

Future Makers



Your name:

Your number:

Code samples and further instructions: www.github.com/Sage/future_makers

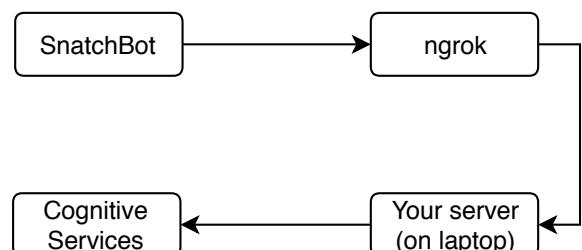
Workshop overview

Welcome to Future Makers Dublin. In this workshop you will learn how to use cutting-edge tools to build chatbots and connect them to artificial intelligence services. For example, you could build a bot that automatically translates what a user says into another language, a bot to describe images automatically, or a bot to recognise faces from an image.

During the course of the day we will look at

- running a web server on your laptop
- using this server to access Microsoft Cognitive Services
- building a chatbot using Snatchbot
- Using a tool called ngrok to connect Snatchbot to your laptop and then to Microsoft Cognitive Services.

We will first look at each service individually, then see how to connect them together. Once everything is wired up, our project will look like this:



Glossary of useful terms

When we **run** a program, we ask the computer to look at its source code and follow the instructions. **Debugging** a program is the same as running it, except that we get more information about errors if anything goes wrong.

Machine learning is a very broad term: it covers all programs which learn to do something, from learning to play chess from learning to recognise faces from photographs. A machine learning algorithm usually needs to be trained - during the training process, it learns to perform its assigned task.

Artificial intelligence is also a very broad term. It doesn't indicate that a program is as intelligent as a human (this would be artificial general intelligence) - merely that it is intelligent in some small way.

A **server** is a program which waits for users (or other programs) to communicate with it. For example, a web server waits to receive a **request** for a web page (via the user's browser) and then prepares and sends out the web page: the **response**.

Often, a server will listen only for requests from browsers. However, many servers talk to each other; for example, when you log into your webmail, there are plenty of servers communicating behind the scenes to retrieve your email from the database and display it as a web page.

A **port** is like a phone line: each server listens on a particular port, and requests have to come in on the right port. Ports are used to keep traffic separate for different applications on your machine: your email program might use one port and an online game might use another. The port for HTTP (Web) traffic is 80.

HTTP stands for **HyperText Transfer Protocol**. Hypertext refers to clickable links; HTTP is the protocol that powers the Web.

URL: Uniform Resource Locator or web address. This is what you see in the address bar when you visit a web page or file online.

An **API, or Application Programming Interface**, is simply a service which an organisation uses to allow other users (or servers) to connect and get some information (or perform a task). For example, the Met Office might publish a weather API so that people all over the world can connect and see the latest weather forecasts. Or an online gaming company might design an API so that many game clients can communicate with the game server.

Technologies we will be using

Python is a friendly, easy-to-learn but also very powerful programming language. We will be using the latest and greatest version: Python 3.

JSON: JavaScript Object Notation. This is a simple code used to send data between servers; it's often used in the JavaScript language. An example of JSON:

```
person: { name: 'Jeremy', age: ' 21' }
```

Microsoft Cognitive Services: apart from selling computers and software, Microsoft also sells artificial intelligence services. They have developed and trained many different machine learning algorithms which customers can use. We will be able to use face detection, object detection or translation.

SnatchBot is an online tool allowing chatbots to be easily built without needing to write code.

ngrok is a tool which allows anyone in the world (in our case, SnatchBot) to connect to your machine. It takes a server running on your machine and, using the ngrok servers, gives it an address which is visible to the public.

General programming tips

- Don't try and do everything at once. Make a small change, then test your program; once it's working, make another small change.
- If you're not seeing a change, make sure you have saved your code file (and restarted your server if the code is running a server).
- Don't worry if you don't understand anything or if something is confusing; just ask the demonstrators as soon as you're unsure and they will be happy to help.

- There's no such thing as a silly question - asking questions (and making mistakes) is how we learn.

Downloading the files we will need

First, make a folder on your desktop called **FutureMakers**. You will store all your work here; you will then be able to copy it onto a USB stick (which we will provide) at the end of the workshop.

Open https://github.com/Sage/future_makers. Click on **Clone or Download**, then **Download zip**. Make sure you extract the zip file - click on **Extract**. Move the resulting folder (which will be called `future_makers-master`) into your **FutureMakers** folder.

Getting to know a server program

From the Start menu, open Visual Studio Code; this is a Microsoft editor which allows us to edit our code files and also to run them.

First, add the **future_makers-master** folder to the side panel so we can see all our files. Click **File > Open Folder**, then navigate to the **future_makers-master** folder (which you should have put inside the **FutureMakers** folder on your desktop) and click **Select Folder**.

Now you can see all our files on the bar on the left hand side.

Make sure you understand what the four buttons on the left hand side do: **Explorer** shows our files, **Search** allows us to search them (we won't be using this), **Source Control** allows us to share them with other programmers (we won't be using this) and **Debug** allows us to run our programs. We will be switching frequently between **Explorer** and **Debug**.

First, open the **simple_server.py** file. This contains a simple server for demonstration purposes. Read all of the code and check you understand what it is doing. Ask one of the demonstrators if you have any questions.

Now let's run the file. Click on the **Debug** icon; you should see a green arrow at the top left. Click the drop down and click **Add Configuration**, then **Python**. You can now click the green arrow to run the program.

Go ahead and click it now. An output window should appear at the bottom of Visual Studio Code and you should see "The server is now listening." If you get a message from Windows Firewall, just click Allow or OK.

Now type **localhost:3003** into the address bar of your browser and hit return. You should see "Hello!". Make sure you understand what's happening here: the server in **simple_server.py** is listening for requests to come in, and responding to them by sending the text "Hello."

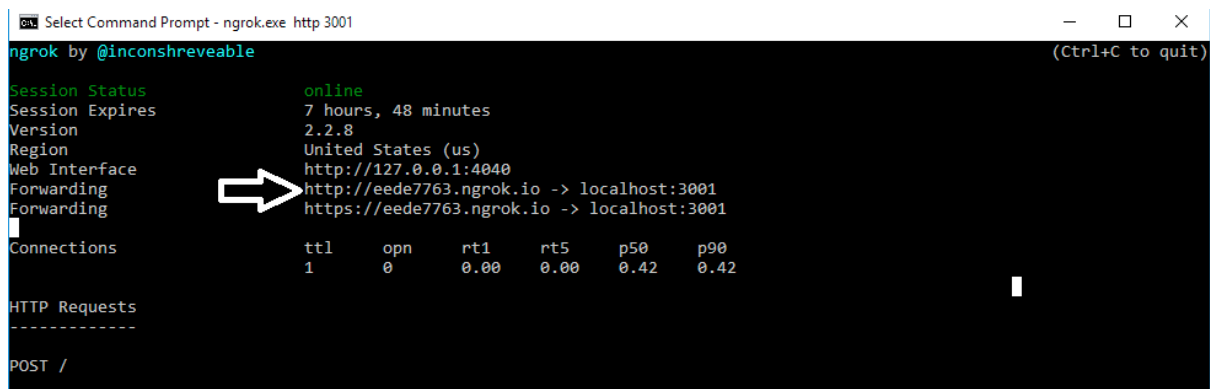
Stop the server by clicking on the red square (stop) in Visual Studio Code. Then change "Hello" to something else in the source code, start the server again, and refresh the page in your browser to check that you can see the new text.

Later in the workshop, we will use a different server to communicate both with Snatchbot and Microsoft Cognitive Services. It will run on your laptop and act as a go-between between the two services so that your chatbot can use exciting AI tools.

Ngrok

Ngrok provides a connection, called a “tunnel”, between Ngrok’s servers and your local machine so that it can receive requests from the Internet. You’ll need to run Ngrok in order to connect SnatchBot to the server running on your machine.

1. Download this [zip file](#) to your desktop
2. Double-click on the zip file
3. Drag the ngrok.exe to your desktop
4. Open Command Prompt (Start -> Run -> cmd)
5. Change to the Desktop folder (cd %USERPROFILE%\Desktop)
6. Run the command (**remove the ./ from the front**):
 - o ngrok.exe http 3001 (or whichever port you want to forward)
7. Leave the command open (you can minimise the window)
8. If you’ve followed this instructions correctly then your command prompt window should look like this!



```
Select Command Prompt - ngrok.exe http 3001
ngrok by @inconshreveable (Ctrl+C to quit)

Session Status      online
Session Expires     7 hours, 48 minutes
Version             2.2.8
Region              United States (us)
Web Interface        http://127.0.0.1:4040
Forwarding           http://eede7763.ngrok.io -> localhost:3001
                   https://eede7763.ngrok.io -> localhost:3001

Connections
  ttl    opn    rt1    rt5    p50    p90
   1      0     0.00   0.00   0.42   0.42

HTTP Requests
-----
POST /
```

The arrow in the above image points to the address assigned to your machine. Other servers can connect to this address and Ngrok will connect them directly to your machine. You will need this later for SnatchBot.

Now you should run `simple_server.py`, make sure it’s on the same port which ngrok is forwarding, and visit both `localhost:port` and the ngrok web address (arrow above). You should get the same results from both. Great - your server is visible to the world!

Testing Microsoft Cognitive Services

Let’s test our connection to Microsoft Cognitive Services by using their “vision” service; this tells us what objects are present in an image using machine learning.

First, open `python_examples/vision_example.py` in Visual Studio Code using the Explorer bar on the left.

Now run the file by clicking on the green arrow. The URL of an image is given in the code; this URL is sent to Cognitive Services, which looks at the image and uses machine learning to try and detect various objects in the image. You should see a window appear showing the image and its description.

Now look for a different image using Google Images, paste its URL into the code file, and see what Cognitive Services comes up with as a description.

Once you have this example working, try it through ngrok (again, make sure the ports are right) and check that you can access it from the public Web too.

Snatchbot

Please go to www.github.com/Sage/future_makers and follow the tutorial in **instructions/SnatchBot.md**. This will introduce you to some existing chatbots and show you how to set up your own.

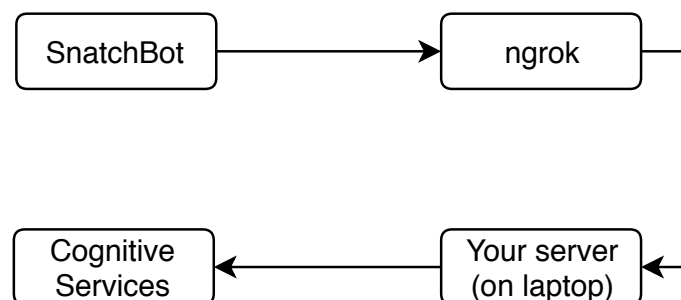
NB: Some of the screenshots are out of date, but the same buttons and functions are there.

Integrating everything

We've now tested all our components individually:

- building chatbots with SnatchBot
- using machine learning tools with Cognitive Services
- running a server locally and allowing the world to connect to it with ngrok.

Now it's time to integrate everything. Our final setup will look like this:



There are three links here and if anything goes wrong we will need to individually check they are working.

- Snatchbot to ngrok: check ngrok is running and you have given SnatchBot the right ngrok address.
- ngrok to your server: check you have launched ngrok on the same port your server is running on (see bottom of Python file). Then test your server by visiting both localhost:port, and the ngrok address.
- Your server to Cognitive Services: check you have the right API key. Check that the standalone Cognitive Services Python files (the ones that run on their own) work fine.

Continuing at home

Your SnatchBot account is yours and will continue to work.

Your Microsoft Cognitive Services key will work for a few weeks, and then stop working. Microsoft Cognitive Services is very cheap unless you make hundreds or thousands of requests to the server. If you can use someone's credit card, it's easy to sign up for your own account.

Visual Studio Code is free to download and you can install many Python libraries with one easy installer by installing the Anaconda scientific Python distribution (make sure you get one of the Python 3 versions, not Python 2). This will also allow you to use the Jupyter Notebook, a code editor which is very useful for learning Python.