

## Contenido

[Configuración. Archivo de configuración para el SDK](#)

[Para verificar cual es el puerto del POS conectado](#)

[En window](#)

[En Linux](#)

[Documentación de Servicios](#)

[Configuración para correr el SDK por defecto al levantar el sistema operativo](#)

[En windows](#)

[En Linux](#)

[Correr Fronted en Linux](#)

## Configuración. Archivo de configuración para el SDK

Modificar el archivo config.json de la carpeta de instalación:

```
{
  "com": "com5",
  "reintentos": 3,
  "transaction_timeout": 1,
  "server": {
    "port": 3001
  },
  "log": {
    "maxSize": 5,
    "backups": 3,
    "absolutePath": "C:/Users/Jose/logs/application.log"
  },
  "tam":100,
  "delayTime":100,

  "baudRate": 9600,
  "bufferSize": 250,
```

```
"useTCP": true,  
"tcpPOS": {  
    "host": "192.168.1.102",  
    "port": 15000,  
}  
}
```

Donde:

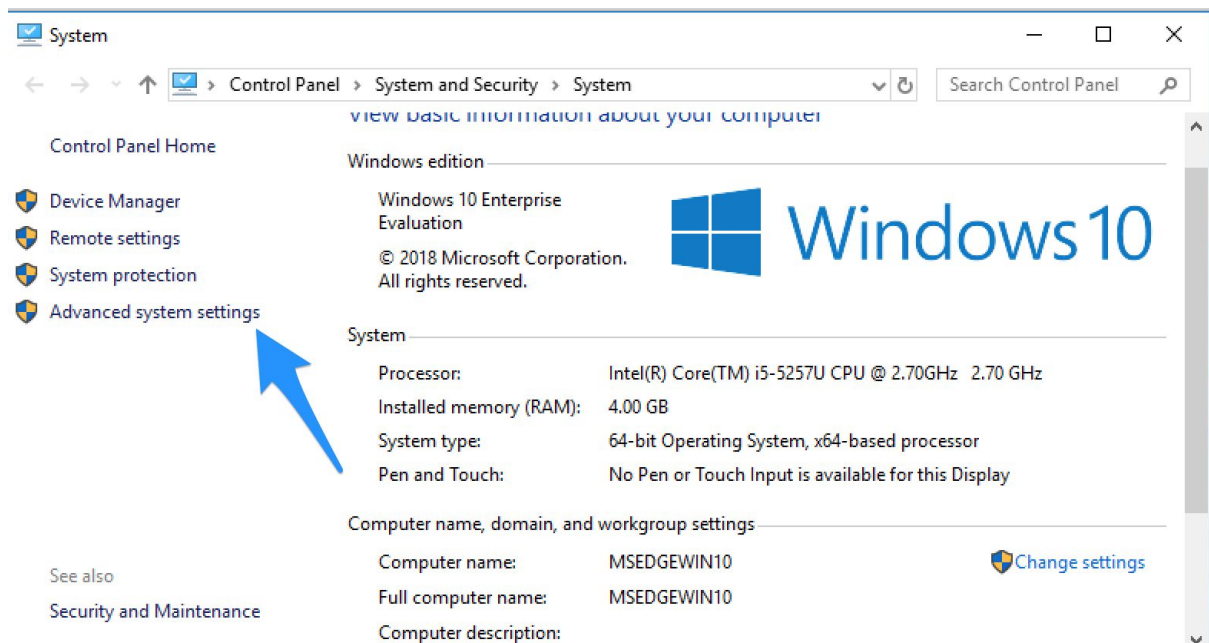
- com: puerto de conexión del dispositivo POS.
- reintentos: número de reintentos de conexión al POS. Esta configuración se sugiere dejar en 3.
- transaction\_timeout: tiempo en minutos que una transacción puede durar antes de asumir que fue cancelada porque el POS se colgó o no se recibió ninguna respuesta.
- port: Puerto donde va a correr el servidor backend. La api de integración se levantará en localhost:port
- log.maxSize: tamaño en MB de los archivos de logs
- log.backups: cantidad de archivos de logs que serán generados
- log.absolutePath: path y nombre del archivo donde se almacenarán los logs de la aplicación.
- tam: tamaño en bytes de cada mensaje que se envía al pos. Si el POS solo puede recibir mensajes de 8bytes y se quiere enviar un mensaje de 16 bytes, se envía en dos partes.
- delayTime: tiempo en milisegundos que espera para enviar entre los mensaje partidos.
- baudRate y bufferSize: son configuraciones propias de la librería serialport que se deben quedar fijos ya que estan adaptados a la configuraciones que puede recibir el POS.
- useTCP: indica si se debe utilizar o no conexión TCP/IP para comunicarse con el POS. Primeramente se valida el valor de los queryParams de las invocaciones, en caso que las invocaciones no especifiquen por queryParam el HOST y el PORT del POS, se verifica este valor de la configuración.
- tcpPOS: si el campo **useTCP** está en **TRUE**, este campo debería tener los datos de conexión del POS.

## Para verificar cual es el puerto del POS conectado

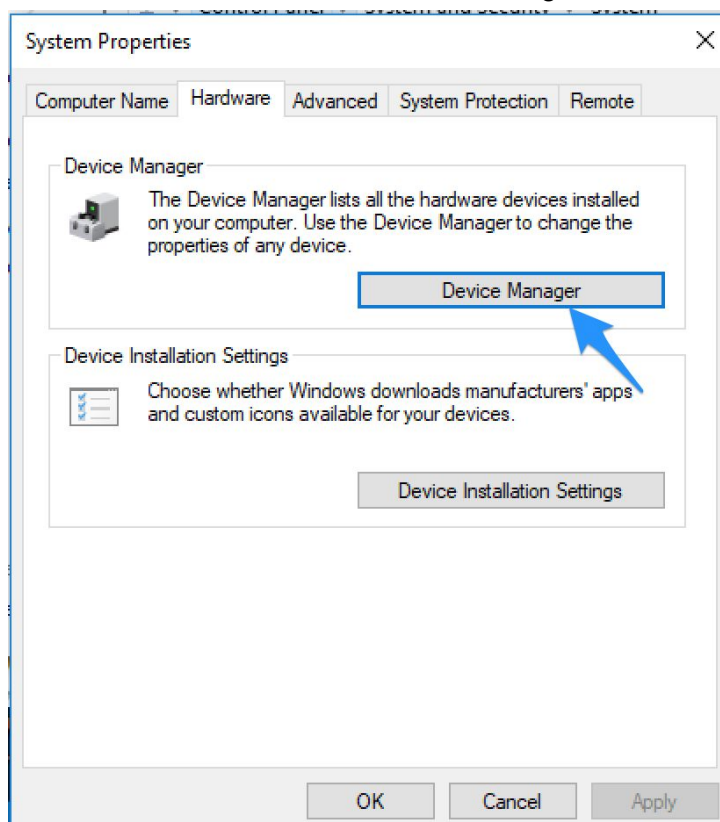
### En window

Conectar el dispositivo POS al puerto usb e instalar los drivers correspondientes.

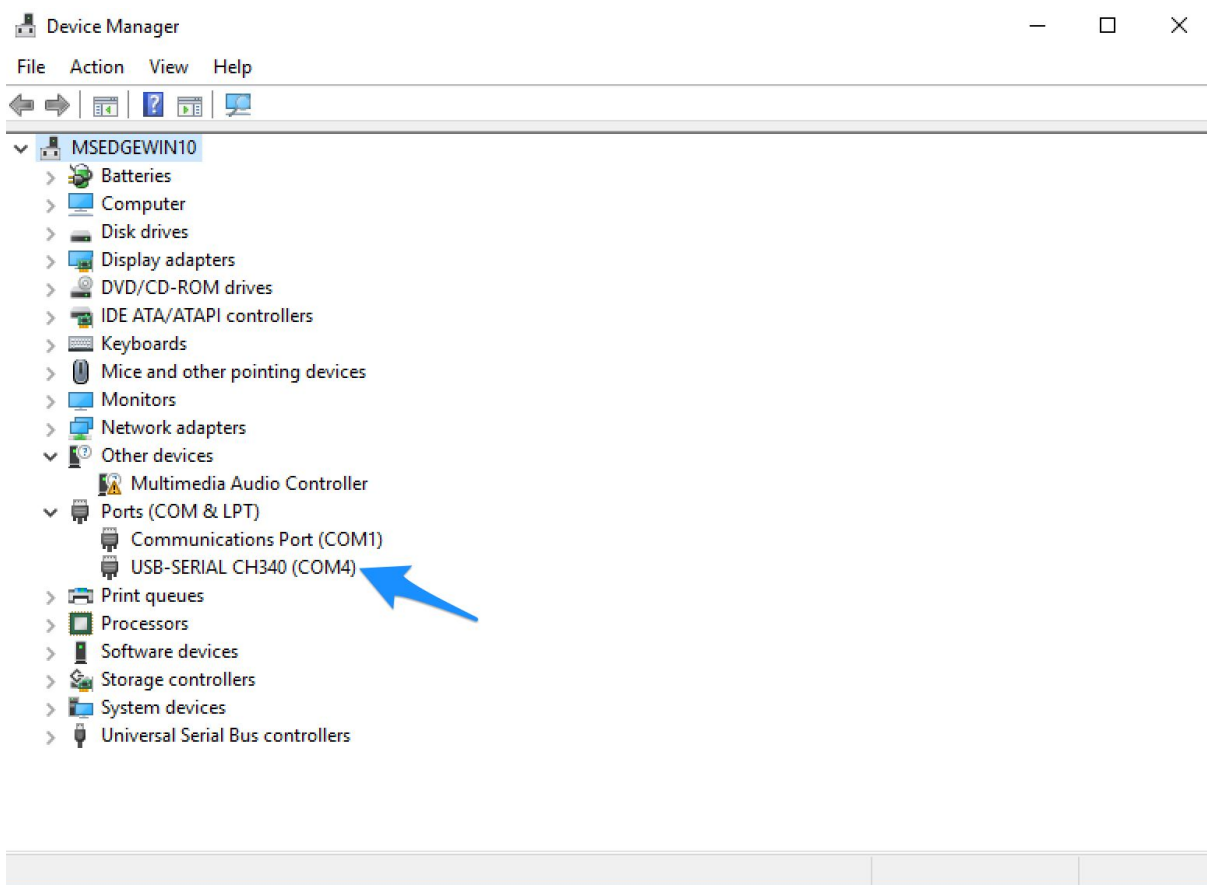
Abrir "Mi PC" → Propiedades → Configuraciones Avanzadas



Seleccionar "Hardware" → "Device Manager"



Ver la opción "Ports" donde se despliegan los dispositivos conectados. El correspondiente a **USB-Serial CH 340** es el dispositivo POS. Por tanto nuestro puerto COM será el 4 según el ejemplo mostrado en la imagen:



## En Linux

Abrir la consola y escribir:

```
dmesg | grep tty
```

Se listan los dispositivos conectados. Utilizaremos el código tty0, tty1, etc según el número de puerto habilitado para el POS

## Documentación de Servicios

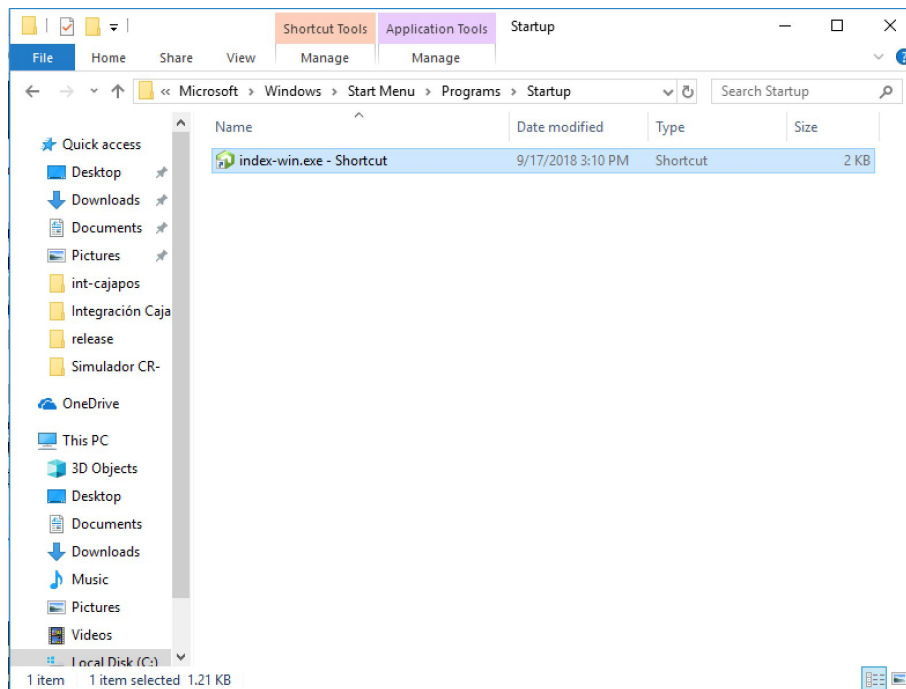
Se pueden encontrar ingresando a la ruta <http://localhost:port> de cualquier navegador una vez que el proyecto backend se encuentre corriendo en la máquina. Se mostrará una página similar a:



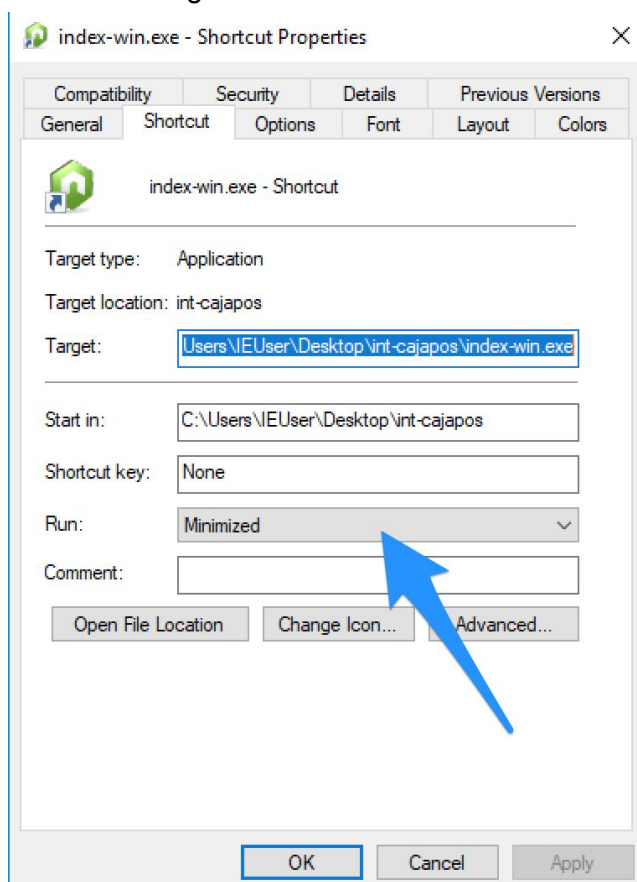
## Configuración para correr el SDK por defecto al levantar el sistema operativo

### En windows

1. Abrir Windows Explorer (Se puede teclear Win+E ).
2. Pegar el siguiente path: %appdata%\Microsoft\Windows\Start Menu\Programs\Startup. Si no funciona, probar con esta opción: %programdata%\Microsoft\Windows\Start Menu\Programs\Startup
3. Abrir en otra ventana la carpeta donde tenemos el .exe del SDK. Click derecho sobre el .exe y seleccionamos "Crear acceso directo".
4. Copiamos el acceso directo a la carpeta que abrimos en el paso 2.



5. Click secundario sobre el acceso directo y seleccionamos "Properties", luego vamos al tab "Shortcut". Ahí cambiaremos la opción "Run" a "Minimized" como se muestra en la figura:

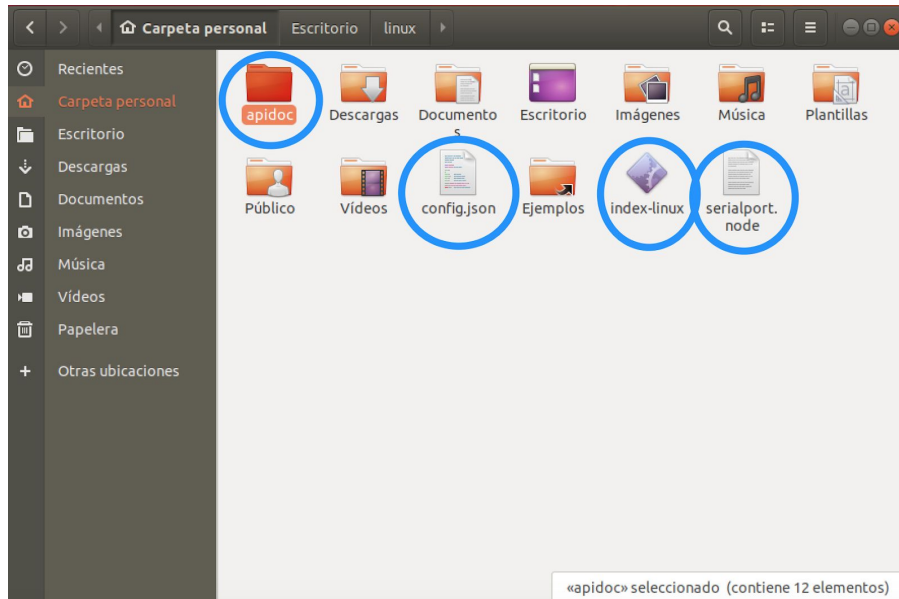


6. Reiniciar la máquina y si todo ha salido bien al iniciar debe levantar nuestro SDK en `http://localhost:port`

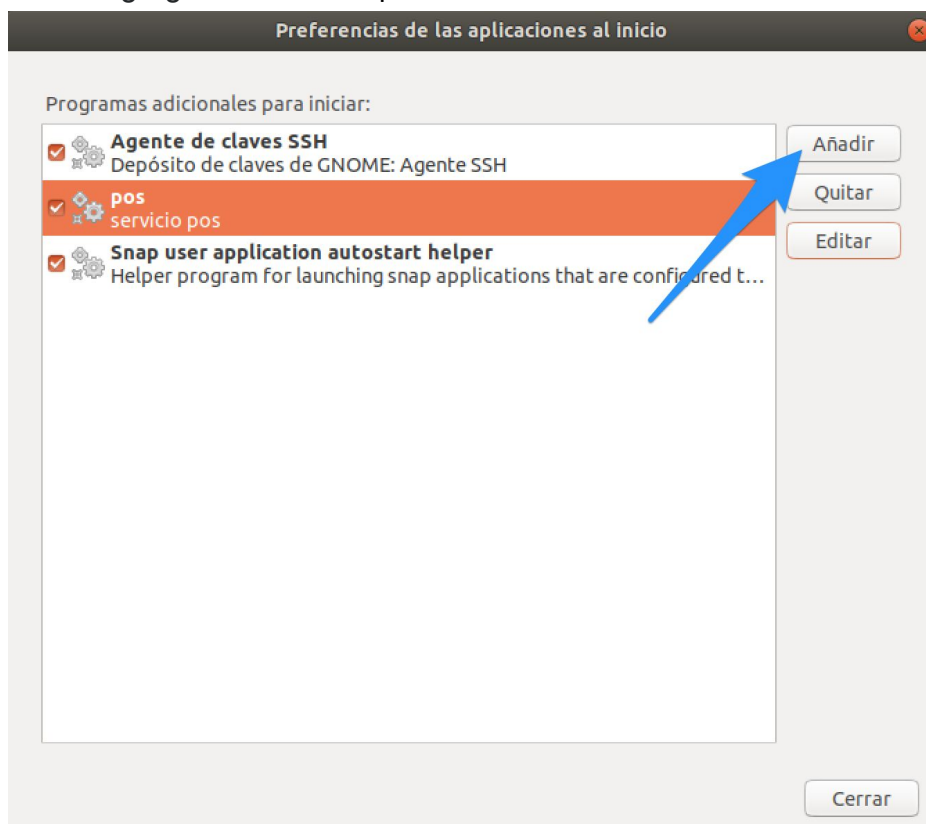
## En Linux

Ref: <https://askubuntu.com/questions/228304/how-do-i-run-a-script-at-start-up>

1. Ubicar los instaladores en una misma carpeta. Por ej: /home



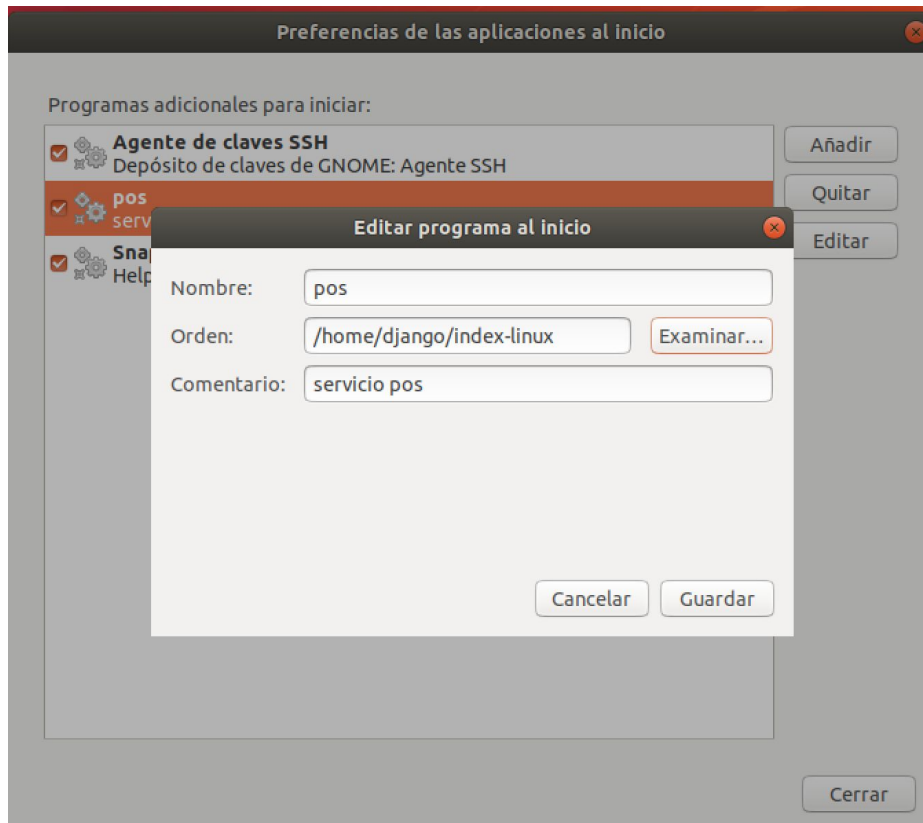
2. Ir a System > Preferences > Startup Applications.
3. Agregar una nueva aplicación



4. Configuramos la nueva aplicación:

Nombre: Pos

Orden: path al ejecutable



- Guardamos los cambios y reiniciamos el sistema operativo para probar que todo haya salido bien.

## Correr Fronted en Linux

Puede dar problemas con algunas librerías. Si al tratar de ejecutar el instalador tenemos el siguiente error:

```
error while loading shared libraries: libgconf-2.so.4: cannot open shared object file: No such file or directory
```

```
django@django-VirtualBox: ~/Escritorio/cajapos-linux-x64
Archivo Editar Ver Buscar Terminal Ayuda
django@django-VirtualBox:~$ cd Escritorio/
django@django-VirtualBox:~/Escritorio$ ls
cajapos-linux-x64 linux virtualenv
django@django-VirtualBox:~/Escritorio$ cd cajapos-linux-x64/
django@django-VirtualBox:~/Escritorio/cajapos-linux-x64$ ./cajapos
./cajapos: error while loading shared libraries: libgconf-2.so.4: cannot open shared object file: No such file or directory
```



Entonces se debe instalar la librería requerida con el siguiente comando desde la terminal:

```
sudo apt -y install libgconf2-4
```