

# Python en el mundo real

---

## Los disertantes

Juan B Cabral

- Ing. en Sistemas.
- UTN-FRC
- Organizador de la última PyCon Argentina, de un PyDay y un Django Day.
- Trabaja en Liricus.

Matías Herranz

- Lic. en Ciencias de la Computación.
  - FAMaF - UNC
  - Disertante en todo lo que organizo Juan y varios eventos mas.
  - Trabaja en Machinalis
- 

## Los 2:

- Ambos trabajan con Python en su día a día.
  - Ambos hicieron su tesis de grado en Python.
  - Ambos son miembros de PyAr.
  - Ambos hacen ciencia con Python.
  - Ambos se conocieron en PyAr.
  - Toman whiskey.
  - Son amigos.
- 

## ¿De qué va la charla?

- Ejemplos de su sintaxis en casos didácticos
- Programas hechos en Python.
- Empresas que apuestan fuerte por (¡y ganan fuerte con!) Python
- Comunidad de Python
- Frameworks hechos en Python

Y vamos a hablar de plata todo el tiempo!



---

## Y Python...

- Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis muy limpia y que favorezca un código legible.
  - Lo crea Guido Van Rossum en 1991.
  - Multiparadigma.
  - Multiplataforma
  - Comunidad MUY activa y con... una particularidad en el nivel técnico.
  - Multiples implementaciones: Jython, IronPython, PyPy, CPython...
- y... es simple (según las estadísticas todo es 1/3 de Java y C#)
- 1/3 de tiempos de desarrollo.
  - 1/3 de costo de mantenimiento (producto de que sea fácil de leer y entender).
  - 1/3 de costos.
  - ¡1/3 de time to market!
- 

## Por cierto...

- Estos slides estan generados con Landslide (que está hecho con Python)
- Homepage: <https://github.com/adamzap/landslide>

# Ejemplos

En los siguientes slides se verá:

- Librerías interesantes de Python.
  - Ejemplos de sintaxis.
  - Ejemplos de código con esas librerías (funcionando).
  - Un video para ilustrar computación científica (con código un poco más complejo)
- ¡Recuerden!** Todo esto tiene un fin demostrativo, son pequeños códigos de ejemplo.

## PyQt



- Es un binding a una librería gráfica hecha en C++ (en realidad es más que eso).
  - Posee un editor visual de formularios (QtDesigner)
  - Homepage: <http://www.riverbankcomputing.co.uk/software/pyqt/intro>
-

## PyQt (Ejemplo)

```
import os
from PyQt4 import QtCore, QtGui
import pycante

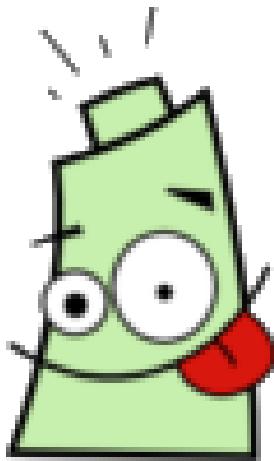
PATH = os.path.abspath(os.path.dirname(__file__))
UI_DIR = pycante.EDir(PATH)

class Dialog(QtGui.QDialog):
    def on_pushButton_clicked(self):
        text = self.lineEdit.text()
        self.label.setText("Hola " + unicode(text) + " !")

app = QtGui.QApplication([])
d = Dialog()
d.show()
app.exec_()
```

---

## Pilas



# Pilas

- Un motor de videojuegos 100% en español (en proceso de traducción a otros lenguajes)
  - Multiplataforma y ridículamente-fácil de aprender.
  - Desarrollado por un miembro de PyAr (Hugo Ruscitti)
  - Embebible en widgets PyQt
  - Homepage: <http://www.pilas-engine.com.ar/>
-

## Pilas (Ejemplo)

```
import pilas
import time

pilas.iniciar()

mono = pilas.actores.Mono()
mono.x, mono.y = 100, 100
mono.aprender(pilas.habilidades.Arrastrable)

bananas = pilas.actores.Banana() * 10
bombas = pilas.actores.Bomba() * 5

def mono_come_banana(mono, banana):
    mono.sonreir()
    banana.eliminar()

def bomba_mata_mono(mono, bomba):
    bomba.explotar()
    mono.gritar()
    mono.eliminar()

pilas.escena_actual().colisiones.agregar(mono, bananas, mono_come_banana)
pilas.escena_actual().colisiones.agregar(mono, bombas, bomba_mata_mono)

pilas.ejecutar()
```

# Beautiful Soup



- Es un parser HTML/XML laxo (funciona bien con código HTML/XML mal formado!).
  - Permite buscar de una manera **muy**, realmente **muy** flexible en código XML-like.
  - Es raro.
  - Hace cosas raras.
  - Homepage: <http://www.crummy.com/software/BeautifulSoup/>
-

## Beautiful Soup (Ejemplo, HTML)

```
html = """
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
    <title>titulo</title>
</head>
<body>
    <p>este es un codigo feo<b>
    <a href="http://google.com">to google</a>
    <a href="http://www.google.com" id="unico">to google</a>
    <a href="http://yahoo.com">to yahoo</a>
    <ul class="elgroso">
        <li>something</li>
    </ul>
    <ul class="elgroso">
        <li>something</li>
    </ul>
    <ul class="elflaco">
        <li>something</li>
    </ul>
</body>
</html>
"""
```

---

## Beautiful Soup (Ejemplo, Python)

```
import re
import bs4

soup = bs4.BeautifulSoup(html)
print soup.find_all("a")
print soup.find("a", href="http://yahoo.com")
print soup.find_all("a", href=re.compile(".*google[.]com"))
print soup.find("a", href=re.compile(".*google[.]com"), id="unico")
print soup.find("ul", class_="elgroso").find_all("li")
```

## Django



- Framework web con más de 7 años de uso, hecho en Python
- Miles y miles de apps para Django, listas para usar (django-registration, pinax, django-bootstrap-toolkit, etc.)
- Muy amplia difusión y uso, muy mantenido, amplia comunidad
- Ahorro de trabajo repetitivo

- Admin interface
  - MVC (model, view, controller)
  - <https://www.djangoproject.com>

# Pequeño ejemplo Django

# Un Microblog

- Usando django-commandline-extensions, django-bootstrap-toolkit, django
  - Muy poco código, una app 100% funcional

## Django (Ejemplo: Models)

## Django (Ejemplo: Views 1/2)

```
# -*- coding: utf-8 -*-
from django.contrib.auth.decorators import login_required
from django.shortcuts import render_to_response
from django.template import RequestContext

from posts.models import Post
from posts.forms import PostForm

@login_required
def show_timeline(request):
    return render_to_response(
        'posts/timeline.html',
        RequestContext(
            request,
            {'posts': Post.objects.order_by( '-date_posted' )})
    )
```

---

## Django (Ejemplo: Views 2/2)

```
@login_required
def write_post(request):
    layout = 'vertical'

    if request.method == 'POST':
        form = PostForm(request.POST)
        if form.is_valid():
            new_post = form.save(commit=False)
            new_post.author = request.user
            new_post.save()
            return show_timeline(request)
    else:
        form = PostForm()

    return render_to_response('posts/form.html', RequestContext(request, {
        'form': form,
        'layout': layout,
    }))
```

---

## Django (Ejemplo: Templates)

```
{% extends "posts/base.html" %}

{% load bootstrap_toolkit %}

{% block content %}
    <h1>Hi! Take a look at the current posts below:</h1>

    {% for post in posts %}
        <div class="well">
            <h1>Author: {{ post.author.username }}</h1>
            <p>{{ post.message }}</p>
        </div>
    {% endfor %}
{% endblock %}
```

---

## numpy / scipy



- Librería para computación científica
  - Core con partes altamente performantes, hechas en C
  - Interfaz 100% Python
-

## numpy / scipy (Ejemplo)

```
def isodata_classification(img, parameters=None):
    global K, I, P, THETA_M, THETA_S, THETA_C, THETA_O, k
    initialize_parameters(parameters)
    N, M = img.shape # for reshaping at the end
    img_flat = img.flatten()
    clusters_list = np.arange(k) # number of clusters availables

    centers = initial_clusters(img_flat, k, "linspace")
    for iter in xrange(0, I):
        last_centers = centers.copy()
        img_class_flat, dists = vq.vq(img_flat, centers)
        centers, clusters_list = discard_clusters(
            img_class_flat, centers, clusters_list)
        centers, clusters_list = update_clusters(img_flat,
            img_class_flat, centers, clusters_list)
        k = centers.size
        if k <= (K / 2.0): # too few clusters => split clusters
            centers, clusters_list = split_clusters(img_flat,
                img_class_flat, centers, clusters_list)
        elif k > (K * 2.0): # too many clusters => merge clusters
            centers, clusters_list = merge_clusters(img_class_flat,
                centers, clusters_list)
        else: # nor split or merge are needed
            pass
    k = centers.size
```

---

## numpy / scipy (Ejemplo)

- (video)
- Isodata: algoritmo de clasificación
- Usos posibles: detección de plagas, cultivos, áreas más húmedas, etc.

## Proyectos

En los siguientes slides se verá:

- Proyectos grandes y pequeños desarrollados por los oradores, en Python.
- Funcionamiento de los proyectos con ejemplos rápidos.
- Resumen de "que" tecnología usa cada proyecto.
- Listado de algunos proyectos famosos que usan Python.
- Un resumen de empresas que apoyan Python

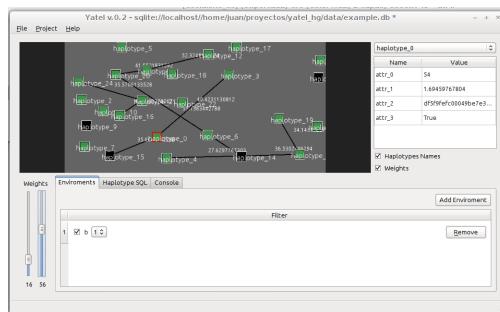
## Yatel



Artwork Based on: <http://ipadwallpapergallery.com/calmity-by-pr09studio/> - cc-by-nc-sa

- Es una herramienta de data-mining.
  - Se utiliza para análisis de perfiles genéticos en el INTA-IFIve.
  - Utilizable como librería.
  - 100% Python
  - Utiliza: PyQt, Pilas, Peewee, IPython.
- 

## Yatel



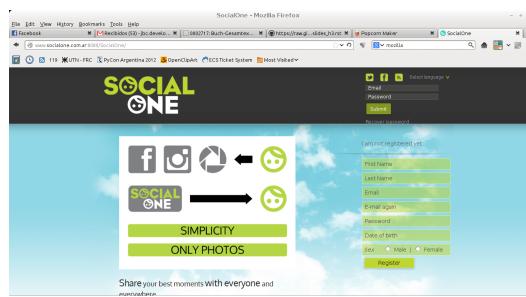
---

## SocialOne



- Es una solución para compartir fotos de manera centralizada.
  - Esta alojado en <http://www.socialone.com.ar> (cuenta con un cliente Android)
  - Backend 100% Python, comunicación rest/json
  - Utiliza: Flask, Python Image Library (PIL)
- 

## SocialOne

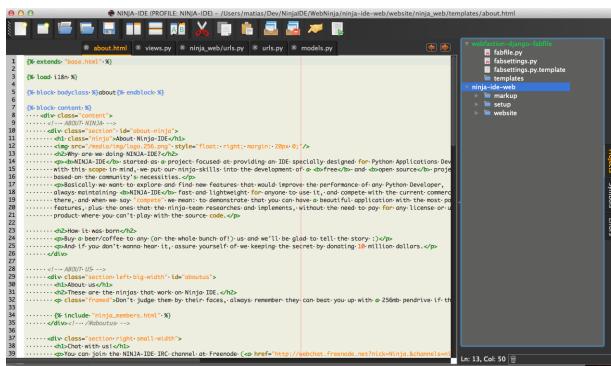


## Ninja-IDE

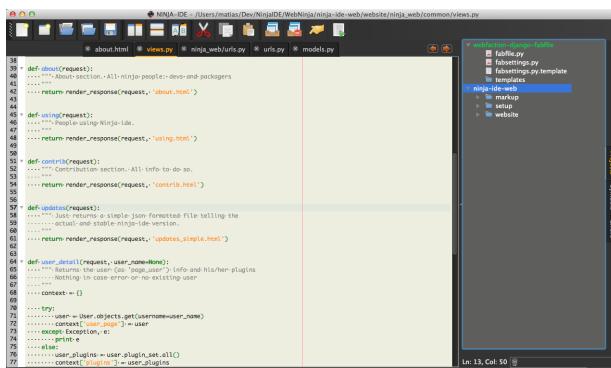


- Project Management
  - Amazing Code Locator
  - Errors and PEP8 Finder
  - Run Project and Files
  - Highly extensible with Plugins
-

## Ninja-IDE: screenshots



## Ninja-IDE: screenshots



## Quepy



- (El logo de Quepy está en camino!)
- Proyecto open-source desarrollado por Machinalis con el grupo de investigación de PNL de FaMAF
- Quepy es un framework para transformar preguntas en lenguaje natural a consultas en un lenguaje de base de datos
- Idea: con poco código podrías construir tu propio acceso a tu base de datos mediante lenguaje natural.
- Quepy provee soporte para el lenguaje de consultas sparql, pero puede ser extendido a otros lenguajes de consulta.

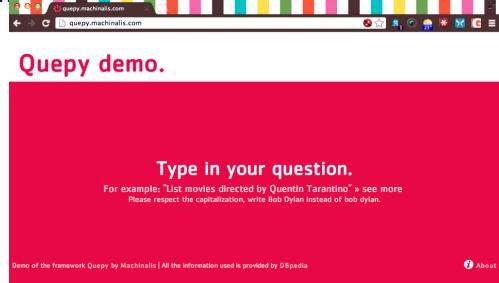
# Quepy: Links!



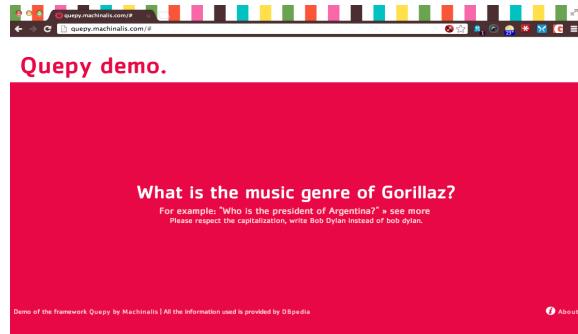
- <https://github.com/machinalis/quepy>
  - <http://quepy.readthedocs.org/>
  - <http://pypi.python.org/pypi/quepy/>
- 

## Quepy

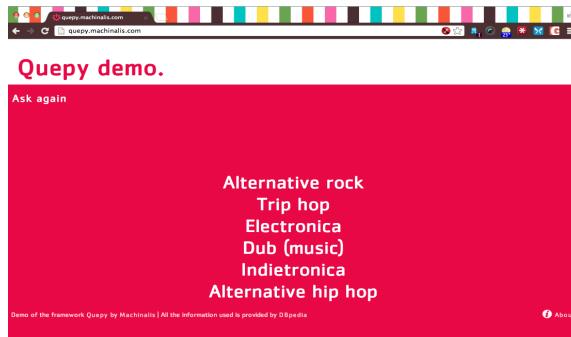
- Tenemos en funcionamiento una instancia de Quepy que consulta a DBpedia en: <http://quepy.machinalis.com/>



## Quepy: pregunta



# Quepy: respuesta!



## Productos Famosos



## Las empresas

Este es el listado de empresas que apuestan FUERTE por Python.

Más casos: <http://www.python.org/about/success/>



## Comunidad

En los siguientes slides se verá:

- La comunidad internacional de Python
- Grandes eventos internacionales
- Comunidad Local
- Eventos locales

## Comunidad Internacional

- Dada la expresividad del lenguaje, muchos hackers y científicos son miembros activos de esta comunidad

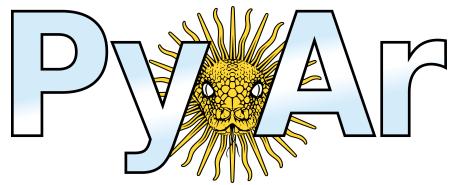
- Se nuclea a través de la Python Software Foundation (hay dos miembros de PyAr)
  - Se organizan cerca de 200 conferencias anuales de Python en todo el mundo.
- 



- Mayor conferencia de python a nivel mundial.
  - 5 mil asistentes.
  - 15 días (03/2013 es la siguiente en Santa Clara)
- 

## Muchas conferencias

- "PyCon" in United States
- "EuroPython" in Europe
- "EuroSciPy" in Europe
- "Kiwi PyCon" in New Zealand
- "PyCon Asia Pacific" in Singapore
- "PyCon AR" in Argentina
- "PyCon AU" in Australia
- "PythonBrasil" in Brazil
- "PyCon Canada" in Canada
- "PyCon China" in China
- "PyCon DE" in Germany
- "PyCon ES" in Spain
- "PyCon Finland" in Finland
- "PyCon FR" in France
- "PyCon India"
- "PyCon Ireland"
- "PyCon Italia" in Italy
- "PyCon Japan"
- "PyCon Philippines" in the Philippines
- "PyCon PL" in Poland
- "PyCon Russia" in Russia
- "PyCon Taiwan" in Taiwan
- "PyCon UK" in the United Kingdom
- "PyCon Ukraine" in Ukraine
- "PyCon Uruguay" in Uruguay
- "PyCon Venezuela" in Venezuela
- "PyCon ZA" in South Africa
- "SciPy" in the United States
- "SciPy.in" in India



- Una lista de correo con cerca de 400 personas activas.
  - Alto nivel técnico.
  - Recientemente fundado Sci-Pyar
  - Homepage: <http://python.org.ar>
- 

## Eventos 2011

- PyDay Córdoba 2011 (211 asistentes)
  - PyDay Gonzales Catán (~70 asistentes)
  - PyDay San Luis (~30 asistentes)
  - Django Day Córdoba 2011 (~30 asistentes)
  - PyConAr Junin 2011 (~270 asistentes)
- 

## Eventos 2012

- PyDay Junín 2012 (~50 asistentes)
  - PyDay Córdoba 2012 (~100 asistentes)
  - PyDay Rafaela 2012 (~30 asistentes (se llovio todo))
  - Pycon 2012 (403 asistentes)
  - Django Day La Plata
- 



- 403 asistentes.
  - 9 sprints.
  - 2 eventos en paralelo.
  - 10 tutoriales y workshops.
  - 51 charlas programadas.
  - 9 invitados internacionales.
-



*Foto final PyConAr 2012*

## Medios de Contacto y Fuentes

Estos slides:

- <https://github.com/matiasherranz/talks> (Git)
- <https://bitbucket.org/lelie12/talk> (Mercurial)

Contacto:

- matiasheranz@gmail.com - @matiasherranz
  - jbc.develop@gmail.com - @juanbcabral
- 

## ¿Preguntas?

---

**¡Muchas gracias!**