AIY PROJECTS FOR DIGITAL MAKERS

Move beyond the Google Assistant SDK and put AIY Projects at the heart of your digital making

Last month, Google and *The MagPi* made history by bundling a complete AIY Projects kit on our cover.

This kit is the first do-it-yourself artificial intelligence project to be launched by Google; more are in development. With it, you can add voice commands and responses to your projects.

WITH GOOGLE

On the AIY Projects website, you will find the latest build of the AIY Projects software, as well as the most recent updated instructions. Take a look at the Maker's Guide in the AIY Projects website for up-to-date information.

aiyprojects.withgoogle.com/voice

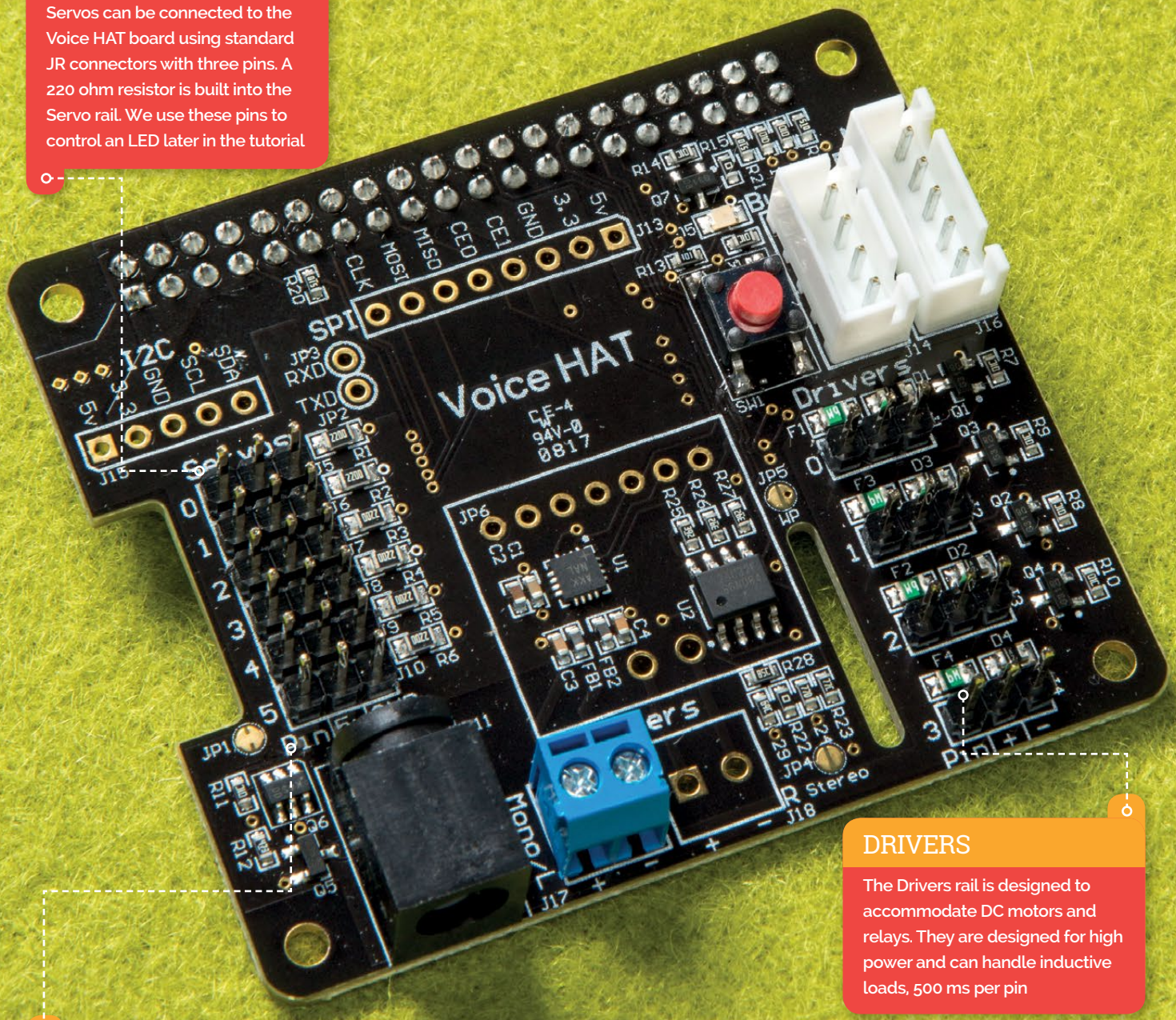
The starter project for the kit has you setup the Google Assistant SDK to create a digital assistant. Push the button on the top of the Voice Kit, ask your AIY Projects kit questions, and the Google Assistant will provide answers.

We hope you had a lot of fun building this starter project for using voice interactions, but it's important not to stop there. There's a lot you can do with this project beyond the Google Assistant SDK. For the curious maker, we have some ideas on how to hack AIY Projects, as well as how to use the Cloud Speech API as an alternative to the Google Assistant SDK. You can also set up Android Things as an OS.

In this feature, we're going to look at some of the things you can do with the AIY Projects voice kit.

SERVOS

Servos can be connected to the Voice HAT board using standard JR connectors with three pins. A 220 ohm resistor is built into the Servo rail. We use these pins to control an LED later in the tutorial



DRIVERS

The Drivers rail is designed to accommodate DC motors and relays. They are designed for high power and can handle inductive loads, 500 ms per pin

PIN GUIDE

Below the Servos and Drivers pins are guides to the function of each pin. The Servo pins are Pin, 5V, and GND; the Drivers pins are Pin, +, and -

WANT TO BUY AIY PROJECTS?

Not everybody was lucky enough to get a free AIY Projects voice kit with *The MagPi* #57, and many readers have asked whether it is possible to purchase the kit separately. Sign up for our newsletter and we'll let you know when more AIY Projects Voice Kits are available. Head to our website and enter your email address.

magpi.cc

SET UP CUSTOM VOICE COMMANDS

Google Assistant SDK

Set up Cloud Speech API to create your custom commands

AIY Projects voice is a hackable project, so we encourage you to make this project your own. This guide gives you some creative extensions, settings, and even a different voice API to use. We hope this project sparks some new ideas for you.

First, though, you need to move from using the Google Assistant SDK (the one that answers general questions) to the Cloud Speech API.

>STEP-01

Cloud Speech API

One interesting project for makers is to create custom commands for the AIY Projects Voice HAT. These can do just about anything with your Raspberry Pi. While it's possible to add new actions and voice commands using the Google Assistant SDK, more options are available if you switch to the Cloud Speech API. This software recognises your voice speech and converts it into text. The Cloud Speech API supports 80 languages, extended audio clips, and the ability to add phrase hints for processing audio.

>STEP-02

Turn on billing

To use Google's Cloud Speech API, you need to activate Billing – see page 28 of *The MagPi* #57 or **magpi.cc/2q5SSF7**. If you use it for less than 60 minutes a month, it's free. Beyond that, the cost is $0.006 for 15 seconds. Don't worry: you'll get a reminder if you go over your free limit.

Open the web browser and click on Google API Console link (or visit **console.developers.google.com**). In the API Manager window, click Enable API and search for Google Cloud Speech API. Click it and click Enable in the top of the window (if it says 'Disable', then it is already enabled and you're ready to go).

>STEP-3

Credentials

Choose Credentials in the sidebar. Click on Create Credentials and select Service Account Key. From the Service account menu, pick the name of your project (if you already have one) or select 'New service account'. Enter a name so that you'll know this is for your voice recognizer stuff, like 'Voice credentials'. Select 'Project viewer' as the role.

Select JSON as the key type and click Create.

>STEP-04

Rename credentials

The credentials file is downloaded automatically. The file name contains your project name and some numbers (like 'aiyproject-c92d36fc7055.json'). Open a Terminal window and move it to your home folder, and rename it to **cloud_speech.json**:

```
cd Downloads/

mv aiyproject-c92d36fc7055.
json /home/pi/cloud_speech.
json
```

>STEP-05

Check Cloud

On your desktop, double-click the Check Cloud icon. Follow along with the script. If everything is working correctly, you'll see this message: 'The cloud connection seems to be working.'

If you see an error message, try restarting your Raspberry Pi

with `sudo reboot`. Then follow the instructions above, or take a look at the instructions on the AIY Projects page (**magpi.cc/2q5SSF7**).

>STEP-06
Config file

The application is configured by adjusting the properties found in **/home/pi/.config/voice-recognizer.ini**. This file lets you configure the default activation trigger and choose which API to use for voice recognition. Don't worry if you mess it up: there's a backup copy kept in **/home/pi/voice-recognizer-raspi/config**.

>STEP-07
Adjust the config

Open a Terminal window and enter the following:

```
nano /home/pi/.config/voice-recognizer
```

Delete the `#` before `cloud-speech = true`.

Now press **CTRL+O**, press **ENTER**, and **CTRL+X** to save the file and exit Nano.

>STEP-08
Start it up

Double-click the 'Start dev terminal' icon and enter:

```
src/main.py
```

The Cloud Speech API will start. This API works like the Assistant SDK, but instead of answering general questions it responds to specific commands. Say out loud:

"What are the three laws of robotics?"

It will read the response from Issac Asimov's famous collection of robot books. This a pre-built example speech. The response is not part of the Assistant SDK, but a specific response (an action) to a voice command. You can create specific voice commands and actions.

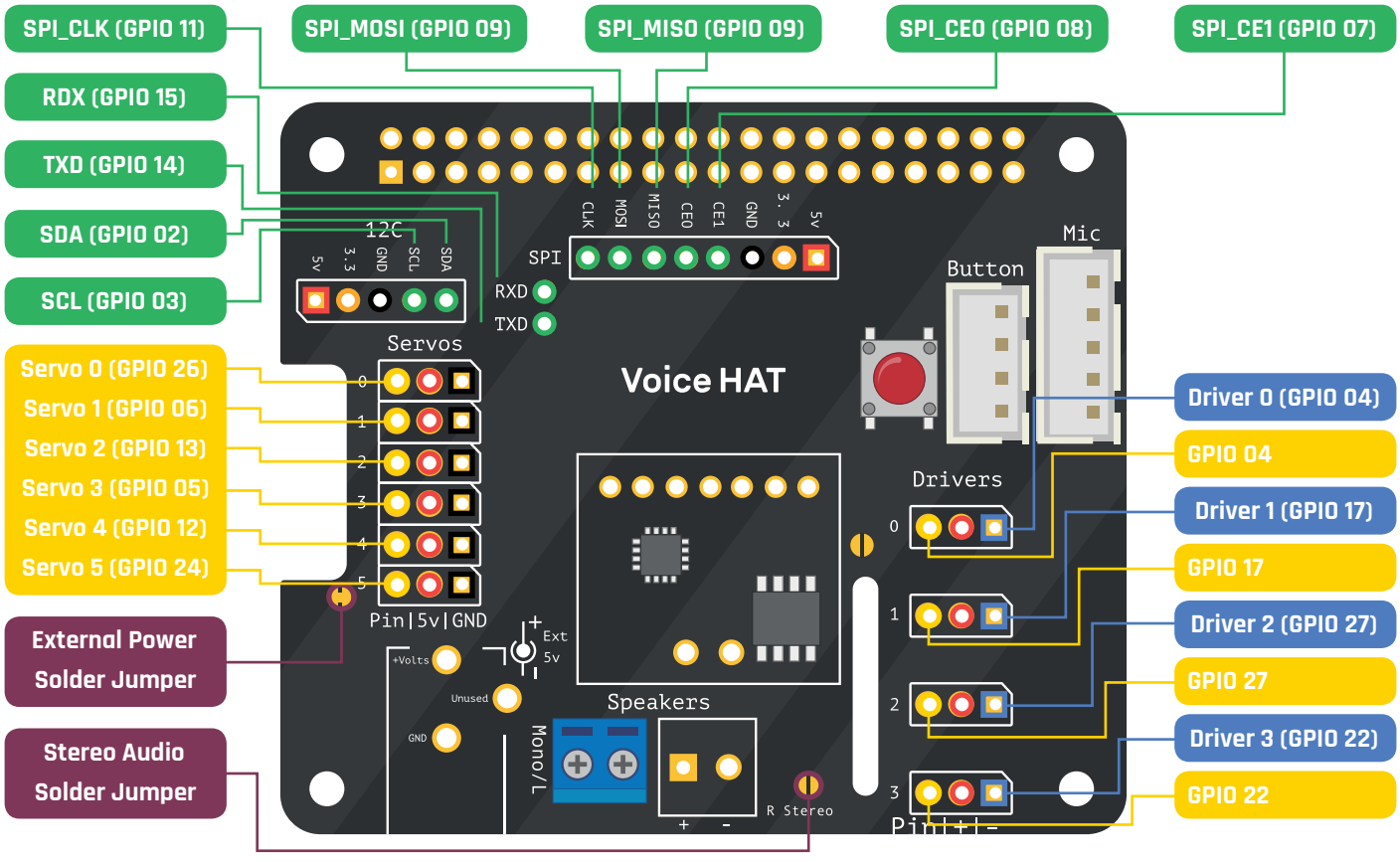
>STEP-09
Back up action.py

You create new actions and link them to new voice commands in **/home/pi/voice-recognizer-raspi/src/action.py**. We think it's a good idea to back up this file before editing it.

```
cd /home/pi/voice-recognizer-raspi/src/

cp action.py action_backup.py
```

VOICE HAT HARDWARE EXTENSIONS



CREATE ACTIONS

Google Cloud Platform

Edit the actions file to create custom voice commands for AIY Projects

You'll Need

- AIY Projects voice kit
- Cloud Speech API
- Breadboard
- LED, resistor, and cables

You can remove the kit's big arcade button and use the small red button on the board to activate the assistant

Now that you've switched from the Assistant SDK to the Cloud Speech API, you'll want to know what you can do with it. You add custom commands to the **action.py** file. There is a selection of example voice commands located right at the end of the code. These include the three laws of robotics that we used when testing the Cloud Speech API.

>STEP-01

Edit action.py

You can edit the **action.py** file in Nano, but we think it's better to use Python IDLE. Enter:

```
xdg-open /home/pi/voice-recognizer-raspi/src/action.py
```

This command opens the program in the Python IDLE window with the code marked up in colours, making it much easier to read.

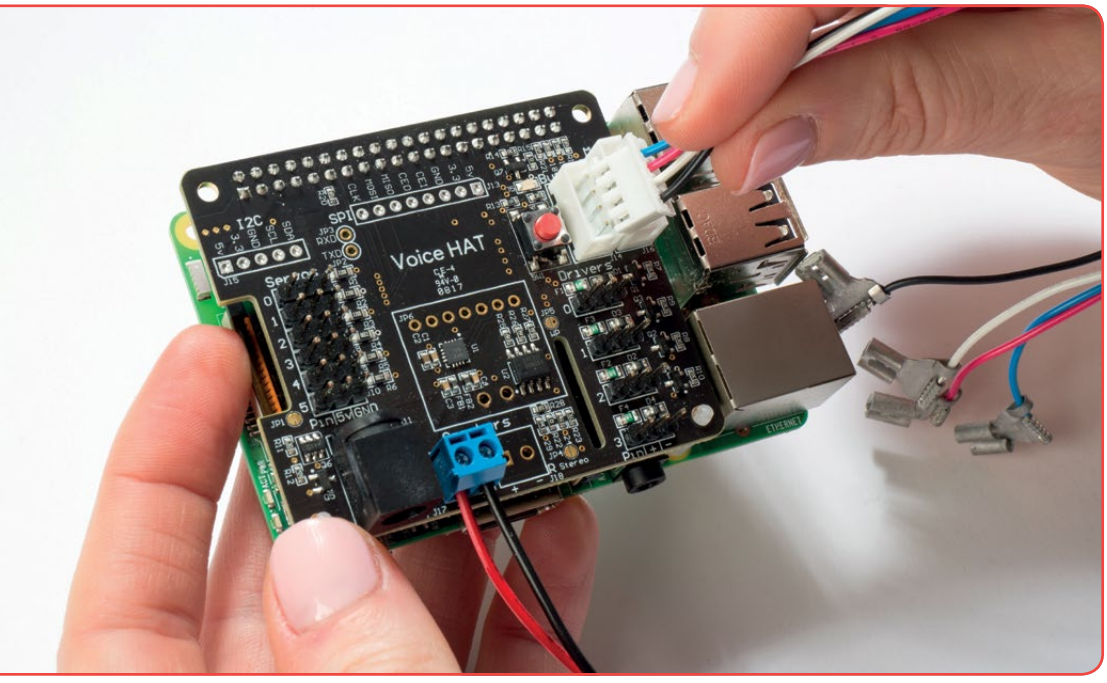
Scroll through and read the example functions. At the end, you'll see a section with:

```
#============================

#Makers! Add your own voice commands here.

#============================
```

Locate the function starting with **def_add_commands_just_for_cloud_speech(actor,say)**. This function contains the custom commands used in Cloud Speech API.



>STEP-02

Add a keyword

We're going to add a simple keyword that creates a custom voice response. Scroll to the very end of the document and enter (with an indent):

```
    simple_command(_('meaning of life'),_('The answer to the great question of life, the universe, and everything is 42.'))
```

Now start up the assistant. Double-click 'Start-dev terminal' and enter:

```
src/main.py
```

Say: **"What is the meaning of life?"**

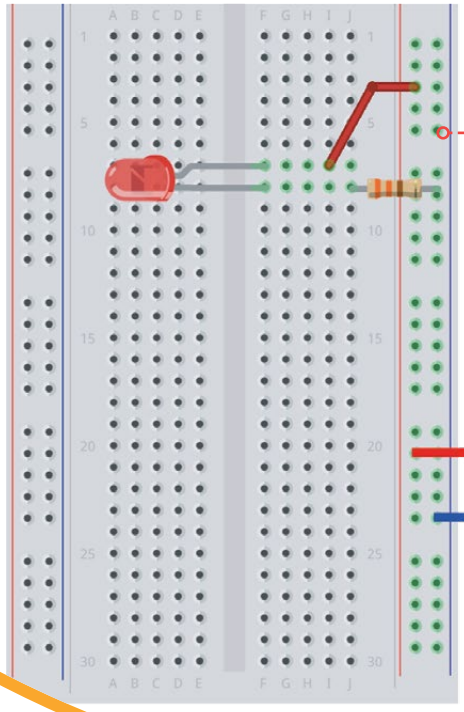
Google Cloud Speech API will detect your keywords and read out the classic quote from Douglas Adams's *The Hitchhiker's Guide to the Galaxy*.

>STEP-03

Control an LED

Now that we can create actions, we're going to use the AIY Projects kit to control some hardware. Set up an LED circuit using a breadboard – follow the diagram shown on the next page. We are connecting the LED via the pins on Servo 0. Connect the live wire to Pin (on the left). This is GPIO 26 using the BCM numbering system. Connect the ground wire to GND (on the right). The middle pin provides a constant 5V of power. You can see the reference for each pin underneath the Servo 5 rail (check the diagram in 'Voice HAT hardware extensions' on p69 for reference).

SET UP AN LED CIRCUIT

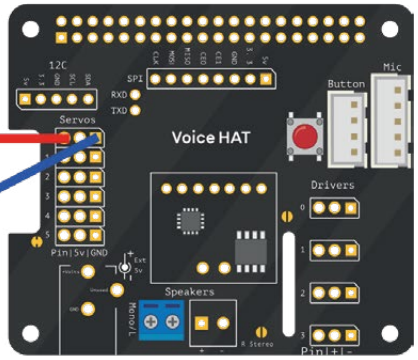


CIRCUIT

Connect an LED to the breadboard and create a circuit (with the longer leg connected to live and the shorter leg connected to ground). Don't forget to use a resistor (around 330 ohms) to protect the LED

GPIO 26

Connect the live wire to GPIO 26, the leftmost pin on Servo 0, and the live rail on the breadboard. See the GPIO layout guide from the previous page for guidance



GND

Connect the ground wire to the GND pin on the Servo 0 rail and the ground rail on the breadboard

We have found that it will work by connecting wires directly to the through-holes on the board. For a more reliable circuit, however, solder the pins supplied with your Voice HAT.

>STEP-04

Add an action

We need to open and edit the **action.py** file again. Enter this command in a Terminal window:

```
xdg-open /home/pi/voice-recognizer-raspi/src/action.py
```

Scroll down and look for this comment:

```
# ============================

# Makers! Implement your own actions here.

# ============================
```

Add the code from **own_actions.py** (page 73) underneath the comment.

>STEP-05

Voice command

Then add the code from **voice_commands.py** (page 73) to **~/voice-recognizer-raspi/src/action.py** below the comment 'Add your own voice commands here'. Make sure it indents with the comment, like this:

```
    # ======================

    # Makers! Add your own voice commands here.

    # ======================

    actor.add_keyword('light on', GpioWrite(26, True))
```

In **voice_commands.py** we've created actors for both "light on" and "lights on". We find this makes recognition more reliable. Add the rest of the code, then select File > Save and exit IDLE.

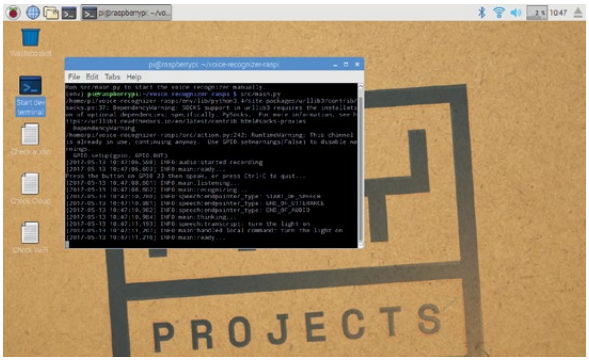
>STEP-06

Light on

Double-click 'Start dev terminal' and start up the assistant:

```
src/main.py
```

Now say "light on" slowly and clearly. If everything is connected correctly, your LED will light up. Say "light off", and it will turn off again. You can also use variants such as "hey, light on" and "turn the lights on".



Use code to control GPIO pins via your assistant. This short program turns an LED light on or off

MAKE A VOICE-CONTROLLED CHESS GAME

Create a working chess program with AIY Projects

You'll Need

- AIY Projects voice kit
- GNU Chess

One of the key things with AIY Projects is that it allows you to add human voice interaction to your projects. So you can move beyond using buttons (real or virtual) and mouse clicks. With the Cloud Speech API, you can talk directly to your Raspberry Pi, and have a natural conversation with your devices.

The hope is that in the near future, you'll be able to walk up to a device and say "what are you?" and "how do I use you?" and have a conversation with it.

We are delighted to find *The MagPi* readers already creating code for AIY Projects that interacts with devices, such as this program by reader Mike Redrobe (**magpi.cc/2q8NW20**). It hacks together AIY Projects with GNU Chess and is an excellent example of building an interface that controls a program.

Quick Tip

SOURCE CODE

If you're using the SD card image provided by AIY Projects, the source for the voice-recognizer app is already installed on your device. You can browse the Python source code at **/home/pi/voice-recognizer-raspi/**. Alternatively, the project source is available on GitHub: **magpi.cc/2pH04KQ**. Browsing code such as **main.py** and **action.py** is the best way to get a feel for what the code is capable of doing.

>STEP-01

Commands

You can issue Terminal commands via Cloud Speech API. Let's test this out by adding a handy function to the AIY Projects voice kit: the ability to shut down and restart the kit with a voice command.

In **shutdown.py** (page 73) there are two `actor.add_keyword()` functions. These contain 'shut down' and 'restart' as the command words. Enter the code from **shutdown.py** into **action.py**, as you did with **voice_commands.py**.

Start up the assistant (enter `src/main.py` in 'Start dev terminal'). Now when you use Cloud Speech API, you can press the button and say "shut down". The AIY Projects kit will run the shutdown scripts and power off safely. Say "restart" to restart your AIY Projects kit.

>STEP-02

Install GNU Chess

We are going to take this idea of command-line interaction to greater lengths by setting up the Cloud Speech API to work with GNU Chess for a command line-based chess game. The first step is to install GNU Chess on your Raspberry Pi.

```
sudo apt-get update

sudo apt-get install gnuchess
```

GNU Chess is played from the command line, although often it will be used with a graphical interface such as XBoard. But we're going to use AIY Projects to enter the moves in the command line.

>STEP-03

AIY chess

Now add the code from **aiy_chess.py** (page 73) to the **action.py** file (in **home/pi/voice-recognizer-raspi/src/action.py**). This code adds interaction with GNU Chess to your Cloud Speech API. Enter the code and start the assistant. Double-click 'Start dev terminal' and enter:

```
src/main.py
```

Now press the button on your AIY Projects voice kit to make a move.

>STEP-04

Playing chess

You say "chess D2 D4" to play a move. You will see a small chessboard displayed in text on the AIY Projects screen. The AIY Projects kit will then speak the computer opponent's move (and show it on the text chessboard). You can also be lazy and say "chess D4" if there is only one piece that could move to D4.

aiy_chess.py

```
# ========================================
# Makers! Implement your own actions here.
# ========================================

p = None
class playChess(object):

    def __init__(self, say, keyword):
        self.say = say
        self.keyword = keyword

    def run(self, voice_command):

        move = voice_command.replace(self.keyword, '', 1)

        global p
        if (p == None):
           p = subprocess.Popen(["/usr/games/gnuchess","
               -q"],stdin=subprocess.PIPE,stdout=subprocess.PIPE)

        p.stdin.write(bytes(move.lower() + '\n', 'utf-8'))
        p.stdin.flush()

        response = ""
        if all(x.isalpha() or x.isspace() for x in move):
           # no numbers (d2d4) so its a command like new,undo,remove
           response = p.stdout.readline().decode("utf-8")
           response = "ok," + move
        else:
            self.say("ok, you played," + move)

            while ("move" not in response):
              response = p.stdout.readline().decode("utf-8")
              logging.info("Chess log: %s", response)

        logging.info("Chess: %s", response)
        self.say(response)

    # ========================================
    # Makers! Add your own voice commands here.
    # ========================================

    actor.add_keyword(_('chess'), playChess(say,_('chess')))
```

voice_commands.py

```
# ========================================
# Makers! Add your own voice commands here.
# ========================================

actor.add_keyword('light on', GpioWrite(26, True))
actor.add_keyword('lights on', GpioWrite(26, True))

actor.add_keyword('light off', GpioWrite(26, False))
actor.add_keyword('lights off', GpioWrite(26, False))
```

own_action.py

```
# ========================================
# Makers! Implement your own actions here.
# ========================================

import RPi.GPIO as GPIO

class GpioWrite(object):

    '''Write the given value
                to the given GPIO.'''

    def __init__(self, gpio, value):
        GPIO.setmode(GPIO.BCM)
        GPIO.setup(gpio, GPIO.OUT)
        self.gpio = gpio
        self.value = value

    def run(self, command):
        GPIO.output(self.gpio, self.value)
```

Quick Tip

SOLDER YOUR PINS

The Voice HAT works more reliably with the pins soldered in the holes. A strip of pins will have been supplied with your AIY Projects kit.

If you want a rough guide to soldering, here is a very handy instructional comic (PDF): **magpi.cc/2pUgMlr**. It's an excellent visual reference and a cool thing to put on the wall. Be careful using the board with soldered pins, as it's easy to get a 5V shock by touching the positive and negative pins. Power down the kit before connecting and disconnecting wires.

shutdown.py

```
# ========================================
# Makers! Add your own voice commands here.
# ========================================

actor.add_keyword(_('shut down'), SpeakShellCommandOutput(say, "sudo shutdown -h now",_('Shutdown failed')))
actor.add_keyword(_('reboot'), SpeakShellCommandOutput(say,"sudo shutdown -r now",_('Reboot failed')))
```