



Artificial intelligence. Robotics creació de prototips IoT amb **Raspberry Pi** laboratori final

**formador**joan.masdemont@gmail.com

Joan Masdemont Fontàs

**participant de l'aula**joan.carles.sv@gmail.com

Joan Carles Simón Vidal



Creació de prototips de IoT amb Raspberry

Especialitat Formativa

Família professional: Informàtica i Comunicacions

Àrea Professional: Sistemes i telemàtica

Especialitat: IFCT80

Nivell: 3

Dates: 26/10/2022 a 22/12/2022

Duració: 210 hores

OBJECTIUS DEL CURS

Desenvolupar prototips que integrin sensors, electrònica, tractament de dades i altres tecnologies d'Internet de les coses (IoT), controlats amb un computador tipus Raspberry i programats amb un llenguatge de programació.

CONTINGUTS FORMATIUS:

Programa formatiu	Hores
Mòdul 1: L'ecosistema Raspberry	60
Mòdul 2: Desenvolupament d'aplicacions d'Internet de les coses (IoT)	140
FCOO03 Inserció laboral, sensibilització mediambiental i en la igualtat de gènere (10 hores)	10
És obligatòria per a tot l'alumnat, excepte que acrediti que l'ha cursada en els darrers 12 mesos.	
Total hores	210

CIFO BARCELONA LA VIOLETA: Plaça Comte de Sert, 25, 08035 Barcelona

Tel. 93 254 17 00 - cifo_violeta.soc@gencat.cat

<https://serveiocupacio.gencat.cat/cifolavioleta>

Història de Revisions

data	versió	descripció	autor
20/10/23	ver. 1.0.0	Document inicial	Joan Carles Simón Vidal

Projecte Robot ARM

Braç robotic controlat per telèfon intel·ligent

Objectius específics.

Amb l'únic requisit d'emprar el mateix ecosistema utilitzat al llarg del curs, aquest laboratori pretén acreditar el coneixements adquirits.

Com que disposem de total llibertat a l'hora d'escollar el repte, jo he complert una vella il·lusió de la infància. Sempre vaig tenir la curiositat de "treballar" amb un braç robòtic. El meu laboratori és la creació d'un braç robotic basat en un excel·lent projecte publicat en Internet (a la web www.thangs.com) adaptat a una Raspberry Pi i en llenguatge Python a diferència de l'original que utilitzava Arduino UNO R3 i llenguatge C.

El laboratori està dividit en sis parts. Les dues primeres compostes per la creació dels models i impressió 3D, atès que no té cabuda en aquest curs, he utilitzat tal qual els del projecte original publicats a la web anteriorment comentada.

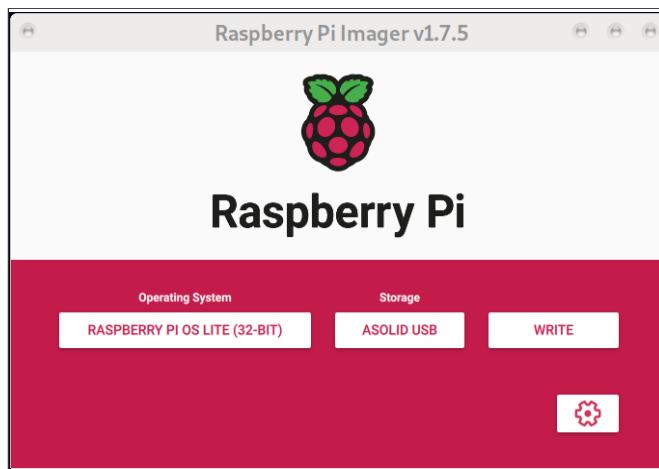
La tercera part, l'assemblatge del model 3D, atès que és el mateix disseny no té cap variació amb el projecte original.

És en aquest punt, els passos 4,5 i 6, es on he adaptat als objectius del curs substituint per un disseny basat en una **Raspberry Pi** comunicada amb el Smartphones mitjançant Wi-Fi i llenguatge Python a diferència de l'original realitzat amb un sistema **Arduino UNO R3** comunicat amb el telèfon intel·ligent a través de Bluetooth i desenvolupat amb llenguatge C.

Creació de l'ecosistema

Configurar i Assegurar el funcionament de la Raspberry Pi

Per crea l'ecosistema he utilitzat el *Raspberry Pi Image* que és la manera ràpida i senzilla d'instal·lar el sistema operatiu a la nostra Raspberry Pi. Aquest procés que en el curs ho hem anomenat com la "cematció de la sd" instal·la i personalitza el sistema operatiu dins d'una targeta microSD a modus de disc dur per la RPi, deixant-la ja llesta per al seu ús.



Aquests són els valors que hem utilitzat per a la configuració del Sistema Operatiu del nostre miniPC

Advanced options

Image customization options for this session only ▾

- Set hostname : raspberrypiC.local
- Enable SSH
 - Use password authentication
 - Allow public-key authentication only
- Configure wireless LAN

SSID : **cifo-Aula-i32-IoT-2023**

Hidden SSID

Password : **123abcd123**

Wireless LAN country : ES ▾
- Set locale settings

Time zone: Europe/Madrid ▾

Keyboard layout: es ▾

Persistent setting

- Play sound when finished
- Eject media when finished
- Enable telemetry

SAVE

Internet de les coses IoT

Desenvolupament d'aplicacions



Braç robotic controlat per telèfon intel·ligent i Raspberry Pi

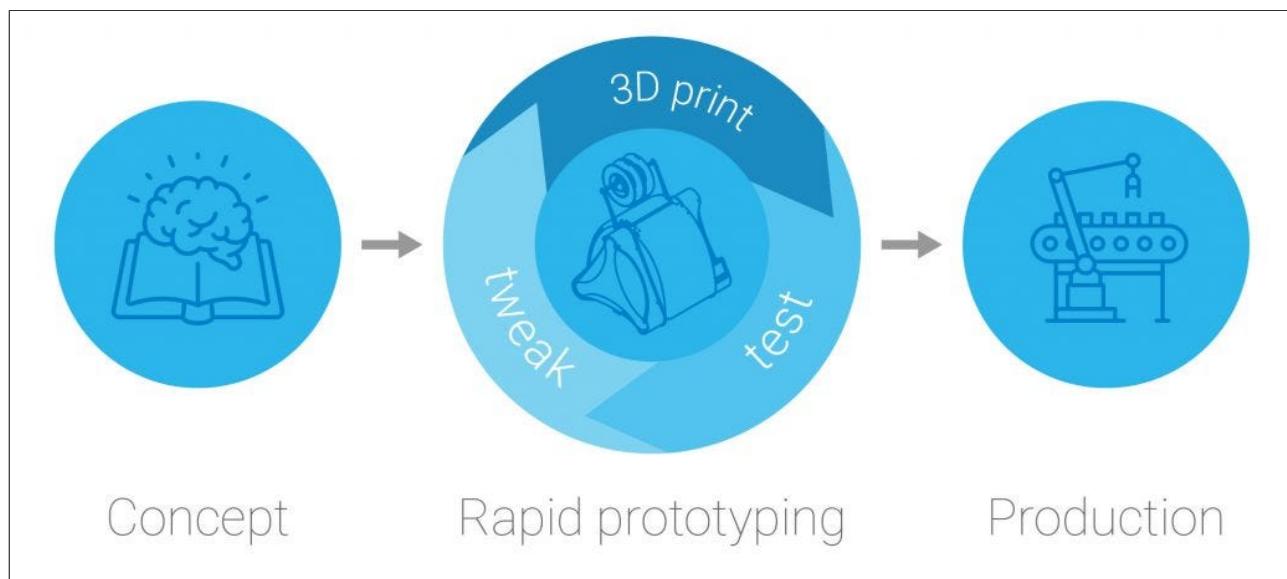
En aquest document mostra les diferents parts per a fer un braç robotic Arduino que es pot controlar i programar sense fil mitjançant una aplicació d'Android personalitzada. Mostraré tot el procés de construcció, el disseny i la impressió en 3D de les peces del robot, la connexió dels components electrònics i la programació de l'Arduino, fins al desenvolupament de la nostra pròpia aplicació d'Android per controlar el Braç del robot.

Si bé el disseny i la impressió en 3D de les peces del robot son molt important, son aliens a la temàtica del curs i per tant no faré més que una simple menció a aquests processos doncs reconexent que són imprecisiables per a la realització del laboratori, com dic, no són part del contingut del nostre curs.

Visió general

Mitjançant els controls lliscants de l'aplicació podem controlar manualment el moviment de cada servo o eix del braç del robot. També utilitzant el botó "Desa" on podem gravar cada posició o pas i després el braç robot pot executar i repetir aquests passos automàticament. Amb el mateix botó podem aturar el funcionament automàtic així com restablir o eliminar tots els passos per poder gravar-ne de nous.

Lifecycle Creació de prototips de IoT amb Raspberry



Braç robotic. Model 3D

Per començar, vaig dissenyar el braç robot amb el programari de modelatge 3D Solidworks. El braç té 5 graus de llibertat.

Per als 3 primers eixos, la cintura, l'espatlla i el colze, he utilitzat els servos MG996R, i per als altres 2 eixos, el rodatge del canell i el pas del canell, així com la pinça, vaig utilitzar els micro servos SG90 més petits.



Podem trobar i descarregar aquest model 3D, així com explorar-lo al nostre navegador a Thangs:
Baixeu el model 3D de muntatge de Thangs. Fitxers STL per a la impressió 3D:

<https://howtomechatronics.com/tutorials/arduino/diy-arduino-robot-arm-with-smartphone-control/#unlock>

Braç robotic. Impressió del model en 3D

Com que el curs no contempla ni el disseny i la impressió 3D he utilitzat els arxius originals descarregats d'internet talcual. Donada la bona sintonia amb els integrants del curs de impressió 3D que es realitza paral·lelament al nostre, com dic gràcies a això he pogut obtenir la impressió 3D de totes les peces necessàries, moltes gràcies nois.

Especialment el meu agraïment màxim a Sr. Robert, docent d'aquest curs, tant per la seva predisposició així com la dedicació necessària i com no per a la impressió de totes i cadascuna de les peces, ja que sense això no m'hauria estat possible realitzar aquest atractiu laboratori.



Braç robotic. Muntatge

D'acord, en aquest punt estem preparats per muntar el braç del robot.

Vaig començar amb la base a la qual vaig connectar el primer servomotor mitjançant els cargols inclosos en el seu paquet. Llavors, a l'eix de sortida del servo, vaig assegurar una banya rodona i un cargol.



I a sobre hi vaig col·locar la part superior i la vaig fixar amb dos cargols.



Aquí de nou, primer va el servo, després la banya rodona a la part següent i, a continuació, es fixen entre si amb el cargol de l'eix de sortida.



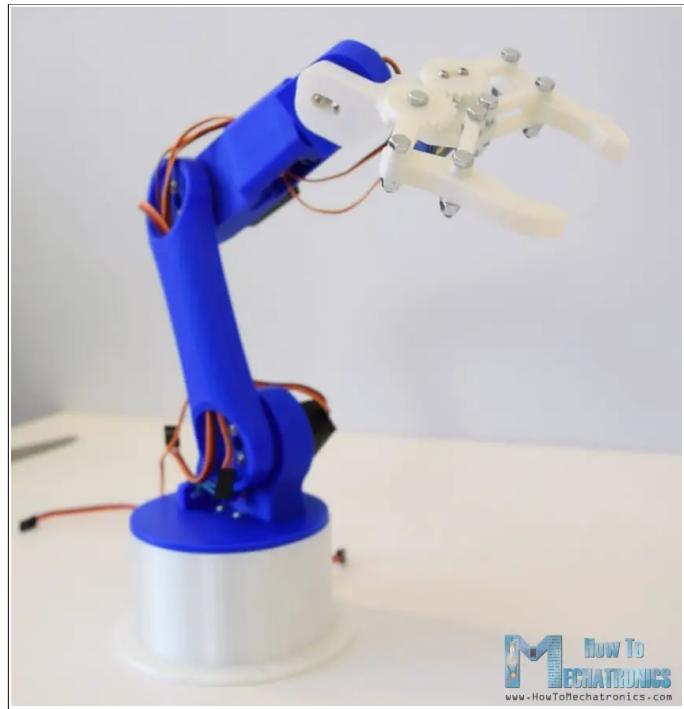
Podem notar aquí que a l'eix de l'espantilla és una bona idea incloure algun tipus de molla o en el meu cas vaig utilitzar una goma elàstica per donar una mica d'ajuda al servo perquè aquest servo també suporta tot el pes de la resta del braç. com a càrrega útil..



De la mateixa manera, vaig continuar muntant la resta del braç del robot. Pel que fa al mecanisme de pinça, vaig utilitzar uns cargols i femelles de 4 mil·límetres per muntar-lo.

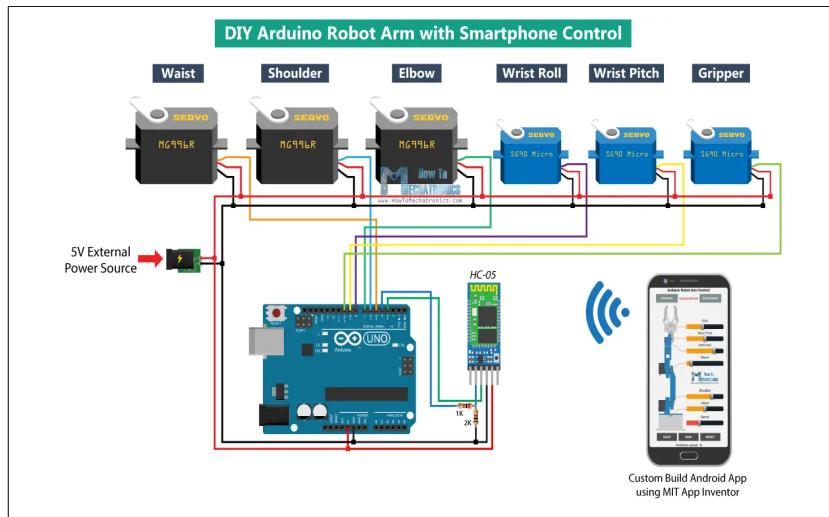


Finalment, vaig connectar el mecanisme de pinça a l'últim servo i es va completar el braç del robot Arduino.



Braç robotic. Diagrama del circuit del braç Arduino/Raspberry

La següent etapa és connectar l'electrònica. El diagrama de circuit d'aquest projecte és realment bastant senzill. Només necessitem una placa Arduino i un mòdul Bluetooth HC-05 per a la comunicació amb el telèfon intel·ligent. Els pins de control dels sis servomotors estan connectats a sis pins digitals de la placa Arduino.



A la nostra versió del braç robòtic el controlarem amb una Raspberry Pi versió 3 en comptes de l'Arduino simplificant el circuit donat que ja porta incorporat tant Bluetooth com connexió Wifi. Pel que fa a l'aplicació també hem adaptat l'app al temari del curs i l'he feta amb Python.

Aquest és lesquema proposat inicialment. Si bé la RPi no té tots els ports del GPIO amb PWM nadiu (només els 12,13,18 i 19), farem servir la resta dels ports a manera de "PWM brut", si bé no genera ones ones no tan netes, hem pogut provar que són perfectament valides per a aquest projecte.

Braç robotic. Codi

A l' hora de realitzar la programació per a aquest projecte us he subdividit en tres àrees. Una primera fase dedicada a "domar els components" necessaris per al control de braços robotic. Això s'aconsegueix mitjançant el us de uns petits servos (SC90) per al control de la mà i altres més grans (MG996R) per al que definiríem com el colze, braç, espalda. En aquesta fase he invertit el temps a trobar la millor manera d'utilitzar aquests servos i els seus valors més optims així com el seus límits.

La segona part l'he dedicat a la creació d'una llibreria composta per un conjunt de classes que em permeti desenvolupar de manera optimitzada el desenvolupament d'app. Aquestes classes contemplen els diferents elements d'IoT aplicats en aquest projecte, com és la gestió dels ports del Raspberry Pi (GPIO), el control del PWM d'aquests ports i els servos. A més, per tal de suavitzar el moviment del braç robotic també he creat un conjunt de classe que em permet gestionar amb multifils els diferents servos.

Per últim és el desenvolupament de la pròpia app que controla el braç robòtic. Tots el desenvolupament ho he realitzat amb llenguatge Python atès que és l'empleat al curs.

Com que el codi és una mica més llarg, per a una millor comprensió, publicaré el codi font del programa en seccions amb descripció per a cada secció. I al final d'aquest article publicaré el codi font complet.

Braç robotic. Codi Fase 1 “domar els components”

El primer que he necessitat per a aquest projecte, ja que treballa amb components de IoT que no hem vist al curs, ha estat "domar" aquests elements. El servos té una peculiaritat, si bé utilitzen el PWM per controlar el moviment ho fan en dues fases. D'una banda per controlar la direcció en què es mouen utilitza en delay del **dutycycle** mentres que el propi **dutycycle** controla l'angle del servo.

```
delay_middle = 1.5
delay_right = 2
delay_left = 1
```

direcció de la rotació del servo

D'altra banda he descobert que millor valor per la freqüència de PWM per als servos SC90 de 180º és de 50.

```
pi_pwm = GPIO.PWM(pin,50)
pi_pwm.start(0)
```

definició de la freqüència aplicada al PWM

En el cas dels servos SC90 de 180 º, per moure'l a la posició de 0 º, o sigui cap a la la dreta, el valor del **dutycycle** és de 2 i per optemar 180º, o sigui cap a la esquerra aquest valor és de 13. Tot i així en les proves realitzades sobre el servo que controla les pinces del braç robotic, aquest rang es veu reduït a, entre 3 i 10, en cas contrari els engrages podrien patir i fins i tot partir-se.

```
def loop(pi_pwm):
    print('-> move servo middle <-')
    moure_servo(pi_pwm, 7, delay_middle)

    print('<- move servo left')
    moure_servo(pi_pwm, 2, delay_left)      # left -90 deg position

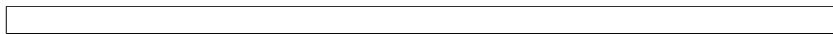
    print('-> move servo middle <-')
    moure_servo(pi_pwm, 7, delay_middle)      # middle position

    print('  move servo right ->')
    moure_servo(pi_pwm, 12, delay_right)      # right +90 deg position
```

Braç robotic. Codi Fase 2 Ilibreria del components

La llibreria està composta per quatre mòduls Python :

- **iot_rpi_componens_lib.py** aquest mòdul conté les classes principals dels components bàsics de IoT: pin, pin PWM i LED
- **iot_rpi_sensors_lib.py** compost per les classes de controlm pels diferents sensors que hem vist al curs com son : sensor de moviment, temperatura, humitat...etc
- **iot_rpi_servos_lib.py** en aquest mòdul tenim les classes per al control dels servos i motors
- **iot_rpi_jobs.py** en aquest modul estan les classes que ens permet crear i gestionar els fils d'execucio sota el paradigma de progrsmacion paral·lela.



Braç robotic. Codi Fase 3 app Robot ARM

A la darrera fase del desenvolupament s'ha creat l'aplicació pròpiament del prototip Robot AMR per això hem aplicat les classes creades en el pas anterior permetent reduir el temps i la complexitat d'aquesta fase de procés.

Per al desenvolupament de l'app he fet servir el paradigma MVC (Model/View/Controller) separant el codi en funció de la seva utilitat. D'una banda la lògica, de l'altra la persistència de dades i última que gestionen la IHM (interfície /home/màquina).

Braç robotic. Conclusions

El primer és reconèixer que el projecte proposat per la meva part, s'ha demostrat ser excessivament ambiciós per al límitat temps disponible. Així, que el que inicialment era un *Robot ARM* a resultat de ser, a causa de diversos incidents, un *Robot HAND*. Tot hi que el curs acaba aquí, tinc com a objectiu personal la voluntat de completar el projecte.

Atès que el curs no cobreix el procés de disseny i impressió 3D he utilitzat una versió disponible a Internet que també he descobert, a posteriori, que hauria d'haver ajustat per a una correcta implementació del meu projecte.

Pel que fa referència pròpiament als dispositius d'IoT utilitzats, tot i haver dedicat una part important del temps del projecte al coneixement i domat dels servos, vaig descobrir gairebé al final (degot al retard en el lliurament del servos per part del proveedor,) que els models de 180° i 360° no tenen en comú la gestió dels angles, ni comparteix els mateixos valors per fer-ho.

D'altra banda, a resultat sorprenent els pics que produeixen els servos SG996R i això m'obliga a redissenyar el circuit electrònic per protegir la placa de la Raspberry Pi d'una més que probable sobre càrrega.

Per últim, la intenció inicial era utilitzar una xarxa creada sobre una Raspberry Pi a mode d'AP i una altra Raspberry Pi a mode controlador del Robot ARM però... tot i haver-ho provat amb èxit a l'ambi domèstic no hi va haver manera que funcionés correctament al centre. Això m'ha obligat a redissenyar el projecte de manera que he simplificat la infraestructura a una única RPi.

Després d'una llarga investigació he descobert que, si bé el disseny inicial era correcte, aquesta infrestructura proposades en el projecte es veia afectat per xocar amb una xarxa oculta per els alumnes que té CTTI en el CIFO.

Annexos del curs

Annexa A. Raspberry Pi Wireless Access Point

Raspberry Pi Wireless Access Point

<https://pimylifeup.com/raspberry-pi-wireless-access-point/>

Un punt d'accés sense fil Raspberry Pi és una manera fantàstica d'allargar la durada de la nostra cobertura Wi-Fi i proporcionar accés addicional a la nostra xarxa.

Haureu de tenir en compte que un dongle Wi-Fi probablement no podrà gestionar tant trànsit com un encaminador normal. Això vol dir que hauríeu d'evitar permetre que hi hagi massa connexions al dispositiu per evitar que s'ocupi massa i quedí lent.

Tot i que podem utilitzar qualsevol Wifi que admeti l'habilitació com a punt d'accés, el nostre tutorial se centrarà directament en com configurar-ho per al mòdul Wi-Fi del Raspberry Pi 3.

Aquest tutorial es pot combinar bé amb el tutorial de punt d'accés VPN. El tutorial del punt d'accés VPN mostrarà com configurar un client OpenVPN i redirigir tot el trànsit a través d'aquest client.

A continuació es mostren tots els fragments i peces que he utilitzat per a aquest tutorial del punt d'accés sense fil Raspberry Pi, no hi ha res especial que necessiteu per poder completar-ho. Recomanat

Recomanat

- Raspberry Pi
- Micro SD Card
- Power Supply
- Wi-Fi
- Ethernet Cable
- Raspberry Pi Case

Opcional

- Raspberry Pi Case

Com amb la majoria de tutorials, aquest només utilitza una versió neta de Raspbian que s'ha actualitzat als darrers paquets. Per configurar el punt d'accés sense fil Raspberry Pi farem ús de dos paquets. Aquests dos paquets són *hostapd* i *dnsmasq*.

- **hostapd** és el paquet que ens permet utilitzar un dispositiu Wi-Fi com a punt d'accés, en el nostre cas, ho farem servir per convertir el Wi-Fi del Raspberry Pi 3 en el nostre punt d'accés.
- **dnsmasq** actua com a servidor DHCP i DNS perquè puguem assignar adreces IP i processar sol·licituds de DNS a través del nostre mateix Raspberry Pi.

Afortunadament, *dnsmasq* és fàcil de instalar i configurar. També té l'avantatge que és una mica lleuger en comparació amb els paquets *isc-dhcp-server* i *bind9*. Recordeu que per això haureu d'utilitzar una connexió de xarxa Ethernet i no la nostra connexió Wi-Fi.

pas 1. Abans de començar a instal·lar i configurar els nostres paquets, primer executarem una actualització al Raspberry Pi executant les dues ordres següents.

```
sudo apt update
sudo apt upgrade
```

pas 2. Fet això ja podem instal·lar els nostres dos paquets, executar la següent comanda per instal·lar *hostapd*, *dnsmasq* i *iptables*.

```
sudo apt install hostapd dnsmasq iptables
```

pas 3. Ara que tenim els paquets instal·lats, encara no volem que s'executin ja que no els hem configurat correctament. Aturem l'execució dels paquets utilitzant les dues ordres següents al terminal. Aquestes ordres indicaran al gestor del sistema que aturi els serveis dnsmasq i hostapd.

```
sudo systemctl stop hostapd
sudo systemctl stop dnsmasq
```

pas 4. Amb *hostapd* i *dnsmasq* ara aturats, voldrem modificar la nostra configuració *dhpcd* perquè puguem prendre el control de la interfície *wlan0*. Amb aquest fitxer, ens establirem una adreça IP estàtica i li direm que no faci ús del fitxer *wpa_supplicant* perquè puguem configurar-lo únicament com a punt d'accés al nostre dispositiu. Executeu l'ordre següent al vostre Raspberry Pi per començar a modificar el fitxer **dhpcd.conf**.

```
sudo nano /etc/dhcpcd.conf
```

pas 5. Dins d'aquest fitxer hem d'afegir la següent línia a la part inferior, això configurarà la nostra interfície *wlan0* de la manera que volem per al nostre tutorial.

NOTA: Si heu actualitzat a Raspbian Stretch, és possible que s'hagi de canviar *wlan0*, si utilitzeu el Raspberry Pi 3 o el wifi integrat del Pi Zero W, podeu continuar utilitzant *wlan0*. Utilizeu l'ordre **ip a** per veure quins són els noms nous, és probable que siguin força llargs. Haureu d'actualitzar qualsevol referència als nous valors al llarg d'aquest tutorial.

```
interface wlan0
    static ip_address=172.16.0.1/16
    nohook wpa_supplicant
```

Ara podem desar i sortir del fitxer prement **Ctrl + X**, després prement **Y** i després Enter.

pas 6. Ara hem de reiniciar el nostre servei *dhpcd* perquè es carregarà en tots els nostres canvis de configuració. Per fer-ho, executeu l'ordre següent per tornar a carregar el servei dhpcd.

```
sudo systemctl restart dhcpcd
```

pas 7. A continuació, hem d'ajustar la nostra configuració *hostapd*, per fer-ho hem de començar a editar el fitxer de configuració amb la següent comanda.

```
sudo nano /etc/hostapd/hostapd.conf
```

pas 8. En aquest fitxer hem d'escriure les línies següents, que bàsicament configuren com volem interactuar amb el dispositiu wlan. Les úniques línies reals de les quals us hauríeu de preocupar en aquest fitxer són la línia **ssid=** i la línia **wpa_passphrase=**. Com a regla general, hauríem de provar de fer que la nostra frase de contrasenya WPA tingui més de 6 caràcters per mantenir la nostra connexió segura.

NOTA: Si esteu fent aquest tutorial amb un dispositiu Wi-Fi diferent del Pi 3 incorporat, és possible que també hagiu de canviar la línia del controlador= al millor controlador per al vostre dispositiu, Google serà el vostre amic per descobrir què és el el millor controlador per utilitzar és.

```
interface=wlan0
driver=n180211
hw_mode=g
channel=6
ieee80211n=1
wmm_enabled=0
macaddr_acl=0
ignore_broadcast_ssid=0
auth_algs=1
wpa=2
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
# This is the name of the network
ssid=Pi3-AP
# The network passphrase
wpa_passphrase=<pi my life up>
```

Recordem canviar **wpa_passphrase** per la nostra pròpia contrasenya, assegureu-nos de configurar-la prou segura perquè persones aleatòries no es puguin connectar al nostre punt d'accés Wi-Fi. Ara podem desar i sortir del fitxer prement **Ctrl + X**, després prement **Y** i després Enter.

pas 9. Fet això, ara hauríem de tenir la nostra configuració hostapd, però abans que es pugui utilitzar hem d'editar dos fitxers. Aquests fitxers són els que llegirà hostapd per trobar el nostre nou fitxer de configuració. Per començar a editar el primer d'aquests dos fitxers, executeu l'ordre següent.

```
sudo nano /etc/default/hostapd
```

pas 10. En aquest fitxer, hem de trobar la línia i la rep

Troba-la

```
#DAEMON_CONF=""
```

substitui per:

```
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

Ara podem desar i sortir del fitxer prement **Ctrl + X**, després prement **Y** i després Enter.

pas 11. Ara hem d'editar el segon fitxer de configuració, aquest fitxer es troba dins de la carpeta init.d. Podem editar el fitxer amb l'ordre següent:

```
sudo nano /etc/init.d/hostapd
```

pas 12. En aquest fitxer, hem de trobar la línia següent i substituir-la.

Troba-la

```
DAEMON_CONF=
```

substitui per:

```
DAEMON_CONF=/etc/hostapd/hostapd.conf
```

Ara podem desar i sortir del fitxer prement **Ctrl + X**, després prement **Y** i després Enter.

pas 13. Amb *hostapd* ara configurat, hem de passar a configurar *dnsmasq*. Abans de començar a editar el seu fitxer de configuració canviarem el nom de l'actual ja que no necessitem cap de les seves configuracions actuals.

Ho podem fer amb la següent comanda mv al nostre Raspberry Pi.

```
sudo mv /etc/dnsmasq.conf /etc/dnsmasq.conf.orig
```

pas 14. Ara que s'ha canviat el nom del fitxer de configuració original, podem començar creant el nostre propi fitxer de configuració. Crearem i editarem el nou fitxer amb l'ordre nano.

```
sudo nano /etc/dnsmasq.conf
```

pas 15. A aquest fitxer afegeix les línies següents. Aquestes línies indiquen al servei *dnsmasq* com gestionar totes les connexions que arriben i per a quina interfície les hauria de gestionar.

```
interface=wlan0          # Use interface wlan0
server=1.1.1.1            # Use Cloudflare DNS
dhcp-range=172.16.0.30,172.16.0.50,12h  # IP range and lease time
```

Ara podem desar i sortir del fitxer prement **Ctrl + X**, després prement **Y** i després Enter.

pas 16. A continuació, hem de configurar el nostre **Raspberry Pi** perquè reenvii tot el trànsit de la nostra connexió *wlan0* a la nostra connexió ethernet.

En primer lloc, hem d'habilitar-lo mitjançant el fitxer de configuració **sysctl.conf**, així que comencem a editar-lo amb l'ordre següent.

```
sudo nano /etc/sysctl.conf
```

pas 17. Dins d'aquest fitxer, heu de trobar la línia següent i eliminar el **#** del principi.

Troba-la

```
#net.ipv4.ip_forward=1
```

substitui per:

```
net.ipv4.ip_forward=1
```

Ara podem desar i sortir del fitxer prement **Ctrl + X**, després prement **Y** i després Enter.

pas 18. Ara, com que estem impacients i no volem esperar que s'activi el proper arrencada, podem executar l'ordre següent per activar-lo immediatament.

```
sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip_forward"
```

pas 19. Amb el reenviament IPv4 ara habilitat, podem configurar un NAT entre la nostra interfície *wlan0* i la nostra interfície *eth0*. Bàsicament, això reenviarà tot el trànsit des del nostre punt d'accés a la nostra connexió Ethernet.

Executeu les ordres següents per afegir les nostres noves regles a l'iptable.

```
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

pas 20. L'iptable es buida a cada arrencada del **Raspberry Pi**, de manera que haurem de desar les nostres noves regles en algun lloc perquè es tornin a carregar a cada arrencada.

Per desar el nostre nou conjunt de regles, executeu l'ordre següent.

```
sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"
```

pas 21. Ara, amb les nostres noves regles desades amb seguretat en algun lloc, hem de tornar a carregar aquest fitxer a cada reinici. La manera més senzilla de gestionar-ho és modificar el fitxer **rc.local**.

Executeu l'ordre següent per començar a editar el fitxer.

```
sudo nano /etc/rc.local
```

pas 22. Ara estem en aquest fitxer, hem d'afegir la línia de sota. Assegureu-nos que aquesta línia aparegui a sobre de la **exit 0**. Aquesta línia bàsicament lleixa la configuració del nostre fitxer **iptables.ipv4.nat** i les carrega a l'iptables.

Troba-la

```
exit 0
```

Afegeix a sobre de "exit 0"

```
sudo hostapd /etc/hostapd/hostapd.conf &
iptables-restore < /etc/iptables.ipv4.nat
```

Ara podem desar i sortir del fitxer prement **CTRL + X**, després prement **Y** i després **ENTER**.

pas 23. Finalment, tot el que hem de fer és iniciar els dos serveis i habilitar-los a **systemctl**. Executeu les dues ordres següents.

```
sudo systemctl unmask hostapd  
sudo systemctl enable hostapd  
sudo systemctl start hostapd  
sudo service dnsmasq start
```

pas 24. Ara per fi hauríem de tenir un punt d'accés sense fil **Raspberry Pi** completament operatiu, podem assegurar-nos que funciona utilitzant qualsevol dels nostres dispositius sense fil i connectant-nos al nostre nou punt d'accés mitjançant l'**SSID** i la frase de contrasenya **WPA** que es vam establir anteriorment al tutorial.

Per assegurar-nos que tot funcioni sense problemes, el millor és provar de reiniciar ara. Això garantirà que tot es torni a habilitar correctament quan s'iniciï una còpia de seguretat del **Raspberry Pi**. Executem l'ordre següent per reiniciar el **Raspberry Pi**.

```
sudo reboot
```

Aquest és un altre gran projecte per al **Raspberry Pi** que es pot ampliar per convertir-lo en una utilitat extremadament útil. Com he esmentat anteriorment, podem convertir-lo en un node d'accés WiFi on podem dirigir tot el trànsit d'Internet a través d'una VPN.

Afegeix a sobre de "exit 0"

```
sudo hostapd /etc/hostapd/hostapd.conf &  
iptables-restore < /etc/iptables.ipv4.nat
```

Annexa B. Arxius de configuració Raspberry Pi Wireless Access Point

Raspberry Pi Access Point

/etc/dhcpcd.conf

```
# static IP lan configuration:  
interface eth0  
static ip_address=192.168.1.3/24  
static routers=192.168.1.1  
static domain_name_servers=8.8.8.8 8.8.4.4  
  
# static IP wlan configuration:  
interface wlan0  
static ip_address=172.16.0.1/16  
nohook wpa_supplicant
```

/etc/wpa_supplicant/wpa_supplicant.conf

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev  
update_config=1  
country=ES
```

servei hostapd ()

/etc/hostapd/hostapd.conf

```
interface=wlan0  
driver=n180211  
  
hw_mode=g  
channel=8  
ieee80211n=1  
wmm_enabled=0  
macaddr_acl=0  
ignore_broadcast_ssid=0  
  
auth_algs=1  
wpa=2  
wpa_key_mgmt=WPA-PSK  
wpa_pairwise=TKIP  
rsn_pairwise=CCMP  
  
# This is the name of the network  
ssid=Cifo-Aula-i32-IoT-2023  
  
# The network passphrase  
wpa_passphrase=123abcd123
```

/etc/default/hostapd.conf

```
DAEMON_CONF=@/etc/hostapd/hostapd.conf@
```

servei dnsmasq (servidor dhcp)

/etc/dnsmasq.conf

```
interface=wlan0
#server=1.1.1.1
dhcp-range=172.16.0.20,172.16.0.49,12h #IP range and
lease time
dhcp-option=option:dns-server,8.8.8.8
```

servei iptables (firewall)

```
Iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

/etc/iptables.ipv4.nat

```
# Generated by iptables-save v1.8.7 on Tue Oct  3 19:16:58 2023
*filter
:INPUT ACCEPT [937:415286]
:FORWARD ACCEPT [8040:7512230]
:OUTPUT ACCEPT [591:63336]
COMMIT
# Completed on Tue Oct  3 19:16:58 2023
# Generated by iptables-save v1.8.7 on Tue Oct  3 19:16:58 2023
*nat
:PREROUTING ACCEPT [0:0]
:INPUT ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
-A POSTROUTING -o eth0 -j MASQUERADE
COMMIT
# Completed on Tue Oct  3 19:16:58 2023
```

servei forward (router)

/etc/rc.local

```
# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
```

servei forward (router)

/etc/sysctl.conf

```
#!/bin/sh -e
#
# rc.local

# Print the IP address
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
    printf "My IP address is %s\n" "$_IP"
fi
```

```
sudo hostapd /etc/hostapd/hostapd.conf & iptable-restore <
/etc/iptables.ipv4.nat
exit 0
```

Raspberry Pi IoT Wireless

/etc/dhcpcd.conf

```
# static IP wlan:
interface wlan0
static ip_address=172.16.170.1/16
static routers=172.16.0.1
static domain_name_servers=80.58.61.254 8.8.8.8 8.8.4.4
```

/etc/wpa_supplicant/wpa_supplicant.conf

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=ES

network={
    ssid="Cifo-Aula-i32-IoT-2023"
    psk="123abcd123"
    scan_ssid=1
    key_mgmt=WPA-PSK
}
```

Annexa C. Llibreries de Python necessàries a la Raspberry Pi

Annexa D. Codi de l'app SmartApp



Instal·lació del programa gestor de paquetes a Python

framework	terminal
pip3	<code>sudo apt install pip</code>

Instal·lació de les llibreries python necessàries per a Tkinter

framework	conda	pip
tkinter	<code>conda install -c notebook tkinter</code>	<code>pip install tkinter</code>