



CLOUD9 INSTALLATION GUIDE

(Using Raspberry Pi)

TABLE OF CONTENTS

INTRODUCTION 2

STEP 1: INSTALLING CLOUD9 2

STEP 2: SET CLOUD9 TO START ON BOOT..... 3

STEP 3: CONFIGURING CLOUD9..... 5

STEP 4: VERIFY BY TESTING 6

INTRODUCTION

Cloud9 is the main software we use to operate the SCUTTLE. It is a cloud-based Integrated Development Environment (IDE) that allows us to write, run, and debug applications from within a browser. The robust Cloud9 IDE also gives us the ability to easily view and manage our file hierarchy and run commands in the terminal.

This guide will give you comprehensive instructions for installing and configuring the Cloud9 IDE on your Raspberry Pi. It is expected that you already have a Raspberry Pi flashed with the latest version of Raspbian OS Lite that you can control using Secure Shell Protocol (SSH) and at least a basic understanding of Linux Command Line.

STEP 1: INSTALLING CLOUD9

In order to install Cloud9 you must have NodeJS installed on your Pi. NodeJS simply allows us to execute code outside of a browser. We need that capability because we get the Cloud9 software by cloning it from an online repository on GitHub. We will save that clone to a directory and then compile it to the Pi. Once Cloud9 has been compiled successfully we will have access to it on our browser.

1. Begin by opening a terminal in Command Prompt or Git Bash, either will work, and login as the user Pi using SSH. Then execute the following command in the terminal to install NodeJS and git:

```
sudo apt install -y git nodejs
```

2. Next, execute the following command to clone the latest version of Cloud9 to the directory '.c9sdk':

```
git clone https://github.com/c9/core.git .c9sdk
```

3. Then to enter the new cloned directory, execute the following command:

```
cd .c9sdk
```

4. Execute the following command to run an installation script to compile the cloned directory to the Pi:

```
./scripts/install-sdk.sh
```

Please note that this installation can take quite some time to complete. It can vary depending on how good of a connection the Pi has to the internet and the quality of the network it is on. Expect it to take at least 10-15 minutes and possibly longer to finish.

- When the compiling is finally completed you should see this in your terminal:

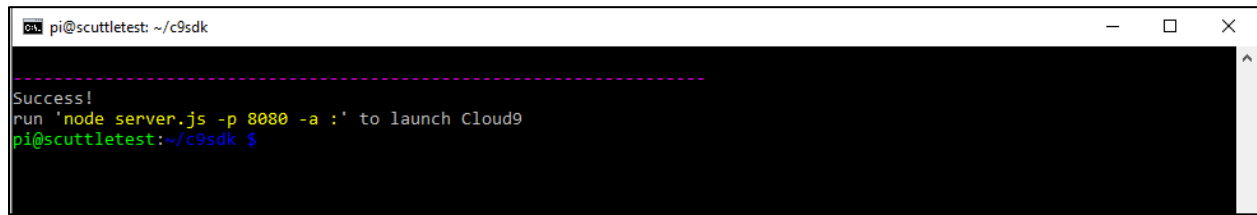
A terminal window titled 'pi@scuttletest: ~/c9sdk' with standard window controls. It shows a dashed line separator, the word 'Success!', a yellow prompt 'run 'node server.js -p 8080 -a :'' to launch Cloud9', and a green prompt 'pi@scuttletest:~/c9sdk \$'.

Figure 2

- To get Cloud9 running, execute the command in yellow that is says to run in order to launch Cloud9:

```
node server.js -p 8080 -a :
```

- Now open a browser and search for *hostname.local:8080* where *hostname* is either the default *raspberrypi* or whatever you may have changed it to such as *scuttle601*. Then the Cloud9 IDE should load.
 - If it loads successfully then congratulations, you have finished the installation of Cloud9 and are ready to proceed to step 2, skip part *b* and *c*.
 - If Cloud9 does not load in the browser, go back to the terminal and execute the following command:

```
node server.js -p 8080 -a : --listen 0.0.0.0
```

- Now open your browser again and refresh the page to see if the Cloud9 IDE loads. If it loads successfully then you are ready to proceed.

STEP 2: SET CLOUD9 TO START ON BOOT

Although we can access Cloud9 now, we are not done setting it up. If we left it as is, we would have to open a terminal and manually start Cloud9 every time the Raspberry Pi booted up. What we want to do is enable Cloud9 to automatically start every time the Pi is booted up so that we can go directly to the browser, open Cloud9, and begin working.

- Go back to the Command Prompt or Git Bash terminal and type the following command, it will create a Cloud9 service file:

```
sudo nano /lib/systemd/system/cloud9.service
```

- The Cloud9 service file will then open in the terminal with a menu at the bottom, the file should be empty since you just created it.

3. In the Cloud9 service file, type the following lines:

```
[Unit]

Description=Cloud9 IDE

After=multi-user.target network.target


[Service]

User=pi

WorkingDirectory=/home/pi/.c9sdk

ExecStart=/usr/bin/node /home/pi/.c9sdk/server.js -p 8080 -a : --listen 0.0.0.0 -w /home/pi

SyslogIdentifier=cloud9ide


[Install]

WantedBy=multi-user.target
```

4. Once you have that all typed, check it once again to make sure you typed it exactly as it is above. If it is all correct then use *ctrl-x* to exit the file menu, then enter *y* when prompted to save the changes and finally hit *enter*.
5. The modified Cloud9 service file will set up Cloud9 as a service and define the process it should follow upon booting the Pi. To run the service file and reflect those changes, execute the following command:

```
sudo systemctl start cloud9.service
```

6. The above command should start Cloud9, verify that you are able to access it in your browser.
7. Then use the following command to verify that the connection is good and see if there are problems:

```
sudo systemctl status cloud9.service
```

8. If the status comes back as all good, then execute the following command to enable the service file to cause Cloud9 to automatically run whenever the Pi is booted.

```
sudo systemctl enable cloud9.service
```

9. To test that everything works as it should, turn off the Pi and re boot it.
10. Then go to your browser and search for your *hostname.local:8080*. Cloud9 should then load successfully without any additional steps.

Congratulations! You have correctly enabled Cloud9 to automatically run each time you re-boot the Pi. That completes the setup part of this guide, but you still have a few steps to configure Cloud9 and verify that it works.

STEP 3: CONFIGURING CLOUD9

So now you have installed Cloud9 and set it up to automatically run when the Pi is booted up, but there are still some things that need to be configured within the Cloud9 IDE before you start using it to control the SCUTTLE. The main configurations we will do in this step involve changing the version of Python used to version 3 and installing a few python modules.

1. Go to your browser and search for your *hostname.local:8080* to open Cloud9.
2. First we need to modify which version of python is supported in Cloud9. To change this, go to the Cloud9 settings by clicking the gear in the top right of the IDE. This will open a preferences menu.
3. In the menu, navigate to Project Settings > Python Support > and change the Python version to Python 3.
4. Then close the preferences menu and navigate to the terminal at the bottom of the IDE.
5. The first python module we want to install is pip. To install pip, execute the following command:

```
sudo apt install python3-pip
```

6. The next python module to install is NumPy. To install NumPy, execute the following command:

```
sudo apt install python3-numpy
```

7. To work with the I2C board we also need to install the Adafruit GPIO. To do this, execute the command:

```
sudo pip3 install Adafruit_GPIO
```

Good job, you now have a properly installed, setup and configured version of the Cloud9 IDE on your Raspberry Pi. The next and final step will have a couple of quick tests you can perform to test Cloud9 and verify that it functions correctly.

STEP 4: VERIFY BY TESTING

In this step we will create a folder and a basic python script to verify that all the steps you have completed up to this point were done correctly. After this step is done you will be ready to start operating your SCUTTLE using Cloud9.

1. To do this, click on *File* in the top left of the IDE and select *New File*.
2. Use *'ctrl+s'* to save the file and save it as *'testing.py'* within the *.c9sdk/* folder.
3. Then in the window for the *testing.py* file, type:

```
print('Hello World!')
```

4. Again use *'ctrl+s'* to save the changes to the file.
5. Now go to the bash terminal at the bottom of the IDE and type the following two commands:

```
cd ~/.c9sdk
python3 testing.py
```

6. This will navigate to the *.c9sdk* directory where the *testing.py* file is saved and then run the file. It should print out *'Hello World!'* in the terminal as you can see in Figure 3 below:

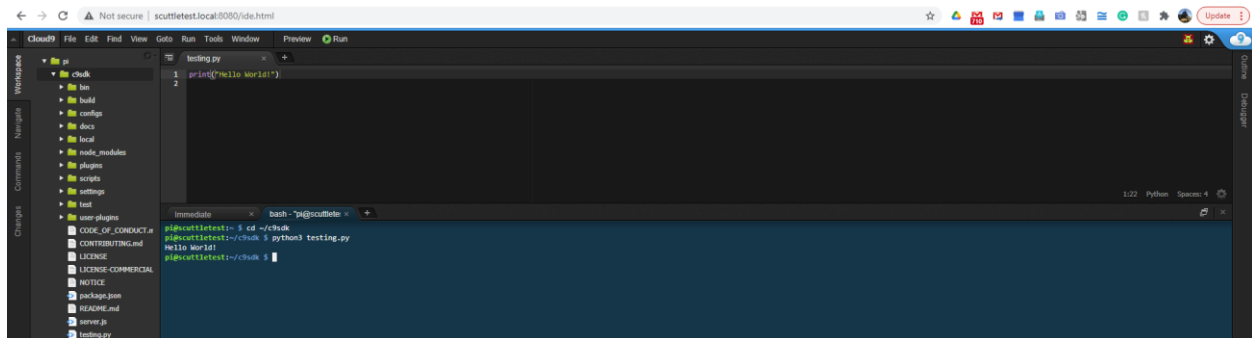


Figure 3

7. Then you can create a folder to store the *testing.py* file in.
8. Use the command *mkdir* followed by the name of the folder you are creating.

```
mkdir test
```

9. When you execute that command, you should notice a new folder appear on the left side of the IDE in the file directory with the name you assigned it.
10. Then simply drag and drop the *testing.py* file into the new folder you created. Open the folder and verify the file is in there. If that all runs properly then it ensures that Cloud9 works the way we set it up.