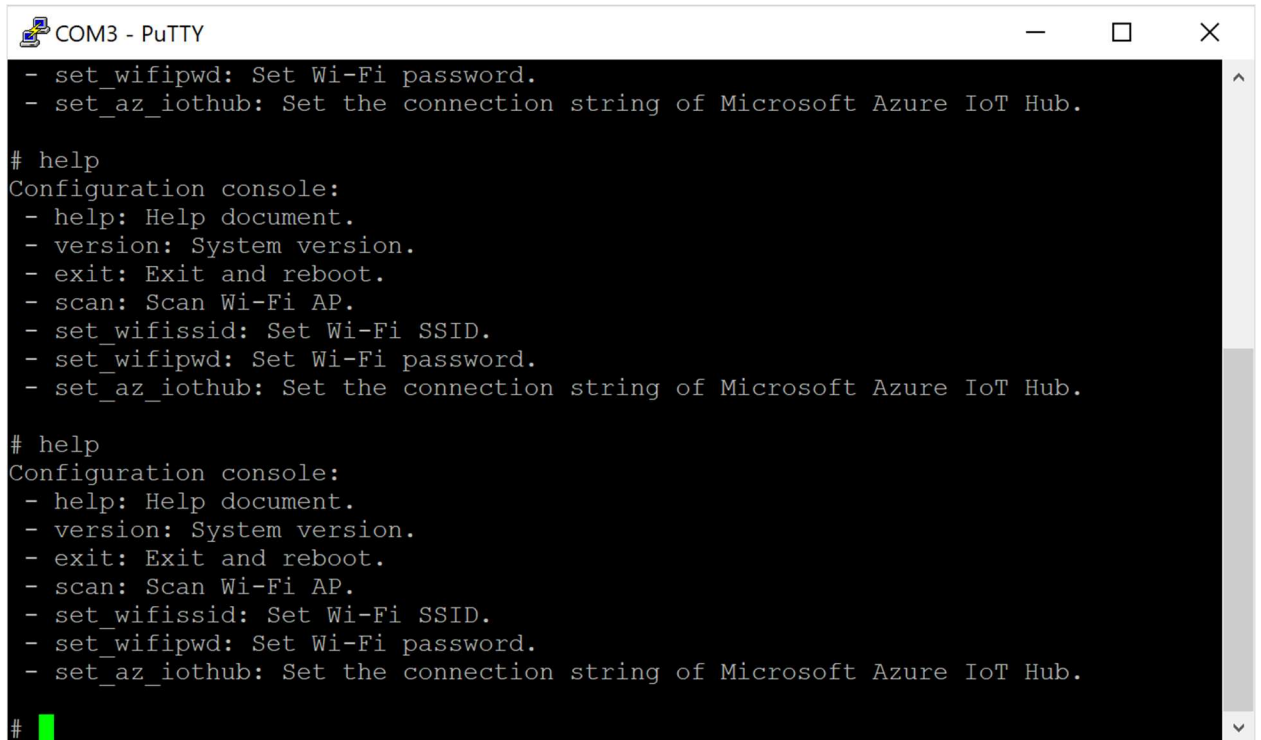
Default Settings

Load Save Delete

Close window on exit:
☐ Always ☐ Never ☒ Only on clean exit

About Open Cancel

5. 成功登录进入 DevKit 后，输入“ help” ，命令行窗口会出现：



```
COM3 - PuTTY
- set_wifipwd: Set Wi-Fi password.
- set_az_iotHub: Set the connection string of Microsoft Azure IoT Hub.

# help
Configuration console:
- help: Help document.
- version: System version.
- exit: Exit and reboot.
- scan: Scan Wi-Fi AP.
- set_wifissid: Set Wi-Fi SSID.
- set_wifipwd: Set Wi-Fi password.
- set_az_iotHub: Set the connection string of Microsoft Azure IoT Hub.

# help
Configuration console:
- help: Help document.
- version: System version.
- exit: Exit and reboot.
- scan: Scan Wi-Fi AP.
- set_wifissid: Set Wi-Fi SSID.
- set_wifipwd: Set Wi-Fi password.
- set_az_iotHub: Set the connection string of Microsoft Azure IoT Hub.

#
```

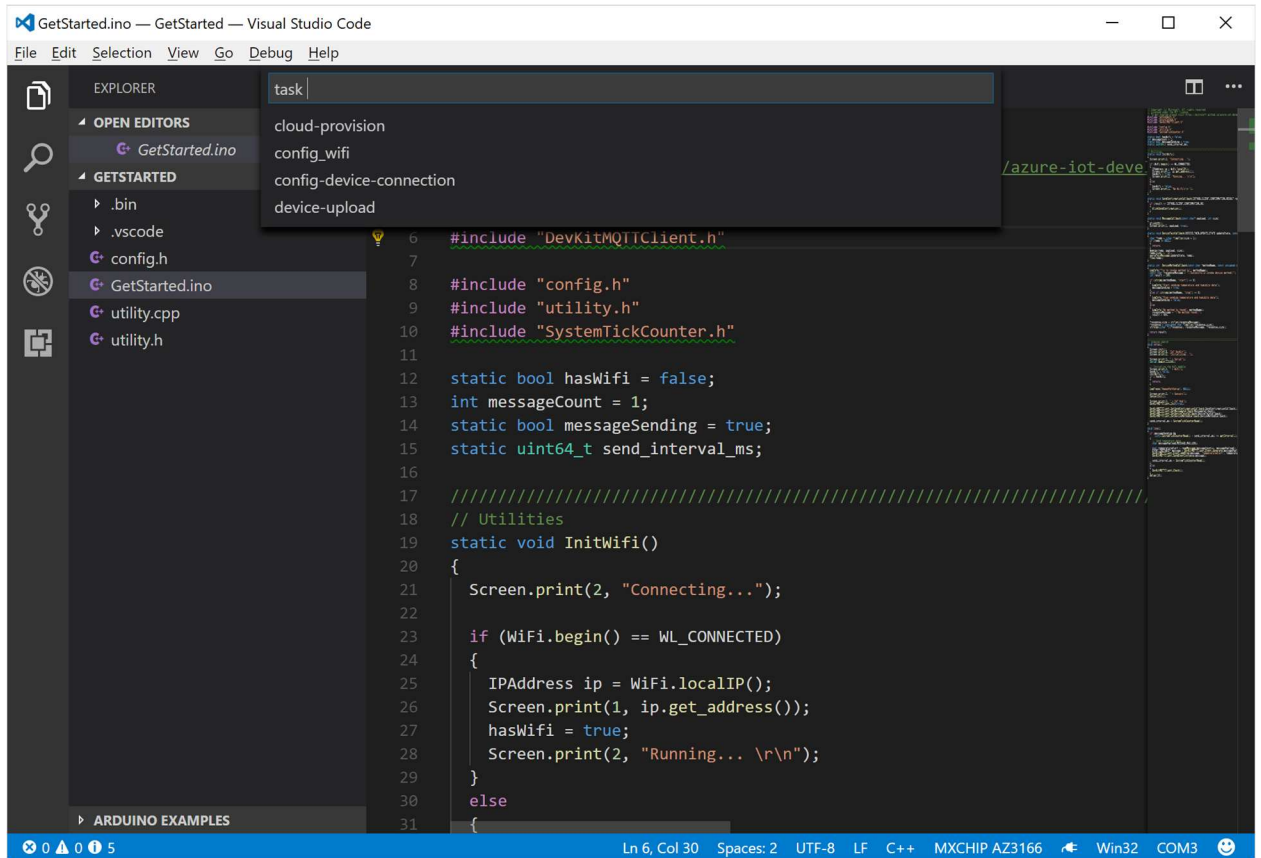
6. 按次序输入如下命令以连接我们会场提供的 Wi-Fi :

```
set_wifissid XXXXX
set_wifipwd XXXXX
exit
```

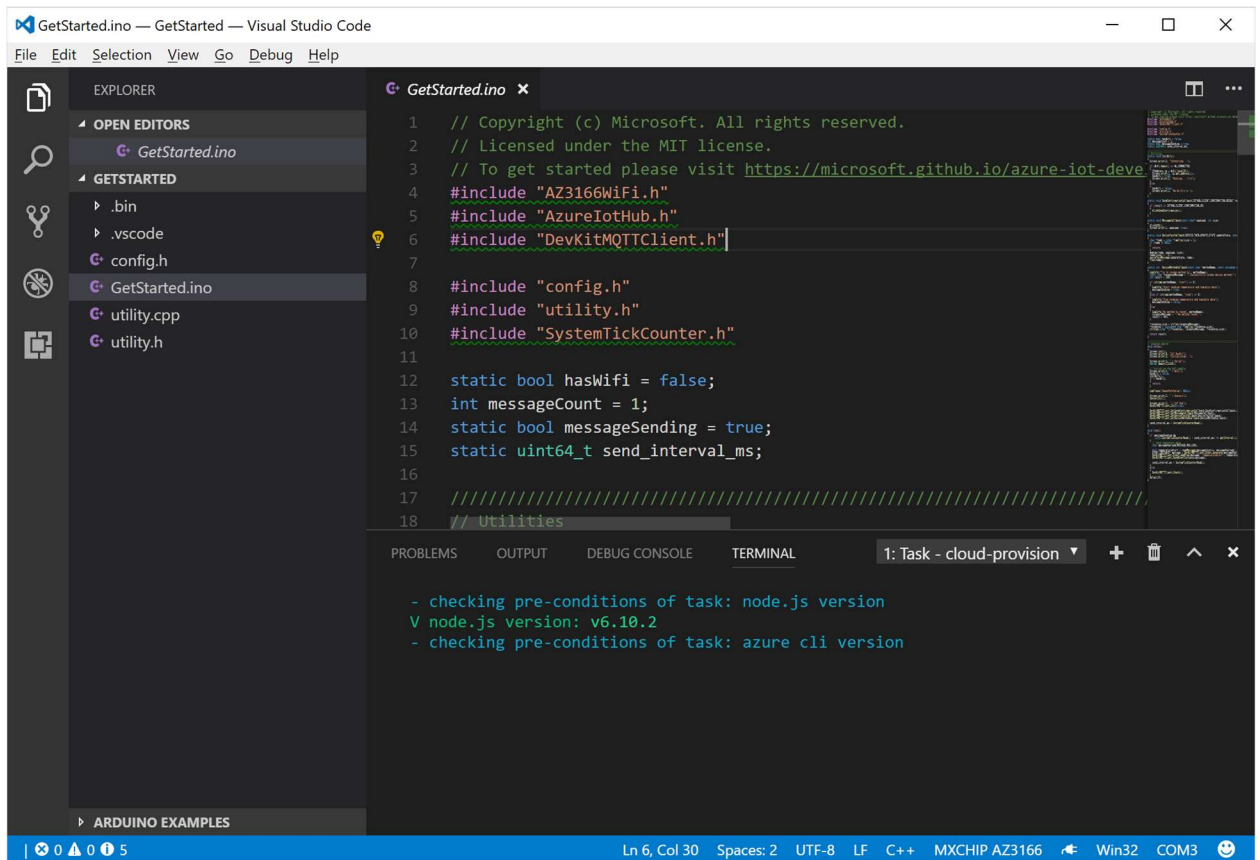
7. 配置成功后请关闭 Putty。DevKit 会自动重启并尝试连接 Wi-Fi。若连接成功在屏幕上将会看到 IP 地址。

四 . 登录 Azure

1. 在菜单中找到“ Go -> Go to file” 或者按“ Ctrl+P” 打开快捷菜单。然后输入“ task” , VS Code 会自动提示可以使用的命令 :



2. 选择“task cloud-provision”，VS Code 会打开命令行窗口。我们接下去会在该窗口用交互的方式完成 Azure 的登录和这个例子所需要的 Azure 各种 IoT 服务的创建工作：



3. 根据提示，首先登录到 Azure。VS Code 会产生一个如 “GZXYPTZVY” 样式的登录码，然后自动打开浏览器。请将登录码复制并粘贴到浏览器输入框中：

GetStarted.ino — GetStarted — Visual Studio Code

File Edit Selection View Go Debug Help

EXPLORER

OPEN EDITORS

- GetStarted.ino

GETSTARTED

- .bin
- .vscode
- config.h
- GetStarted.ino
- utility.cpp
- utility.h

GetStarted.ino

```
1 // Copyright (c) Microsoft. All rights reserved.
2 // Licensed under the MIT license.
3 // To get started please visit https://microsoft.github.io/azure-iot-developer-get-started
4 #include "AZ3166WiFi.h"
5 #include "AzureIoTHub.h"
6 #include "DevKitMQTTClient.h"
7
8 #include "config.h"
9 #include "utility.h"
10 #include "SystemTickCounter.h"
11
12 static bool hasWifi = false;
13 int messageCount = 1;
14 static bool messageSending = true;
15 static uint64_t send_interval_ms;
16
17 // Utilities
18
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: Task - cloud-provision

```
- checking pre-conditions of task: node.js version
V node.js version: v6.10.2
- checking pre-conditions of task: azure cli version
V azure cli version: 2.0.9
- checking pre-conditions of task: subscription
C:\Program Files (x86)\Microsoft SDKs\Azure\CLI2\wbin\az.cmd login
Please input the following code: G7T5B8TY4
[]
```

ARDUINO EXAMPLES

Ln 6, Col 30 Spaces: 2 UTF-8 LF C++ MXCHIP AZ3166 Win32 COM3

Sign in to your account

https://login.microsoftonline.com/common/oauth2/deviceauth

Device Login

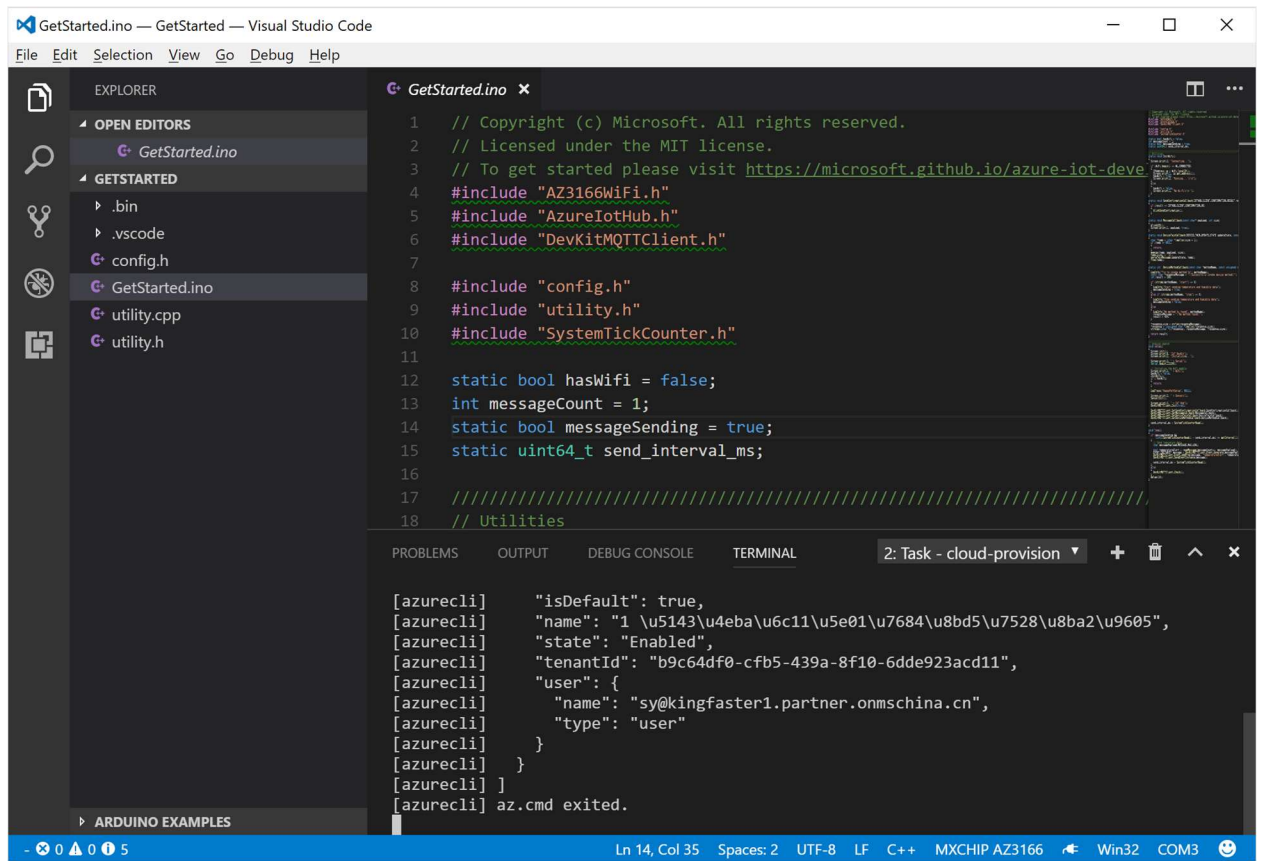
Enter the code that you received from the application on your device

Microsoft Azure Cross-platform Command Line Interface

Application publisher:

Click Cancel if this isn't the application you were trying to sign in to on your device.

4. 然后请输入实验课上我们向你提供的 Azure 试用帐号和密码完成登录。登录完毕后，根据提示，你可以关闭浏览器窗口回到 VS Code 中。
5. 随后 VS Code 中会显示该账户的一些详细信息，并提示你已经成功登录：



The screenshot shows the Visual Studio Code interface with the 'GetStarted.ino' file open in the editor. The file contains C++ code for an Azure IoT Hub connection. The Explorer pane on the left shows the project structure, including 'bin', '.vscode', 'config.h', 'GetStarted.ino', 'utility.cpp', and 'utility.h'. The Terminal pane at the bottom shows the output of the 'azurecli' command, which displays the user's profile information and confirms that the 'az' command has exited successfully.

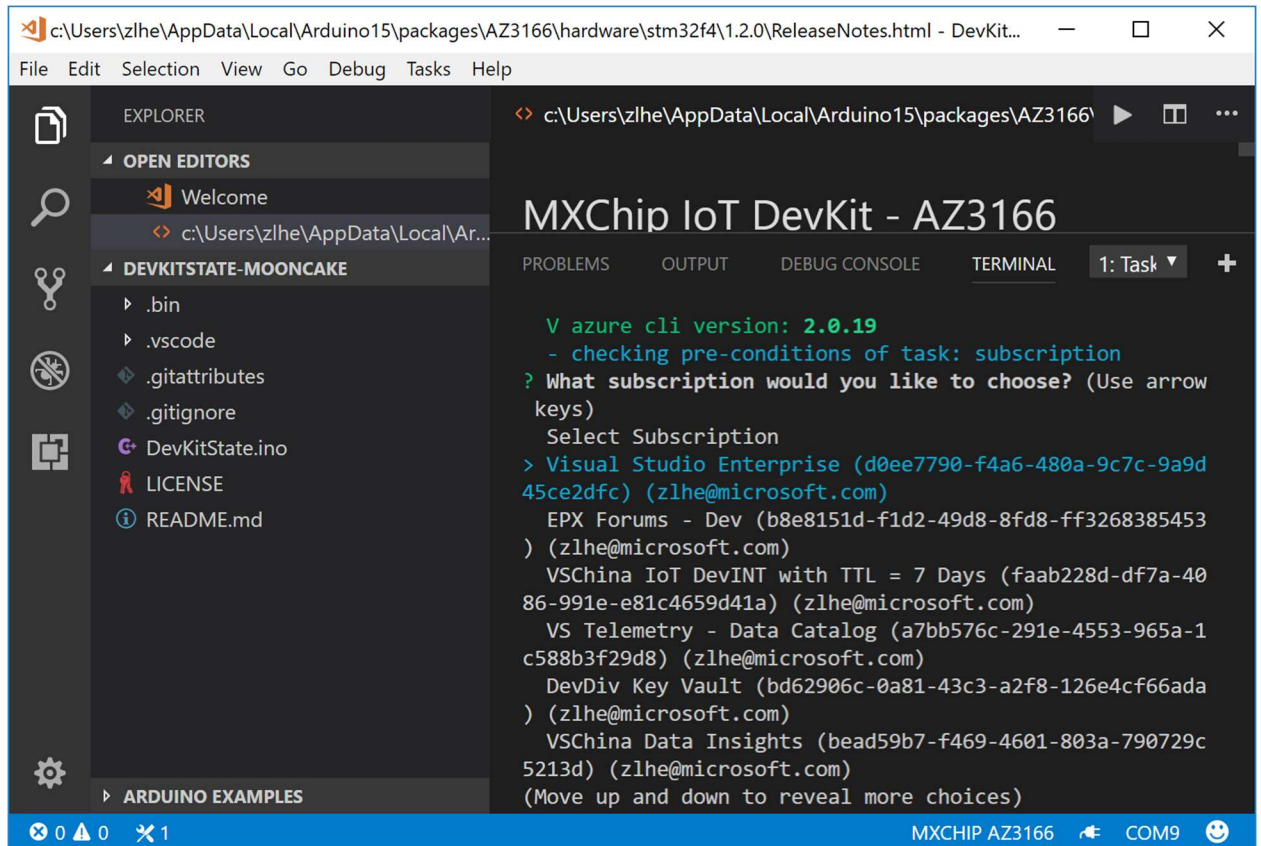
```
1 // Copyright (c) Microsoft. All rights reserved.
2 // Licensed under the MIT license.
3 // To get started please visit https://microsoft.github.io/azure-iot-developer-toolkit/
4 #include "AZ3166WiFi.h"
5 #include "AzureIotHub.h"
6 #include "DevKitMQTTClient.h"
7
8 #include "config.h"
9 #include "utility.h"
10 #include "SystemTickCounter.h"
11
12 static bool hasWifi = false;
13 int messageCount = 1;
14 static bool messageSending = true;
15 static uint64_t send_interval_ms;
16
17 ////////////////////////////////////////////////////
18 // Utilities
```

```
[azurecli] "isDefault": true,
[azurecli] "name": "1 \u5143\u4eba\u6c11\u5e01\u7684\u8bd5\u7528\u8ba2\u9605",
[azurecli] "state": "Enabled",
[azurecli] "tenantId": "b9c64df0-cfb5-439a-8f10-6dde923acd11",
[azurecli] "user": {
[azurecli]   "name": "sy@kingfaster1.partner.onmschina.cn",
[azurecli]   "type": "user"
[azurecli] }
[azurecli] ]
[azurecli] az.cmd exited.
```

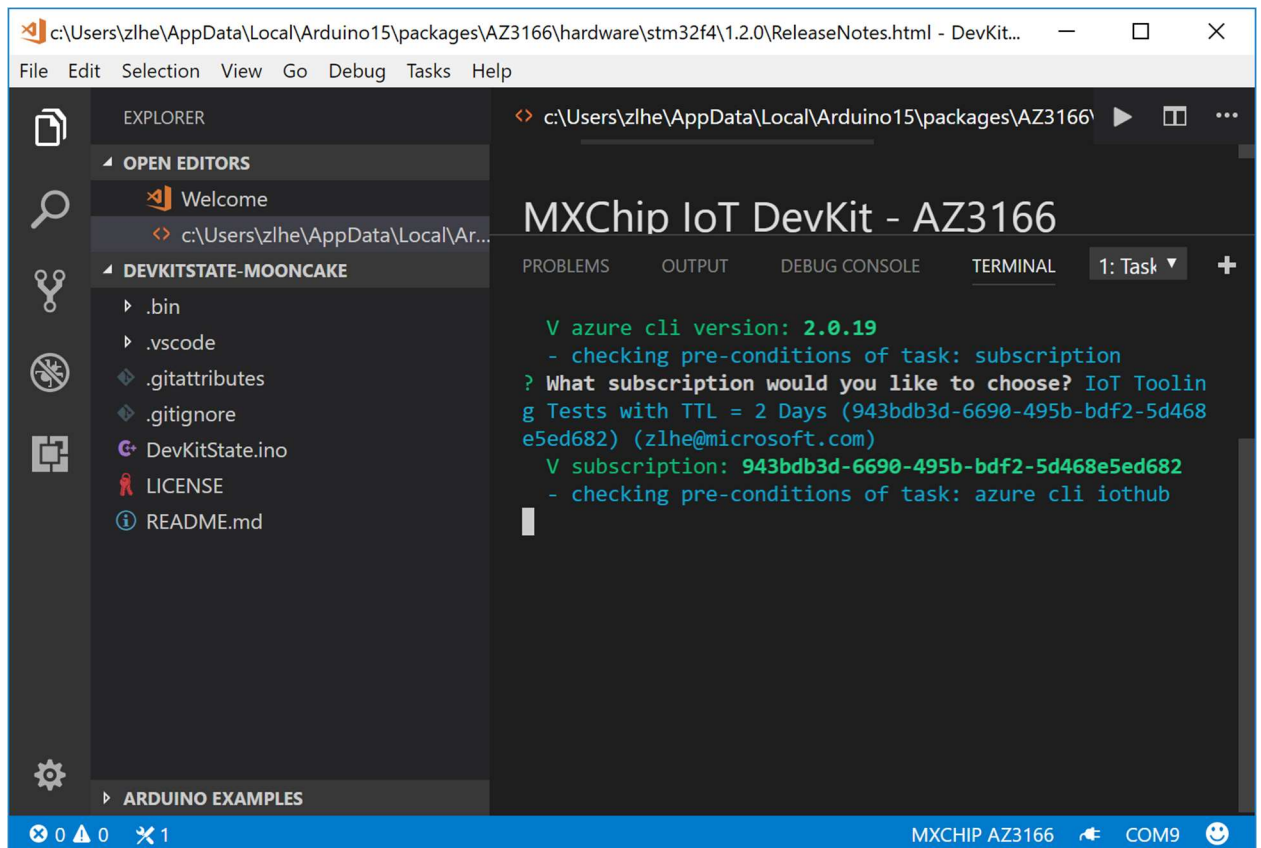
成功登录后请不要立即关闭命令行窗口。

五．配置 Azure 服务

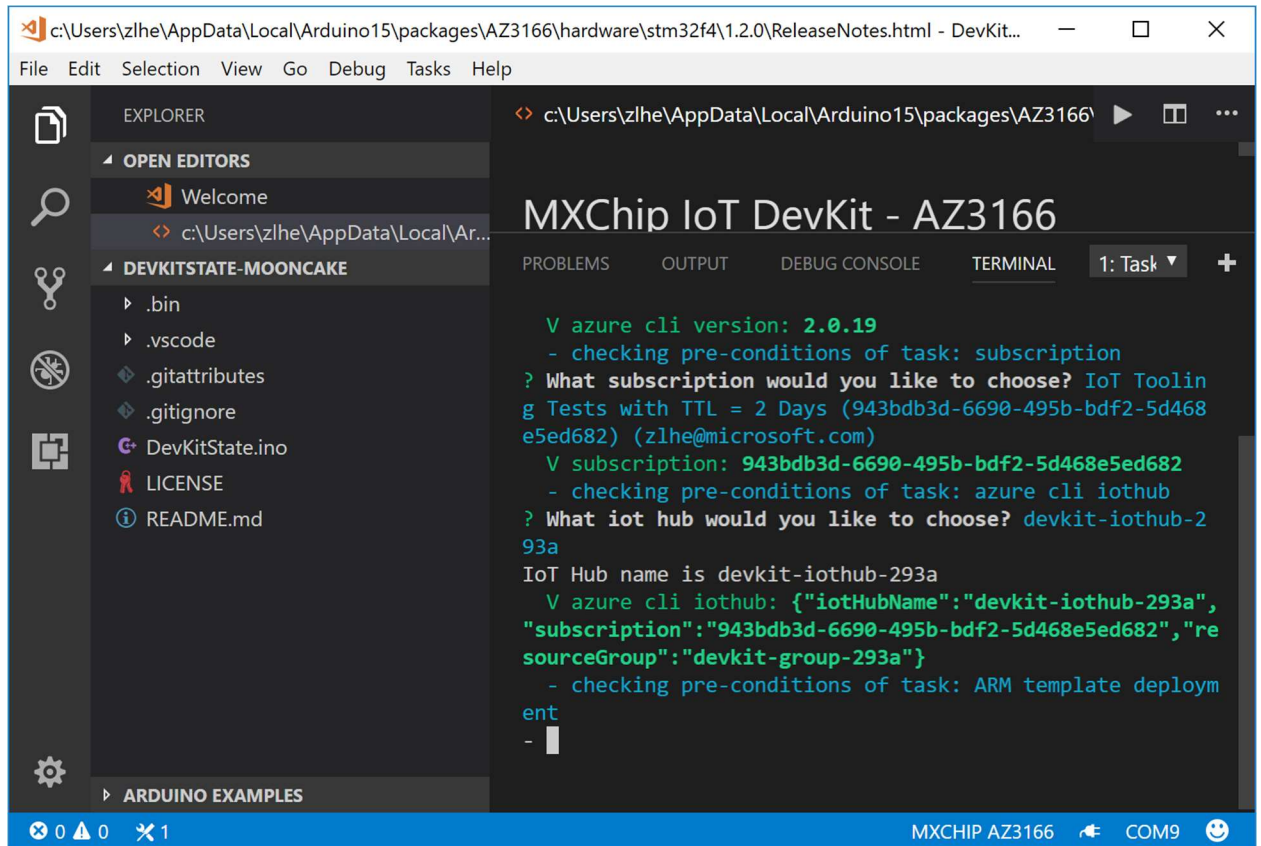
1. 点击 Visual Studio Code 的 Tasks 菜单，选择 Run Task...，运行 cloud-provision 任务



2. 使用上下方向键选择订阅，按回车键确定



3. 选择或者新建一个 IoT Hub，然后等待 Azure 创建服务。新建 IoT Hub 时需要先创建一个资源组，根据提示填写资源组名称或者选择已有的资源组即可

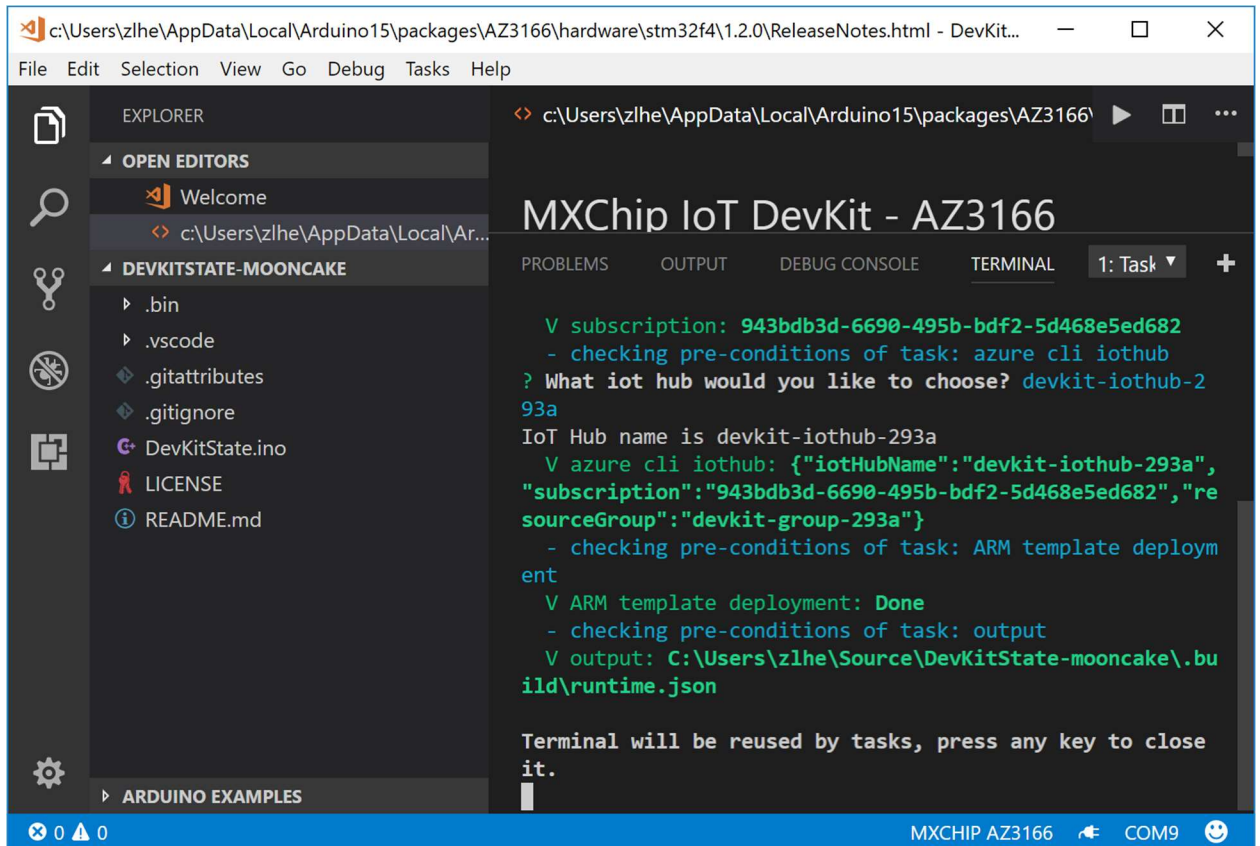


The screenshot shows the MXChip IoT DevKit - AZ3166 IDE interface. The left sidebar contains the Explorer, Open Editors, and DevKitState-MOONCAKE panels. The main editor area displays the terminal output of the following commands:

```
V azure cli version: 2.0.19
- checking pre-conditions of task: subscription
? What subscription would you like to choose? IoT Toolin
g Tests with TTL = 2 Days (943bdb3d-6690-495b-bdf2-5d468
e5ed682) (zlh@microsoft.com)
V subscription: 943bdb3d-6690-495b-bdf2-5d468e5ed682
- checking pre-conditions of task: azure cli iotHub
? What iot hub would you like to choose? devkit-iotHub-2
93a
IoT Hub name is devkit-iotHub-293a
V azure cli iotHub: {"iotHubName":"devkit-iotHub-293a",
"subscription":"943bdb3d-6690-495b-bdf2-5d468e5ed682","re
sourceGroup":"devkit-group-293a"}
- checking pre-conditions of task: ARM template deploym
ent
-
```

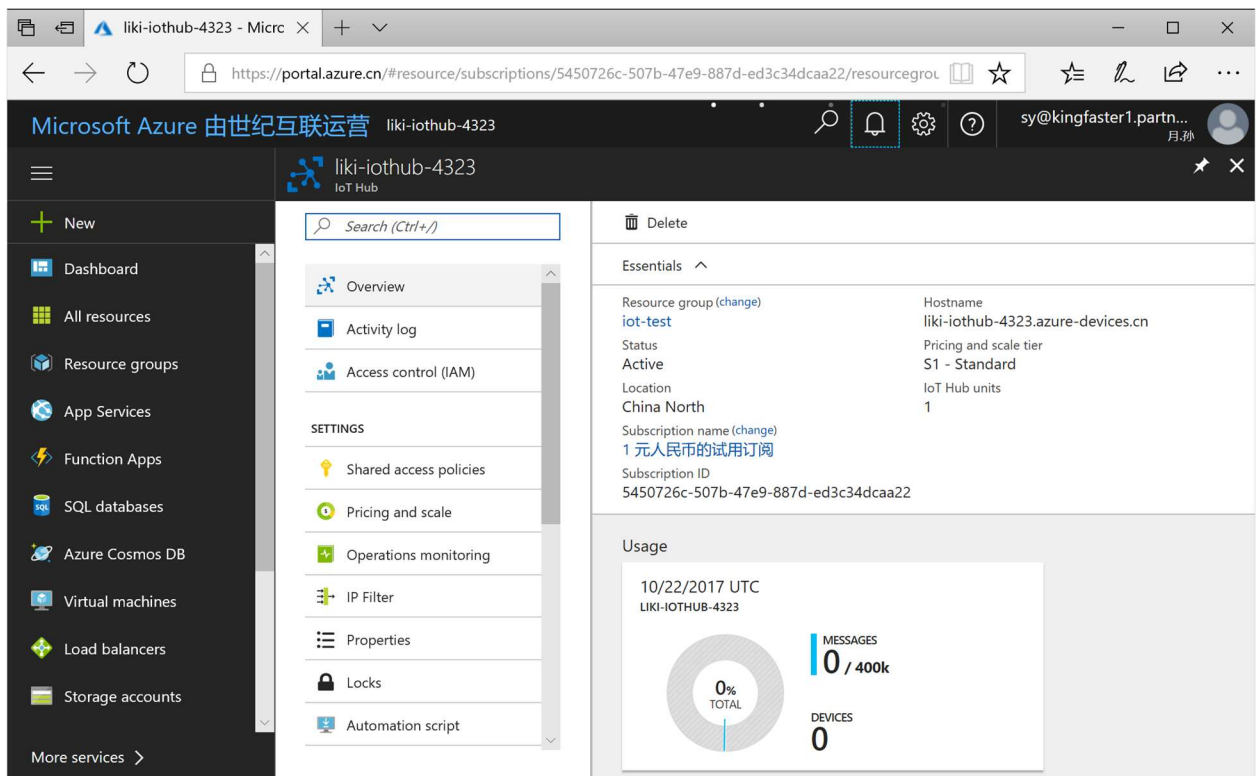
The status bar at the bottom indicates the device is MXCHIP AZ3166, connected to COM9, and shows 0 errors, 0 warnings, and 1 task.

4. Azure 服务创建完成后按任意键结束

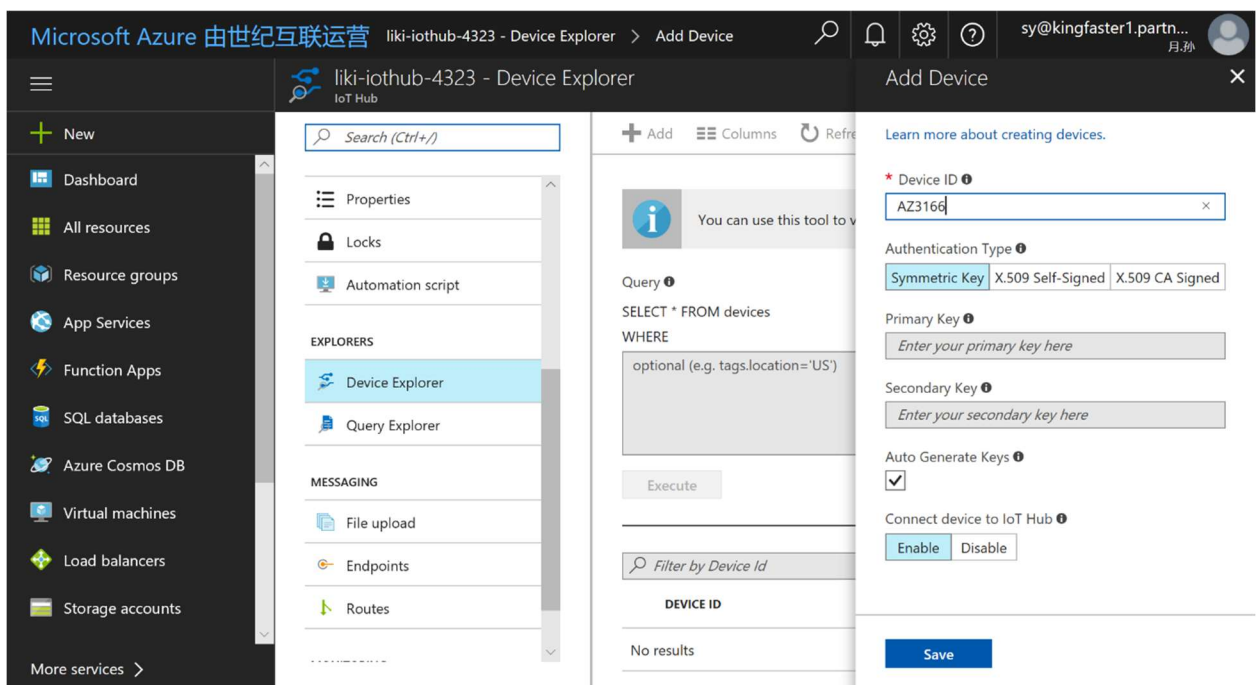


六 . 设置 Azure device connection string

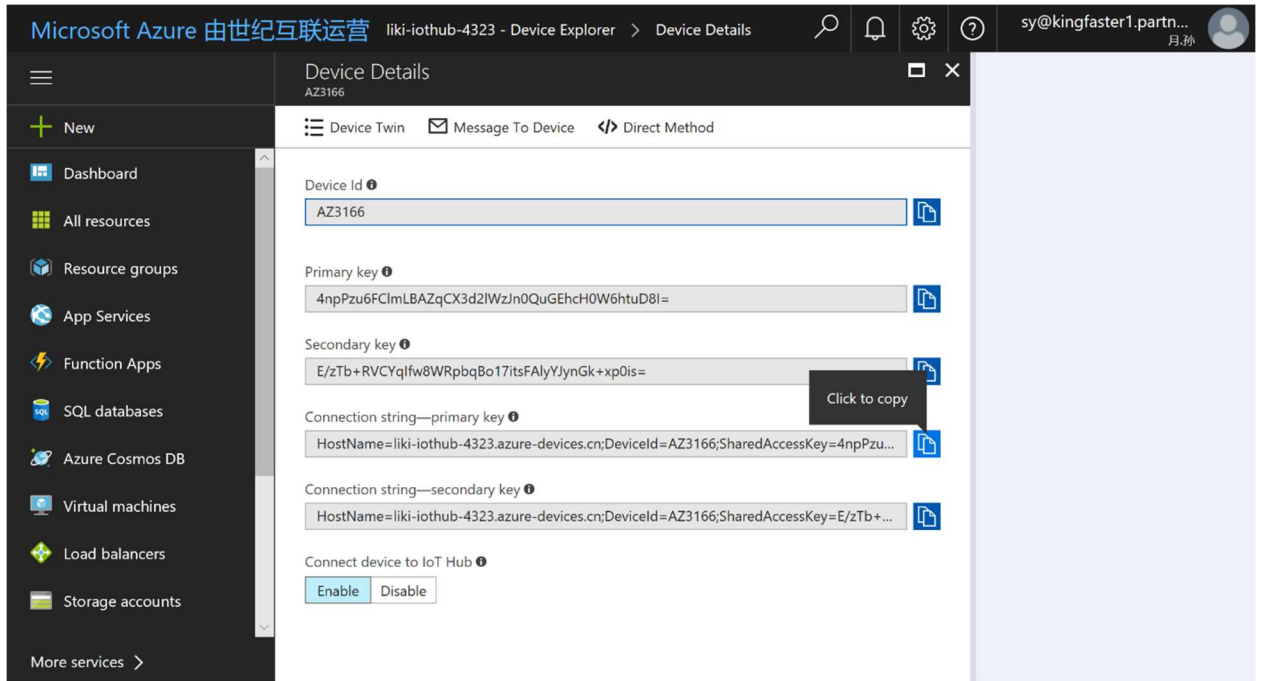
1. 打开浏览器并输入：<https://portal.azure.cn>。
2. 此时浏览器应该能够自动进入 Azure 管理界面。如果仍需登录，请输入实验课上我们向你提供的 Azure 试用帐号和密码。
3. 找到并进入我们刚才创建的 Azure IoT Hub：



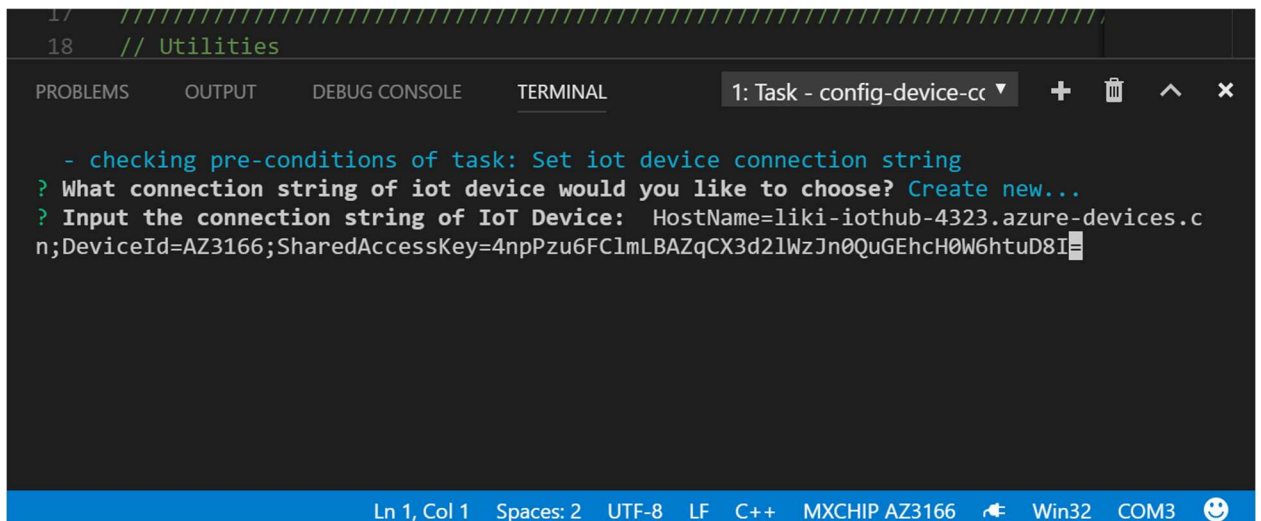
4. 选择“ Device Explorer” ，然后选择“ Add” ，然后将设备名字命名为“ **AZ3166**” （请确保该名字输入无误）。安全类型为“ Symmetric key” ，然后点 Save 创建：



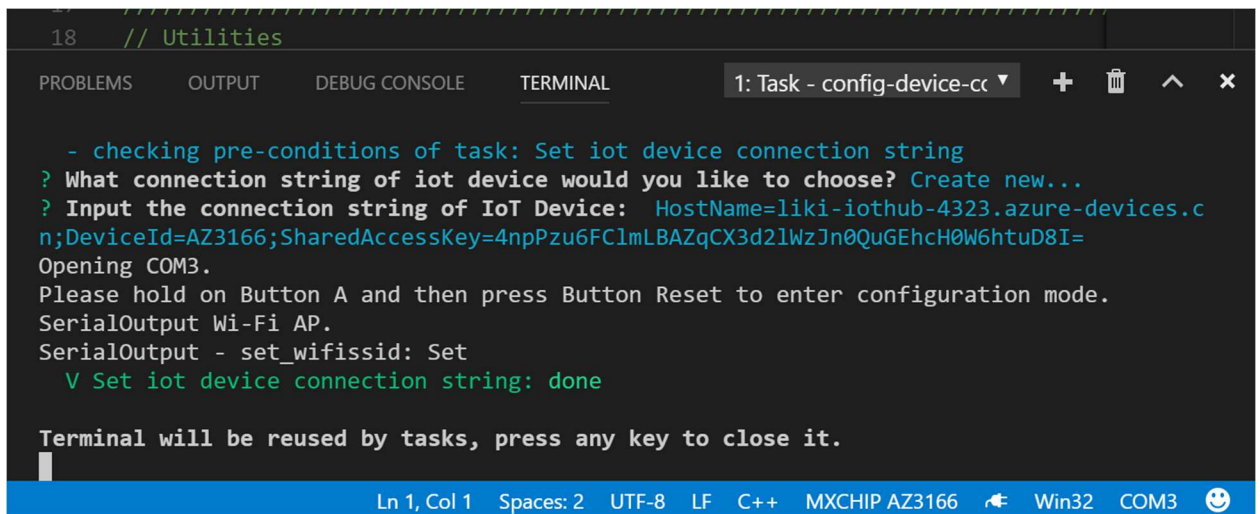
5. 然后选择该设备，找到“ Connection string – primary key” 点击右侧图标复制设备连接字符串：



6. 回到 VS Code 中。在菜单中找到“Go -> Go to file”或者按“Ctrl+P”打开快捷菜单。然后输入“task”，选择“task config-device-connection”，VS Code 会打开命令行窗口。然后选择“Create New...” ，并在提示下粘贴刚才复制的设备连接字符串：



7. 随后根据提示再次进入 DevKit 设置模式（首先按住 A 键，然后保持按住并同时按一下 Reset 键后放开 Reset 键。最后再放开 A 键。），会自动将 Connection string 写入 DevKit 中：



```
18 // Utilities

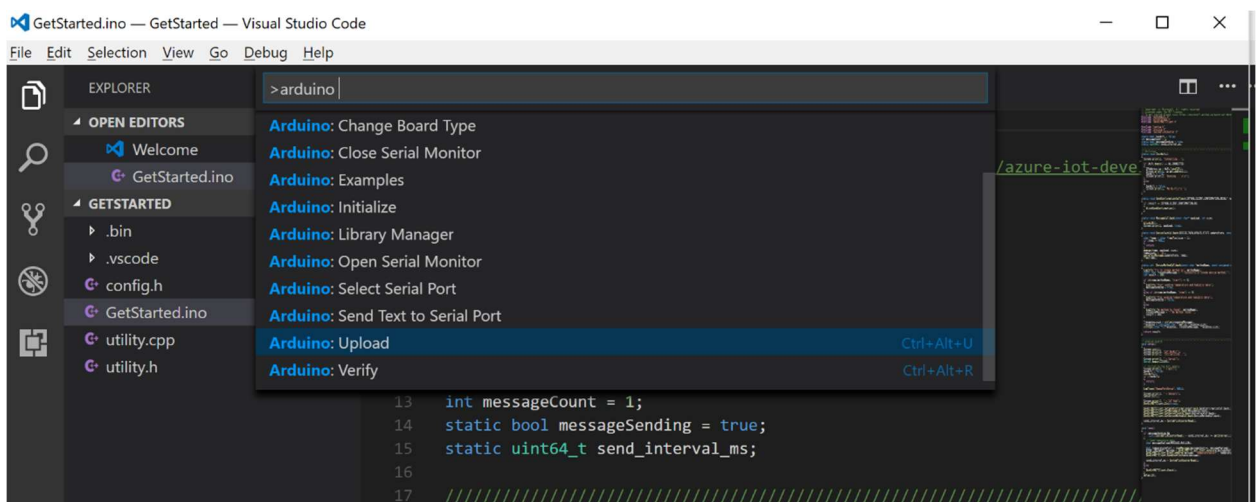
- checking pre-conditions of task: Set iot device connection string
? What connection string of iot device would you like to choose? Create new...
? Input the connection string of IoT Device: HostName=liki-iothub-4323.azure-devices.c
n;DeviceId=AZ3166;SharedAccessKey=4npPzu6FC1mLBZqCX3d2lWzJn0QuGEhcH0W6htuD8I=
Opening COM3.
Please hold on Button A and then press Button Reset to enter configuration mode.
SerialOutput Wi-Fi AP.
SerialOutput - set_wifissid: Set
  V Set iot device connection string: done

Terminal will be reused by tasks, press any key to close it.
```

8. 完毕后在 VS Code 命令行窗口中按任意键或点击垃圾桶图标关闭命令行窗口。

七. 编译并部署设备端代码

1. 在 VS Code 中按 F1 打开快捷命令菜单。然后输入“ arduino” , VS Code 会自动提示可以使用的命令：



2. 选择“ Arduino: Upload” , VS Code 会打开命令行窗口，并开始编译并部署代码至 DevKit：

```
16
17 ////////////////////////////////////////////////////
18 // Utilities

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL Arduino
xPSR: 0x61000000 pc: 0x2000002e msp: 0x200073fc
verified 507340 bytes in 1.674186s (295.934 KiB/s)
** Verified OK **
** Resetting Target **
Info : Unable to match requested speed 2000 kHz, using 1800 kHz
Info : Unable to match requested speed 2000 kHz, using 1800 kHz
adapter speed: 1800 kHz
shutdown command invoked
[Done] Uploaded the sketch: GetStarted.ino

Ln 17, Col 53 Spaces: 2 UTF-8 LF C++ MXCHIP AZ3166 COM3
```

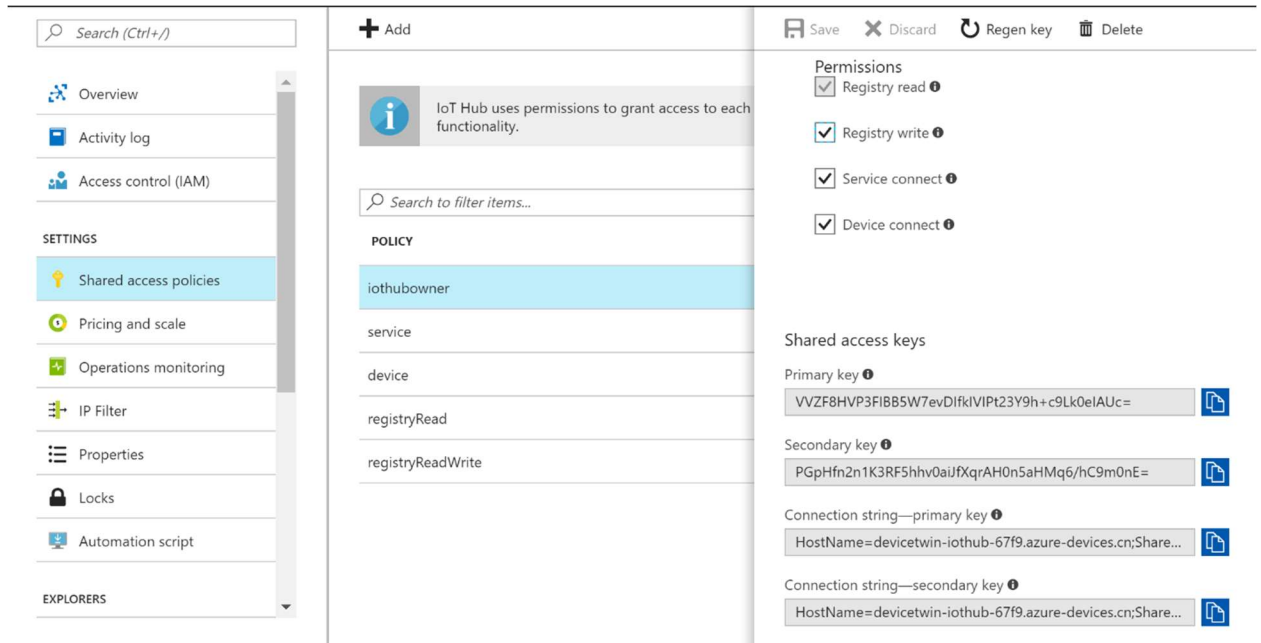
第一次代码编译并部署所需较长时间，大概需要 2-3 分钟。请耐心等待。

3. 最后编译并部署成功后，DevKit 会立即重启并开始运行刚才编译的程序。
4. 在 VS Code 命令行窗口中按任意键或点击垃圾桶图标关闭命令行窗口。

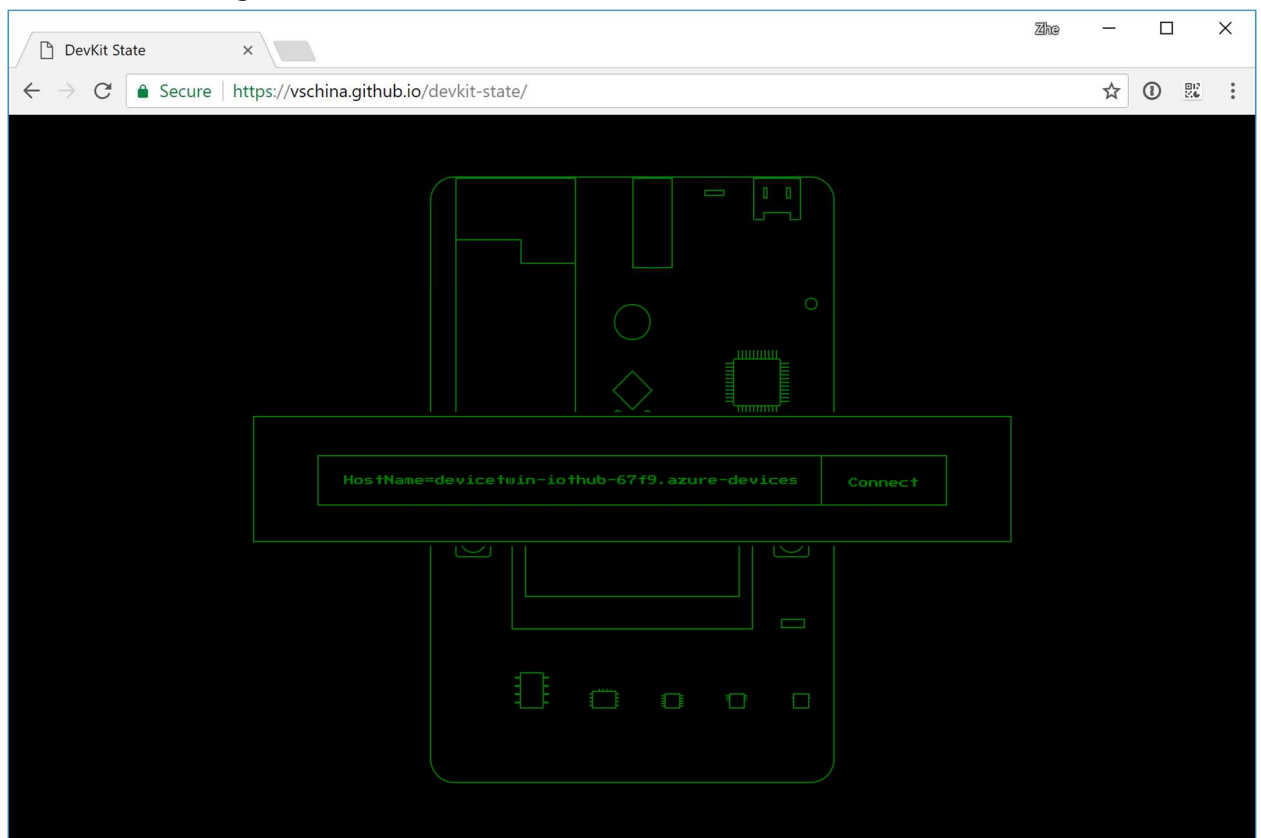
八．查看 IoT DevKit 开发套件状态

1. 打开 <https://portal.azure.cn> 访问 Azure 控制台

2. 打开 IoT Hub 页面，打开左侧 Shared access policies 菜单，点击 iothubowner，复制右侧 Connection string – primary key 文本框中的字符串



3. 在 Edge 浏览器中打开 <https://vschina.github.io/devkit-state/>，粘贴 IoT Hub connection string，点击 Connect



4. 几秒钟后可以看到 IoT DevKit 开发套件状态显示在页面上。点击页面上的 User LED 控制 IoT DevKit 开发套件上 User LED 的状态

