Linux
Málaga
@linux_malaga
www.linux-malaga.org

Taller avanzado de Python

Juan Miguel Taboada Godoy
@juanmitaboada
https://www.linkedin.com/in/juanmitaboada

Juan José Soler Ruiz
@soleronline
http://es.linkedin.com/in/soleronline

Roberto Becerra García
@idertator
https://www.linkedin.com/in/idertatorº

Bienvenido - Welcome - Witam

Centrologic

Bio

Juan Miguel Taboada Godoy (1980 - …)

1996 – Primer ordenador y primera LAN (coaxial)

1999 – Universidad de Málaga y **Linux Málaga**

2001 – Investigación en la UMA

- Cluster de computación masiva
- Servidores y hosting
- Mercados bursátiles
- Beca Neurociencia en New York

2005 – Axargua (Adquisición de datos industriales)

2008 – Pontgrup Correduría de Seguros

2011 – Bética fotovoltáicas (Adquisición de datos para Red Eléctrica España)

2012 – Centrologic

Juan José Soler Ruiz

2001 – CFGS Administración Sistemas Informáticos

2003 – Primer premio en el concurso "Javier Benjumea"
- Adquisición de datos con Visual Basic

2003 – Montaje y configuración de:
"Cluster Heterogéneo de Computadoras"
bajo SO Red Hat 7.2.

2005 – STEA Telemática
- Desarrollador y analista de sofware

2007 – Primer CRM en PHP

2010 – Bética fotovoltáicas
- Administrador de sistemas
- Desarrollador y analista de software

2010 – Opositometro (Desarrollador web)

2012 – Centrologic

Centrologic

Linux
Málaga

python™

Bio

Roberto Antonio Becerra García (1986 - …)

2001 – Primer programa en Pascal

2004 – Medalla de Oro en concurso nacional de Computación de Cuba e integrante de la Preselección nacional a participar en la IOI

2009 – Finalista del concurso internacional de Programación para universitaroios ACM-ICPC

2010 – Construcción de una plataforma de procesamiento de movimientos oculadores

2011 – Profesor de Arquitectura de Ordenadores e Inteligencia artificial en la Universidad de Holguin

2012 – Doctorado en la UMA en procesamiento de movimientos oculares

2017 – Centrologic

Centrologic

Linux
Málaga

python™

Linux Málaga

Linux Málaga

Historia

Junio 1998 (Campus Party '98)

Mayo 1999 (Legal)

Noviembre 2003 (Final juvenil)

Y nuevo comienzo

Meetup

674 inscritos y 17 eventos celebrados

5 eventos programados en 2017

Contacto

@linux_malaga

www.linux-malaga.org

Centrologic

Linux
Málaga

python™

Open South Code

Edición 2017 - MAYO

Viernes día 5

- 3 salas (aforo 25 personas/sala)

Sábado día 6

- 40 charlas (8 charlas por sala)
- Stand específico
- Mesas de exposiciones
- Posible catering

Año 2016:

16 charlas y 5 talleres

Año 2017:

40 charlas planificadas

Centrologic

Centrologic

python™

django

A

B

UI BOOTSTRAP

HTML 5

3

Centrologic

CODENERIX

python™

django

B

UI BOOTSTRAP

HTML 5

3

python™

AVANZADO

Centrologic

Linux
Málaga

python™

¿Quién? ¿qué? ¿por qux! ¡pldzo1oaxd? ¡xadfsl! #$a

Guido van Rossum

Centrum Wiskunde & Informatica (Paises Bajos)

Finales de los '80

Humoristas Monty Python

1991 :: 0.9.o (PZÑsfa)

pofpd :: f.w (pzosñasf)

asdf1f :: p.ñ k p.j

fasf :: fjf.asf y faf.a (Jfalskf)

fdfda :: 2.Z

fkaq :: X.Y

Lenguaje interpretado

Sintaxis favorf fdljf

Masfd Ikasdf aer

Tsad Ikjafqknba

Piflanewer => FYA

Centrologic

Linux Málaga

python™

¡Vamos al lío!

Centrologic

Linux
Málaga

python™

Programación funcional (1)

Problema

lista = [1, 4, 3, 2, 5, 7, 2 ,9, 8, 7, 8, 1]

Crear una función que retorne los cubos de los elementos impares de la lista.

Programación funcional (2)

Solución tradiccional

```
resultado = []
for x in lista:
    if x % 2 != 0:
        resultado.append(x * x * x)
print resultado
```

Programación funcional (3)

Solución tradiccional

```
resultado = []
for x in lista:
    if x % 2:
        resultado.append(x ** 3)
print resultado
```

Programación funcional (4)

Utilizando generadores

```
def impares(lista):
    for x in lista:
        if x % 2:
            yield x
resultado = []
for x in impares(lista):
    resultado.append(x**3)
print resultado
```

Programación funcional (5)

Utilizando map

```
def impares(lista):
    for x in lista:
        if x % 2:
            yield x
def cubo(x):
    return x ** 3
print map(cubo, impares(lista))
```

Programación funcional (6)

Utilizando map+lambda

```
def impares(lista):
    for x in lista:
        if x % 2:
            yield x
print map(lambda x: x ** 3, impares(lista))
```

Programación funcional (7)

Utilizando comprensión de listas

```
def impares(lista):
    for x in lista:
        if x % 2:
            yield x
print [x ** 3 for x in impares(lista))
```

Programación funcional (8)

Utilizando comprensión de listas con filtrado

```
print [x ** 3 for x in lista if x % 2]
```

Centrologic

Linux Málaga

python™

Clases y métodos (9)

Clase básica

```
class A(object):
    attr = None
    def __init__(self):
        self.attribute = True
    def foo(self,x):
        print "executing foo(%s,%s)"%(self,x)
    @staticmethod
    def static_foo(x):
        print "executing static_foo(%s)"%x a=A()
```

Centrologic

Linux
Málaga

python™

Herencia (10.1)

```
class A(object):
    def a(self):
        print "Soy a y estoy en la clase A"

    def b(self):
        print "Soy b y estoy en la clase A"

    def c(self):
        print "Soy c y estoy en la clase A"

class A1(object):
    def c(self):
        print "Soy c y estoy en A1"
        super(A1, self).c()
```

Herencia (10.2)

```
class B(A):
    def b(self):
        print "Soy b y estoy en la clase B"



class C(A1, B):
    def c(self):
        print "Soy c y estoy en la clase C"
        super(C, self).c()

a = A()
b = B()
c = C()
```

@property (11.1)

```
class Celsius:
    def __init__(self, temperature = 0):
        self.temperature = temperature

    def to_fahrenheit(self):
        return (self.temperature * 1.8) + 32

    @property
    def temperature(self):
        print("Getting value")
        return self._temperature
```

@property (11.2)

```
    @temperature.setter
    def set_temperature(self, value):
        if value < -273:
            raise ValueError("Temperature below -273 is not possible")
        print("Setting value")
        self._temperature = value

c = Celsius()
print c.temperature
c.temperature = 2
print c.temperature
print c.to_fahrenheit()
```

Indexable (12.1)

```
from os import listdir
from os.path import isfile, join

class Folder(object):
    def __init__(self, folder):
        self.folder_name = folder
        self.files = [f for f in listdir(self.folder_name) if isfile(join(self.folder_name,
f))]
    def __getitem__(self, key):
        self.read_file(key)
    def get_files(self):
        return self.files

    def read_file(self, file):
        path_file = "{}/{}".format(self.folder_name, file)
        with open(path_file) as f:
            print f.read()
```

Indexable (12.2)

```
ruta = '/tmp'

directorio = Folder(ruta)

print directorio.get_files()

directorio['jojo.txt']
```

DUDAS

Centrologic

Linux
Málaga

python™

Linux
Málaga
@linux_malaga
www.linux-malaga.org

Muchas Gracias

Juan Miguel Taboada Godoy
@juanmitaboada
https://www.linkedin.com/in/juanmitaboada

Juan José Soler Ruiz
@soleronline
http://es.linkedin.com/in/soleronline

Roberto Becerra García
@idertator
https://www.linkedin.com/in/idertatorº

Thank you - Dziękujemy

Centrologic