



ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

FACULTAD DE INGENIERIA EN ELECTRICIDAD Y  
COMPUTACIÓN

---

## Parsing XML with Python

---

*Autor:*

Adriana RODRÍGUEZ

*Profesor:*

Ing. Javier TIBAU

18 de febrero de 2014



# Índice general

|                   |    |
|-------------------|----|
| 1. Introducción   | 5  |
| 2. Alcance        | 7  |
| 3. Descripción    | 9  |
| 4. Implementación | 11 |
| 5. Observaciones  | 13 |
| 6. Conclusiones   | 15 |



# Capítulo 1

## Introducción

XML es un lenguaje de marcado que permite almacenar datos, para ello esta conformado de una estructura abstracta para que así se puedan trabajar con datos grandes.

En este caso, el propósito de este documento es dar a conocer como en el lenguaje Haskell se puede trabajar para la administración de archivos XML y así mostrar datos que sean específicos para el usuario.

Para llevar un mejor control en los datos, decidimos trabajar por medio del *parsing* que permite que los datos sean trabajados de una manera mas textual en la observación de todo el documento en si.



Figura 1.1: Logo de Python.

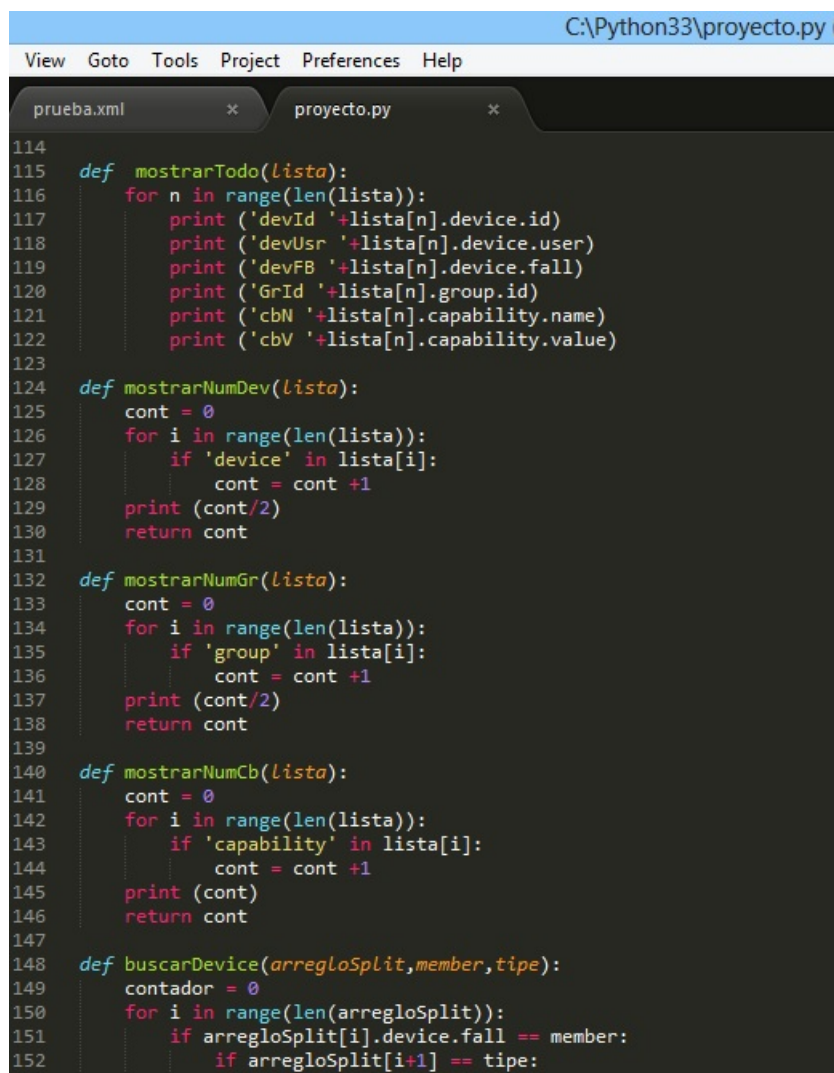


## Capítulo 2

### Alcance

El alcance del proyecto es poder realizar un procesamiento de la información que está contenida en el archivo *wurfl23.xml* y mostrar datos correspondientes a los dispositivos que cumplan con ciertas características dichas por el usuario.

En este caso el parseo se lo llevará a cabo por medio del lenguaje python.



The image shows a screenshot of a Python IDE window titled "C:\Python33\proyecto.py". The window has a menu bar with "View", "Goto", "Tools", "Project", "Preferences", and "Help". Below the menu bar, there are two tabs: "prueba.xml" and "proyecto.py". The "proyecto.py" tab is active, showing the following Python code:

```
114
115 def mostrarTodo(lista):
116     for n in range(len(lista)):
117         print ('devId '+lista[n].device.id)
118         print ('devUsr '+lista[n].device.user)
119         print ('devFB '+lista[n].device.fall)
120         print ('GrId '+lista[n].group.id)
121         print ('cbN '+lista[n].capability.name)
122         print ('cbV '+lista[n].capability.value)
123
124 def mostrarNumDev(lista):
125     cont = 0
126     for i in range(len(lista)):
127         if 'device' in lista[i]:
128             cont = cont + 1
129     print (cont/2)
130     return cont
131
132 def mostrarNumGr(lista):
133     cont = 0
134     for i in range(len(lista)):
135         if 'group' in lista[i]:
136             cont = cont + 1
137     print (cont/2)
138     return cont
139
140 def mostrarNumCb(lista):
141     cont = 0
142     for i in range(len(lista)):
143         if 'capability' in lista[i]:
144             cont = cont + 1
145     print (cont)
146     return cont
147
148 def buscarDevice(arregloSplit,member,tipe):
149     contador = 0
150     for i in range(len(arregloSplit)):
151         if arregloSplit[i].device.fall == member:
152             if arregloSplit[i+1] == tipe:
```

Figura 2.1: Código en Python.





# Capítulo 3

## Descripción

El objetivo principal del proyecto es apreciar y entender las formas de programar en python. Python es un lenguaje interpretado, todo trabaja con todo. Se puede decir que la manera de trabajar en python es mas sencilla en comparacion de Haskell ya que existe muchas herramientas que nos hacen posible el desarrollo del problema.

Los archivos XML trabajan por medio del marcado, llevan un control de los grupos de información para asi mostrarlos como *elementos* que proporcionan una lista de estos elementos. El archivo antes mencionado, está conformado por una serie elementos principales, estos son: Device, Group y los capability que son las listas principales que nos ayudarán a manipular los datos.

En el caso de los Device están conformado por:

- id
- user agent
- fall back

A su vez, los Group están conformado por:

- id

Y por último los capability están conformado por:

- name
- value



# Capítulo 4

## Implementación

El proyecto esta conformado por funciones y clases. Las clases estan conformadas por: Device, Group, Capability y General. Que contiene informacion de los atributos de cada tag en particular.

Las funciones son las encargadas de hacer la limpieza y la estructura de la informacion que deseamos manipular. Para este caso nos ayudamos mucho de varias funciones implementadas como son el replace, translate entre otros.



# Capítulo 5

## Observaciones

Cabe recalcar que para la realizacion de este proyecto se tuvo que reordenar los conocimientos, ya que en mi opinion Python trabaja de una manera muy diferente a otros lenguajes.

Uno de los problemas que tuve fue el ordenamiento creado al momento de crear las funciones, separandose con enter y no con tab.



# Capítulo 6

## Conclusiones

Python es útil en varios aspectos como:

- Simplifica mucho la programación.
- Las librerías que más necesitas ya están dentro del código.
- Es un lenguaje ordenado.