presenta_python

August 17, 2020

1 Python para ingeniería

1.1 Que es Python?

Python es un lenguaje de programación interpretado, open source (código abierto, libre y gratuito), con una gran cantidad de librerías que nos permiten resolver la mayoría de los problemas de cálculo numérico para ingeniería que podamos encontrar.

1.2 Python es parecido a Matlab. ¿Porque usar Python?

Matlab es también un lenguaje que nos permite resolver problemas de cálculo numérico, con un valor de aproximadamente 100 dólares por año la licencia de estudiante.

Es verdad que Matlab dispone de Simulink, un editor gráfico para simulaciones de sistemas dinámicos. Python carece de algo semejante.

En mi opinión, la ventaja definitiva de Python es que podemos usarlo no solo en nuestras notebooks y máquinas de escritorio, sino también en la nube, en Raspberry Pi, Jetson Nano, e incluso en microcontroladores. En forma libre y gratuita.

1.2.1 Python en la nube

1. https://colab.research.google.com/
2. https://www.pythonanywhere.com/

1.2.2

1.3 Python es parte de algo más grande

Gran parte de los usuarios de Python también estamos acostumbrados a usar ciertas herramientas y estilos de trabajo:

1. Git: un sistema de control de versiones
2. Integración y Entrega Continua: una forma de trabajar en el desarrollo, implementación, testeo y puesta en marcha de sistemas donde la evolución del sistema se da en pequeños pasos, graduales pero muy rápidos.

1.4 Como podemos usar Python (y Git) en electrónica

Para concretar, supongamos que estamos trabajando en una experiencia de laboratorio en forma grupal: Tienen que diseñar e implementar un circuito electrónico, realizar un experimento, y tomar medidas de tensiones y corrientes. Estas medidas experimentales deben ser procesadas y comparadas con los resultados de una simulación.

El reporte final debe incluir diagramas y planos del circuito electrónico, un archivo con la simulación y las mediciones, gráficos, y textos de análisis y conclusiones.

1.4.1 Arduino y Raspberry Pi

1. Podemos hacer sistemas de adquisición de datos y de generación de señal con un Arduino conectado a una Raspberry Pi.

2. Si bien es conveniente programar el Arduino en processing (el lenguaje de programación "de fábrica" de los Arduinos), también podemos hacerlo en MicroPython (ver https://realpython.com/arduino-python/ y https://micropython.org/)

3. Recordemos que la Raspberry Pi dispone del sistema operativo Raspbian, una variante de la distribución Debian de Linux. Por lo tanto puede programarse en varios lenguajes, por ejemplo C y C++ además de Python. La programación en Python se ve simplificada por la disponibilidad de la librería pySerial que nos permite usar la puerta serie para una comunicación bidireccional con la placa Arduino.

4. Por lo tanto, es posible recibir mediciones y enviar comandos entre placas Arduinos y Raspberry Pi como podemos ver en https://www.raspberrypi.org/forums/viewtopic.php?t=97368.

1.4.2 Raspberry Pi

1. Estamos acostumbrados a que nuestras computadoras tienen un monitor, teclado y ratón. Sin embargo, en nuestro presente y en el futuro inmediato, la nube (the cloud) e Internet de las cosas (IOT) son dos temas muy importantes.

2. Tanto la nube como Internet de las cosas están formadas por computadoras que no tienen ni monitor ni teclado. Es lo que se llama headless mode. ¿Como accedemos a esas computadoras? Necesitamos emplear Internet.

3. También podemos trabajar con las Raspberry Pi en headless mode, es decir con acceso a través de Internet, empleando un protocolo que se llama SSH (ver https://en.wikipedia.org/wiki/Secure_Shell).

4. Otra posibilidad es implementar un pequeño servidor web empleando la librería Flask (ver https://flask.palletsprojects.com/en/1.1.x/) que nos permite interactuar con la Raspberry Pi a través de un sitio web.

5. También podemos programar en Python bots para Telegram (ver https://github.com/python-telegram-bot/python-telegram-bot) y Twitter (ver https://realpython.com/twitter-bot-python-tweepy/)

1.5 En nuestra computadora

1. En una notebook o computadora de escritorio podemos instalar Python empleando el instalador Anaconda (ver https://www.anaconda.com/), que resuelve todas las dificultades de instalar y actualizar Python, tanto en Linux como en Windows y Mac.

2. En Python existe la librería sshfs (Ver https://pypi.org/project/fs.sshfs/), que nos permite emplear el protocolo SSH para conectarnos con otras computadoras, en nuestro caso la Raspberry Pi. Con sshfs podemos intercambiar archivos con la Raspberry Pi.

3. Además, también disponemos de las librerías para bots de Telegram y Twitter, y para servidores web.

4. Podemos también implementar clientes web programables (o sea navegadores o browsers) empleando las librerías requests (ver https://requests.readthedocs.io/en/master/) y selenium (ver https://selenium-python.readthedocs.io/)

1.5.1 ¿Que queremos hacer?

Vamos a usar la placa Arduino para generar una señal de entrada al circuito, tomar medidas y transmitirlas a la Raspberry Pi, donde vamos a mostrar los resultados del experimento en una página web, además de preparar archivos con datos para ser descargados a través de la página web o a través de sshfs.

1.6 Volviendo al problema de realizar el experimento, procesar datos y escribir el informe

1. Necesitamos programar la placa Arduino, la Raspberry Pi y nuestra computadora. Vamos a tener que diseñar e implementar varios programas, probarlos por separado, corregirlos, probarlos en conjunto, corregirlos, y después mostrar el sistema funcionando a los profesores que seguramente van a decir "Porque no cambian esta parte..."

2. Además, cuando todo esté listo, los profesores seguramente van a pedir que el sistema que funcionaba a la perfección en los equipos de los alumnos, ahora funcione igualmente en placa Arduino, la Raspberry Pi y una computadora del Departamento de Electrónica. En otras palabras, hay que entregar o desplegar el sistema.

3. Parece que vamos a escribir varios programas, hacer planos de circuitos, correr simulaciones y además hacer correcciones. Por lo tanto, vamos a tener variantes de cada archivos, cada uno con pruebas distintas o parámetros distintos. Estas variantes son las versiones de cada archivo. Además, tenemos que compartirlos con todo el grupo. ¿Como podemos asegurarnos que cada uno tiene la versión correcta de cada archivo en cada momento? Para esto usamos un sistema denominado Git, que podemos usar en forma local o junto a un servidor remoto (ver https://github.com/ y https://bitbucket.org/).

4. Un punto importantísimo para nosotros, si pensamos trabajar con Internet de las cosas, es usar Git para mover los programas escritos en nuestra computadora hasta los dispositivos. En nuestro caso, podemos usar Git para pasar los programas desde nuestra computadora hasta un servidor como Github o Bitbucket. Entonces, conectándonos por SSH con nuestra Raspberry Pi, podemos usar un cliente Git para descargar los archivos del servidor. LLegado el caso, incluso **es posible automatizar la descarga de archivos empleando Python**.

5. Además, el otro punto importante para un desarrollo como este, es automatizar también el testeo de las distintas funciones del sistema. **La automatizacion del testeo de los programas** puede hacerse con distintas librerias de Python (ver https://docs.python.org/3/library/unittest.html y https://coverage.readthedocs.io/en/coverage-5.2.1)

6. La automatización de testeo no tiene que limitarse al software. También es posible hacerlo con circuitos que implementemos. Mediante cálculo y simulación, podemos conocer cuál es la respuesta de cualquier circuito frente a una entrada determinada. A través de la Raspberry Pi y la placa Arduino, podemos excitar al circuito en estudio con una entrada determinada y medir su respuesta. Mediante un programa escrito en Python podemos ver si la salida real se asemeja a la esperada.

1.6.1 Automatizar todo lo que sea posible

1. Cuando tenemos que escribir un informe o artículo, las partes que tenemos que escribir necesariamente nosotros son análisis, conclusiones y bibliografía. Todo lo que sea tabla de parámetros, configuración, datos, imágenes e información que pueda ser almacenada en una base de datos debe manejarse mediante programa. Existen librerías que permiten generar reportes en forma automática mediante formatos (templates). Así es posible leer y escribir archivos pdf, html, xlsx, csv y tex con un programa Python.

2. Además, con un programa Python es posible interactuar con servidores de correo electrónico, Telegram, Twitter, WhatsApp, sitios web, y enviar y recibir SMS a través de Twillio.

1.6.2 Python en sistemas automatizados de diseño electrónico

Existen sistemas de diseño de circuitos integrados que incorporan Python como una herramienta para procesar sistemas con gran cantidad de componentes. Si bien existen componentes básicos diseñados por personas, el diseño de circuitos integrados con miles de millones de transistores solo puede hacerse automatizando tareas. Además, la búsqueda de valores adecuados para los distintos parámetros de un sistema debe hacerse necesariamente con librerías de cálculo numérico especializado (ver https://numpy.org/ y https://matplotlib.org/).

Ver también https://www.tdx.cat/bitstream/handle/10803/457967/mjllr1de1.pdf

1.7 Resumiendo

Creo que Python, Git, integración y despliegue continuo son totalmente necesarios en ingeniería electrónica hoy y en el futuro cercano. Por eso he preparado un curso virtual introductorio sobre estos temas, listo para comenzar la primera semana de Septiembre de 2020.

1.8 Adicionales: Inteligencia Artificial

Es posible desarrollar aplicaciones de inteligencia artificial totalmente en Python (ver https://www.tensorflow.org/ y https://pytorch.org/).)

```
In [1]: ##
```