

Integrare Google Assistant con Python via IFTTT

Lo scopo di questa attività è integrare il sensore e l'attuatore nell'ambiente Google Assistant, utilizzando i servizi di IFTTT. Nel seguito si vedrà come realizzare un semplice Web Server con Flask e un applet su IFTTT che attivi il Web Server tramite Google Assistant. Il tutto poi potrà essere arricchito con le funzionalità di controllo del sensore e dell'attuatore.

*Nella **Sezione A** si configura l'applet IFTTT e nella **Sezione B** è riportato un esempio di programma associato attivabile dall'applet.*

Quanto descritto fa riferimento alle funzionalità disponibili da browser, ma è possibile effettuare le stesse operazioni dall'app IFTTT su Android.

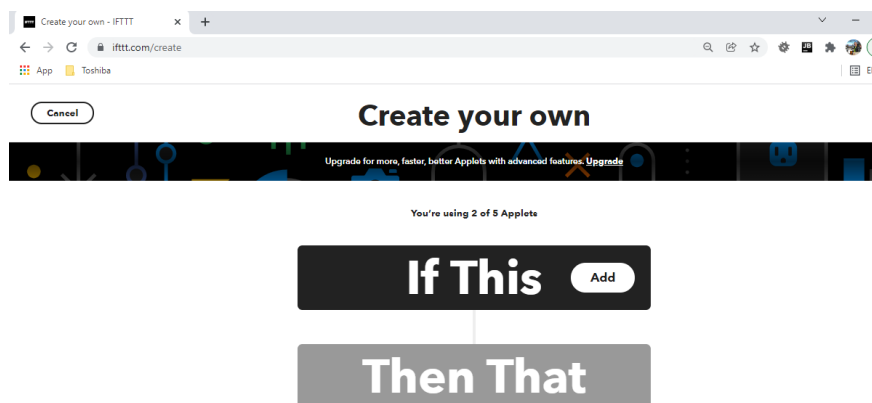
Sezione A: configurazione IFTTT

Parte 1: accesso alla console di sviluppo

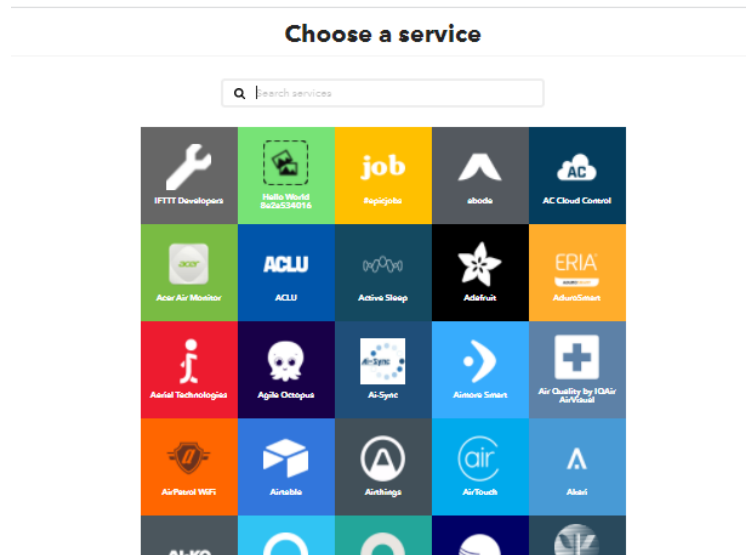
Fase 1.1: creare un account IFTTT su <https://ifttt.com/> utilizzando le stesse credenziali Google del dispositivo dal quale si utilizzerà Google Assistant (per funzionalità avanzate accedere poi eventualmente a <https://ifttt.com/developers>)

Parte 2: configurazione dell'applet

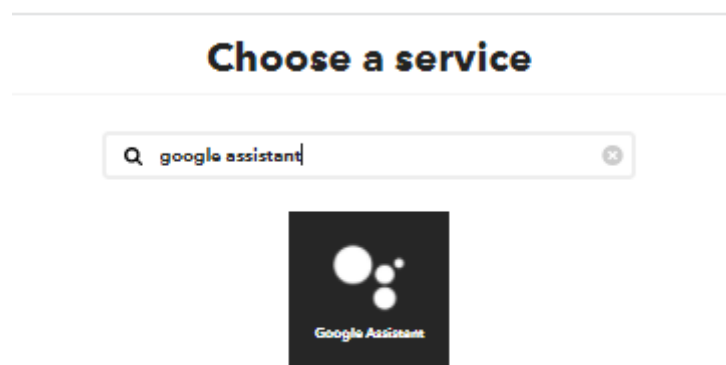
Fase 2.1: dal pannello iniziale scegliere **My applets** e poi **Create**, ottenendo:



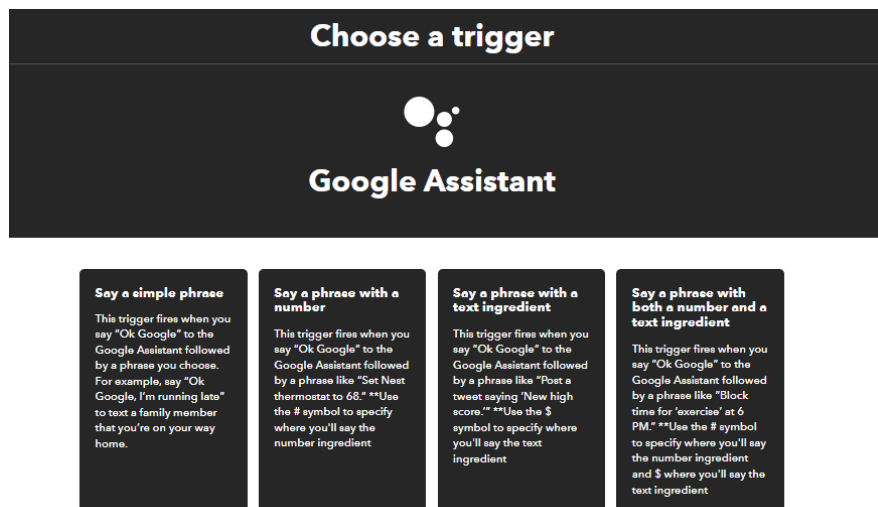
Fase 2.2: cliccare su **Add** a fianco di **If This**: si configura in questo modo l'evento che scatena (trigger) un altro evento; nel nostro caso l'evento scatenante è il comando vocale che daremo a Google Assistant, ad esempio il comando **Stampa**. Si ottiene quanto segue:



Fase 2.3: scegliere *Google Assistant*



Fase 2.4: scegliere *Say a phrase with a text ingredient*



Fase 2.5: configurare il colloquio con Google Assistant; diremo:

1. Noi: *Hey Google*
2. Noi: *Stampa la ricetta per fare il risotto*
3. Google: *OK, stampato la ricetta per fare il risotto*

Stampa è il comando mentre ciò che segue è il parametro, rappresentato di seguito dal carattere \$.

Completare e cliccare *Create Trigger*

Complete trigger fields

Say a phrase with a text ingredient

This trigger fires when you say "Ok Google" to the Google Assistant followed by a phrase like "Post a tweet saying 'New high score.'" **Use the \$ symbol to specify where you'll say the text ingredient

What do you want to say?

stampa \$

Enter a \$ where you'll say the text ingredient

What's another way to say it? (optional)

Enter a \$ where you'll say the text ingredient

And another way? (optional)

Enter a \$ where you'll say the text ingredient

What do you want the Assistant to say in response?

Ok, stampato \$

You can enter a \$ where you want to hear the text ingredient in the response

Language

Italian

Create trigger

Fase 2.6: creare l'evento scatenato dal trigger, cliccando su **Add** a fianco di **Then That**:

If

 Say a phrase with a text ingredient

EditDelete

+

Then That

Add

Choose a service

Q Search services




Hello World
8e2e534016



(more:trees)



abode




AC Cloud Control



Adafruit



ERIA



AI-Sync







Fase 2.7: scegliere *Webhooks* e poi *Make a web request*: completare con l'url del web server (ngrok nel nostro caso – dovrà ovviamente essere stato lanciato precedentemente) e la stringa JSON da passare al server, all'interno della quale sarà presente ciò che abbiamo detto a Google Assistant (nell'esempio: *la ricetta per fare il risotto*), rappresentato dal `{{TextField}}`

The screenshot shows a configuration screen for a 'Make a web request' action. At the top, it says 'Complete action fields' and 'Make a web request'. Below this, there are several fields: 'URL' with the value 'https://368c-151-38-167-87.ngrok.io', 'Method' set to 'POST', and 'Content Type' set to 'application/json'. There is also an 'Additional Headers' section and a 'Body' section containing the JSON string `{"perm": "{{TextField}}"}.`. Each field has an 'Add ingredient' button next to it. At the bottom, there is a large blue button labeled 'Create action'.

Fase 2.8: scegliere *Create Action*, poi *Continue* e *Finish*.

My Applets

The screenshot shows the 'My Applets' interface. At the top, there is a search bar with a magnifying glass icon and the word 'Filter'. Below this, there are tabs for 'All (2 of 5)', 'Published', and 'Archive'. To the right of these tabs is a link that says 'Get Pro to get 20 Applets'. The main area displays a card for an applet titled 'If You say "esegui \$", then Make a web request' by 'emiliopennati'. The card has a 'Connected' status indicator and a small icon at the bottom left.

Sezione B: Web Server

Parte 3: script Python-Flask

Fase 3.1: considerare il programma IFTTT.py di esempio (riportato in versione copiabile in coda a questo documento)

```
1  from flask import Flask,request
2  import json
3
4  app = Flask(__name__)
5
6  @app.route("/",methods = ['POST'])
7  def stampa():
8      d=request.get_json()
9      print("stampato: "+d["parm"])
10     return("ok")
```

Fase 3.2: lanciare il programma di esempio

```
$env:FLASK_APP = "ifttt.py"
py -m flask run
```

Fase 3.3: lanciare *ngrok*

```
.\ngrok.exe http 5000
```

```
Session Status      online
Account             (Plan: Free)
Version             2.3.40
Region              United States (us)
Web Interface        http://127.0.0.1:4040
Forwarding           http://368c-151-38-167-87.ngrok.io -> http://localhost:5000
Forwarding           https://368c-151-38-167-87.ngrok.io -> http://localhost:5000

Connections          ttl      opn      rt1      rt5      p50      p90
30                 0        0.00     0.00     0.33     0.34

HTTP Requests
-----
```

Parte 4: test

Fase 4.1: chiedendo a Google Assistant:

1. Noi: *Hey Google*
2. Noi: *Stampa la ricetta per fare il risotto*
3. Google: *OK, stampato la ricetta per fare il risotto*

si ottiene:

```
* Serving Flask app 'ifttt.py' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
stampato: la ricetta per fare il risotto
127.0.0.1 - - [12/Dec/2021 21:07:04] "POST / HTTP/1.1" 200 -
□
```

```
from flask import Flask, request
import json

app = Flask(__name__)

@app.route("/", methods = ['POST'])
def stampa():
    d=request.get_json()
    print("stampato: "+d["parm"])
    return("ok")
```