

¿Por qué Python?



**KEEP
CALM
AND
LOVE
PYTHON**

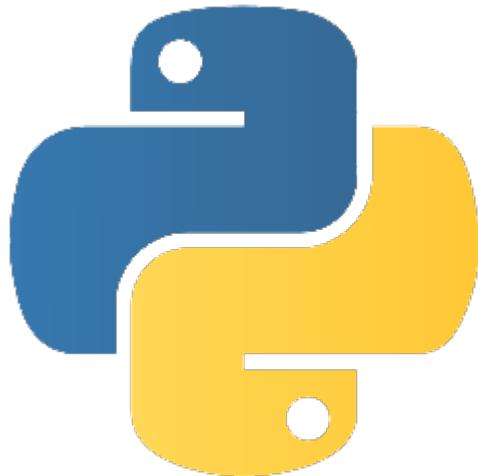
Sobre yo

Juan B Cabral

- Ing. en Sistemas.
- UTN-FRC
- Organizador de la última PyCon Argentina, de un PyDay y un Django Day.
- Se ocurrió esto del SciPyAr y SciPyConAr (Celia y Nahuel están felices con esto)
- Trabaja en Liricus.
- Trabajo con Python en mi día a día.
- hice su tesis de grado en Python.
- Hago ciencia con Python.
- Tomo whiskey.

¿De qué va la charla?

- En las PyCons y PyDays siempre se hablo de como hacer ciencia en python. Esto es lo mismo pero al revés.
- Ejemplos de algunas librerías que basan su funcionamiento en python.
- Programas hechos en Python.
- Empresas que apuestan fuerte por (iy ganan fuerte con!) Python
- Comunidad de Python
- Frameworks hechos en Python



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.
- 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).

¡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(UI_DIR("Dialog.ui")):

    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 engine

- 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

file:///home/juan/proyectos/talks/scipyconar2013/whypython/img/bsoup_logo.png

- Es un parser HTML/XML laxo (ifunciona 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> <!-- FE0000000000 -->
  <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

The Django logo consists of the word "Django" in a bold, dark green sans-serif font. The letter "D" is smaller than the other letters. The "j" has a long, thin vertical stroke extending downwards.

- 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 %}
```

Más para ver

WEB2PY

SKULPT



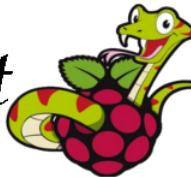
SUGAR LABS



BRYTHON

SPHINX
PYTHON DOCUMENTATION GENERATOR

*re*StructuredText



pypy

Android Python
with SL4A

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

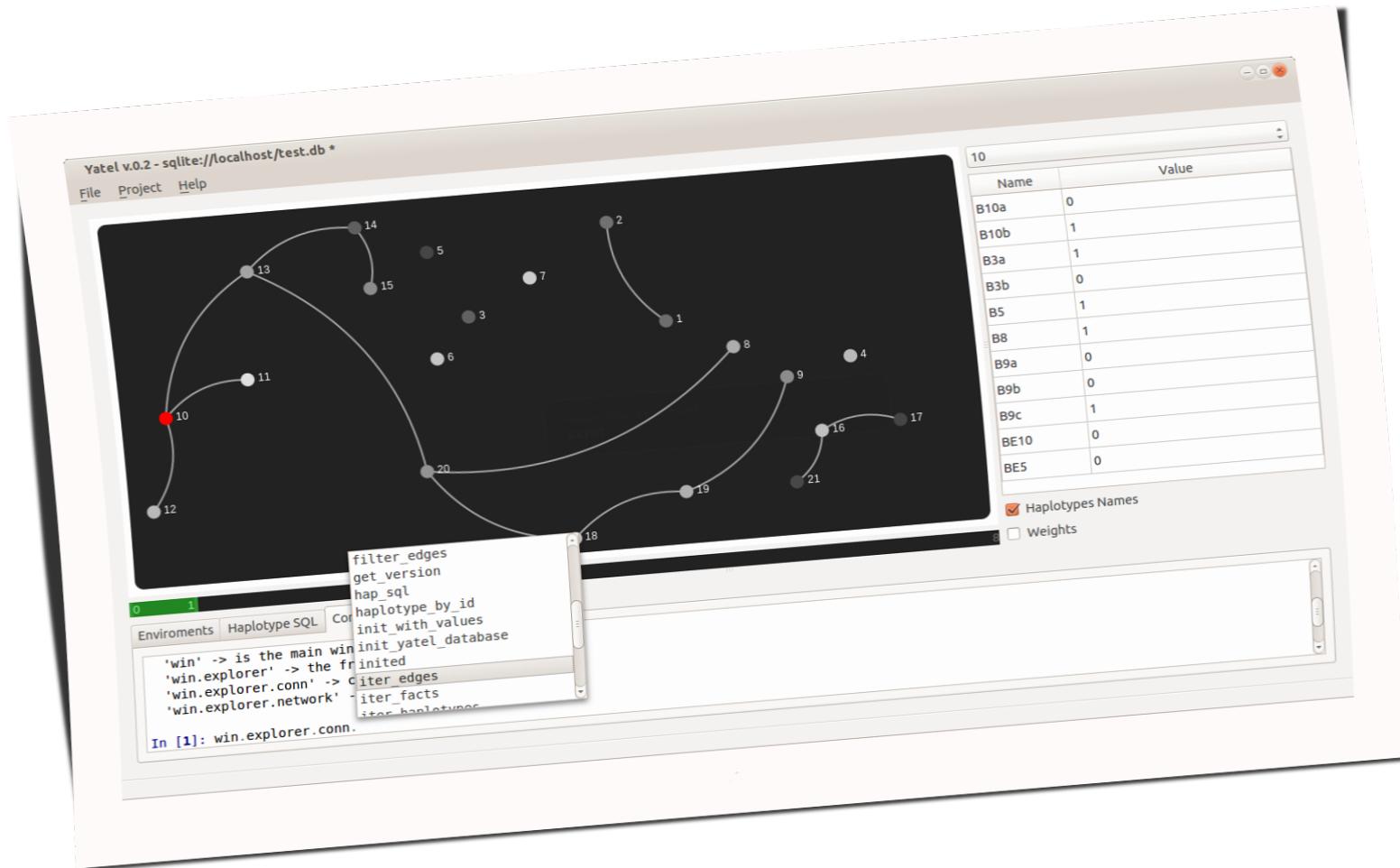
Yatel



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.
- Python + Javascript.
- Utiliza: PyQt, Pygments, Peewee, IPython.

Yatel

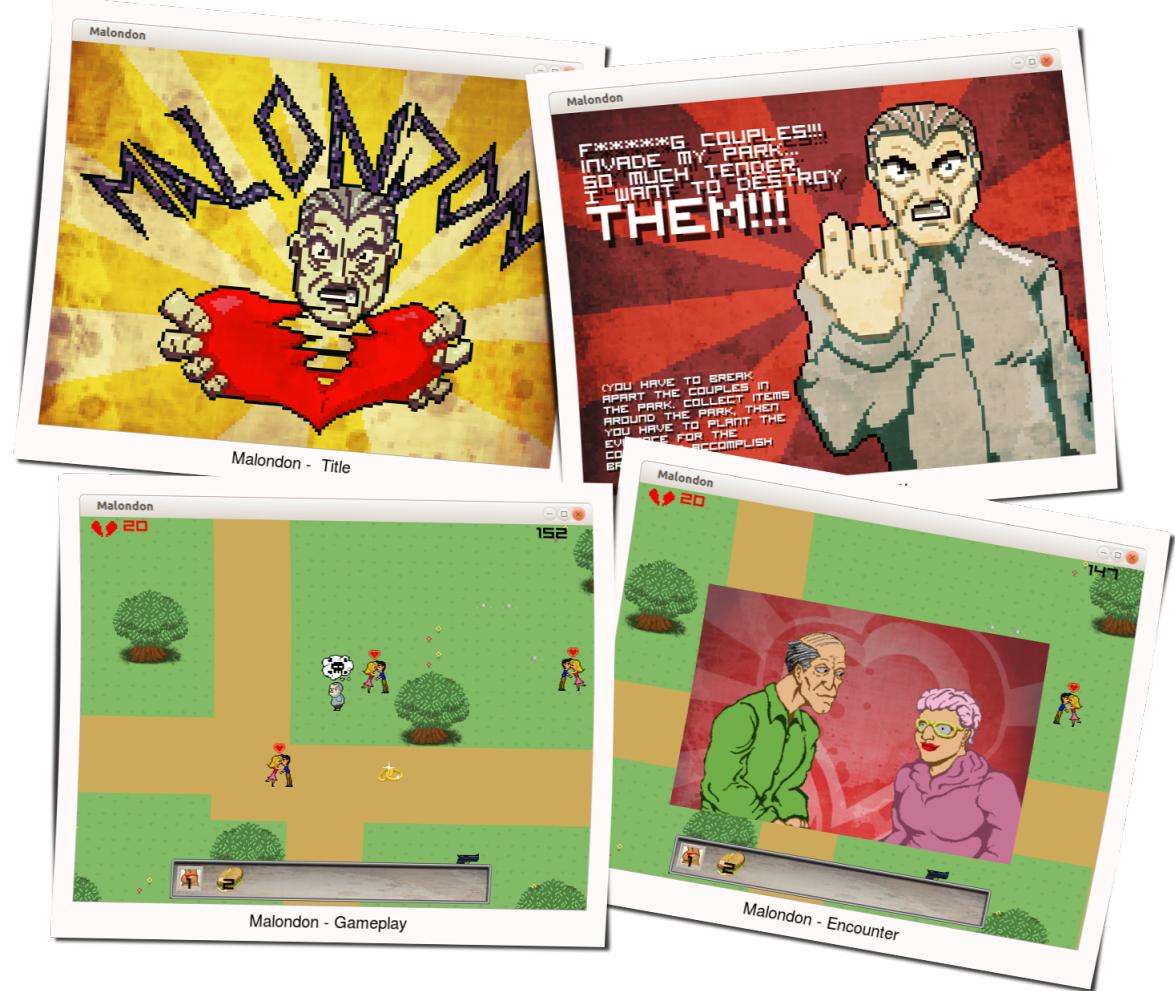


Malondon



- Córdoba Game Jam 2013
- Puro Pilas
- en 48 hs teníamos un prototipo funcional.
- Blues en ruso acompaña tu partida.
- <https://github.com/hugoruscitti/gamejam2013>

Malondon



Ninja-IDE



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

Ninja-IDE: screenshots

The screenshot shows the Ninja-IDE interface with the following details:

- Title Bar:** NINJA-IDE (PROFILE: NINJA-IDE) - /Users/matias/Dev/NinjaDE/WebNinja/ninja-ide-web/website/ninja_web/templates/about.html
- Toolbar:** Includes icons for file operations like Open, Save, Find, Copy, Paste, and Undo/Redo.
- Tab Bar:** Displays tabs for about.html, views.py, ninja_web/urls.py, urls.py, and models.py.
- Code Editor:** The main window contains the content of the about.html template file. The code includes HTML, CSS, and Python template tags (e.g., {% extends "base.html" %}, {% load i18n %}). A vertical red margin line is visible on the right side of the editor.
- Project Explorer:** Located on the right, it shows the project structure:
 - webfaction-django-fabfile
 - fabfile.py
 - fabsettings.py
 - fabsettings.py.template
 - templates
 - ninja-ide-web
 - markup
 - setup
 - website
- Status Bar:** Shows "Ln: 13, Col: 50" at the bottom right.

Ninja-IDE: screenshots

The screenshot shows the Ninja-IDE interface with the following details:

- Title Bar:** NINJA-IDE - /Users/matias/Dev/NinjalDE/WebNinja/ninja-ide-web/website/ninja_web/common/views.py
- Toolbar:** Includes icons for file operations like Open, Save, Find, Copy, Paste, and Undo/Redo.
- Code Editor:** Displays the contents of `views.py`. The code defines several views for a website, including `about`, `using`, `contrib`, `updates`, and `user_detail`. The `user_detail` view handles user authentication and retrieves user plugins.

```
38 def about(request):
39     """
40     All ninja people: devs and packagers
41     """
42     return render_response(request, 'about.html')
43
44
45 def using(request):
46     """
47     People using Ninja-ide.
48     """
49     return render_response(request, 'using.html')
50
51 def contrib(request):
52     """
53     Contribution section. All info to do so.
54     """
55     return render_response(request, 'contrib.html')
56
57 def updates(request):
58     """
59     Just returns a simple json-formatted file telling the
60     actual and stable ninja-ide version.
61     """
62     return render_response(request, 'updates_simple.html')
63
64 def user_detail(request, user_name=None):
65     """
66     Returns the user (as 'page_user') info and his/her plugins
67     Nothing in case error or no existing user
68     """
69     context = {}
70
71     try:
72         user = User.objects.get(username=user_name)
73         context['user_page'] = user
74     except Exception, e:
75         print e
76     else:
77         user_plugins = user.plugin_set.all()
78         context['plugins'] = user_plugins
```

- Project Tree:** Shows the project structure under `ninja-ide-web`. It includes `fabfile.py`, `fabsettings.py`, `fabsettings.py.template`, and a `templates` folder. Within `ninja-ide-web`, there are `markup`, `setup`, and `website` subfolders.
- Sidebar:** Features tabs for **Projects**, **Symbols**, and **Errors**.
- Status Bar:** Shows "Ln: 13, Col: 50" and a trash icon.

Quepy

file:///home/juan/proyectos/talks/scipyconar2013/whypython/img/logo-machinalis.png

- (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!

file:///home/juan/proyectos/talks/scipyconar2013/whypython/img/logo-machinalis.png

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

Quepy

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



Quepy demo.

Type in your question.

For example: "List movies directed by Quentin Tarantino" » see more
Please respect the capitalization, write Bob Dylan instead of bob dylan.

Quepy: pregunta



Quepy demo.

What is the music genre of Gorillaz?

For example: "Who is the president of Argentina?" » [see more](#)

Please respect the capitalization, write Bob Dylan instead of bob dylan.

Quepy: respuesta!



Ask again

Alternative rock

Trip hop

Electronica

Dub (music)

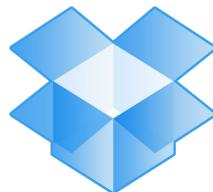
Indietronica

Alternative hip hop

Demo of the framework Quepy by Machinalis | All the information used is provided by DBpedia

[About](#)

Productos Famosos

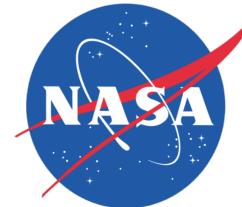
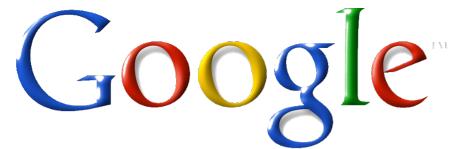


Dropbox

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

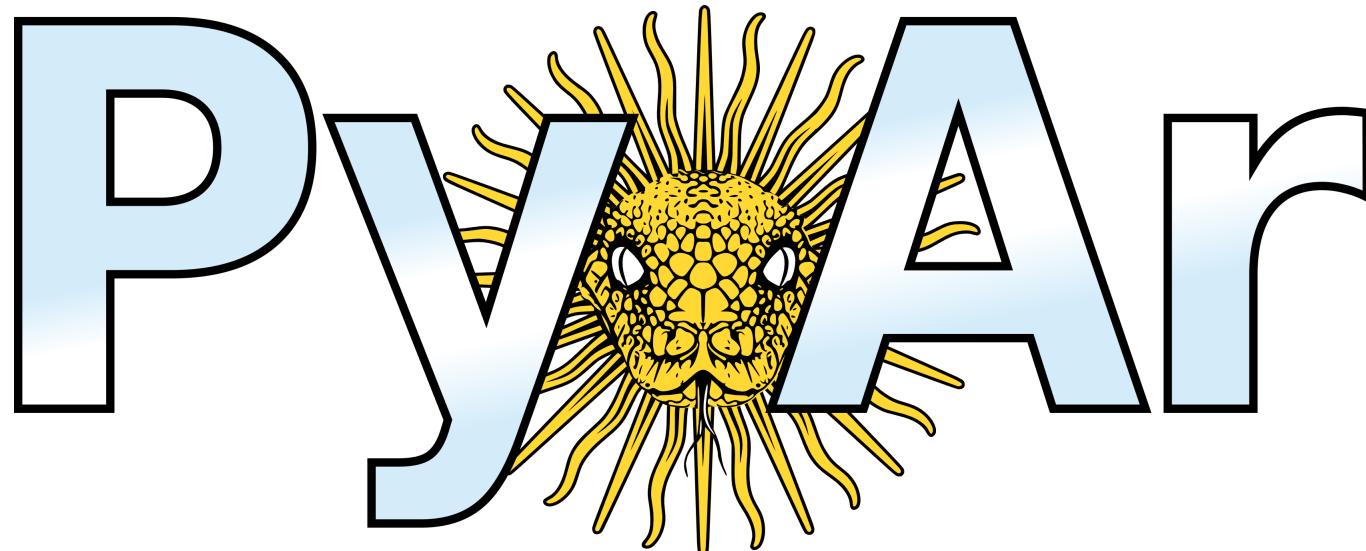
- Dada la expresividad del lenguaje, muchos hackers y científicos son miembros activos de esta comunidad
- Se nuclea a través de la Python Software Foundation (hay dos miembros de PyAr)
- Se organizan cerca de 200 conferencias anuales de Python en todo el mundo.



- 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

PyConAr 2013

- Rosario.
- Fines de Setiembre.
- <http://ar.pycon.org/>

Inscripción ▾ Acerca de ▾ Agenda ▾ Propuestas ▾ Auspiciantes ▾ Lugar ▾ Estadísticas ▾



PyCon Argentina 2013

5ta Conferencia Nacional de Python



Organiza



Rosario, Sede de PyCon Argentina 2013

Urbe cosmopolita con diversos atractivos culturales y turísticos, recibirá este año a cientos de desarrolladores y usuarios del lenguaje de programación Python, en la 5ta edición de la conferencia nacional que reúne expertos, docentes, científicos, alumnos y entusiastas de todo el país.

Medios de Contacto y Fuentes

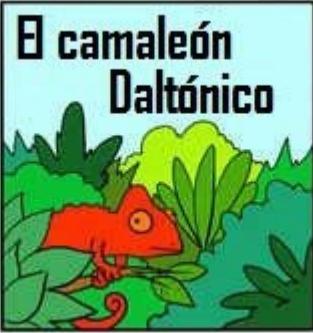
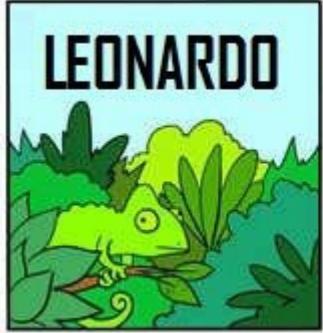
Estos slides:

- <https://bitbucket.org/leliel12/talks> (Mercurial)

Contacto:

- jbc.develop@gmail.com - @juanbcabral

¿Preguntas?



¡Muchas gracias!