- Данилов Константин
- kdanilov@mirantis.com - тему начинайте с "XXX"

- Слайды и доп материалы - github/koder-ua

Почему Python (-)

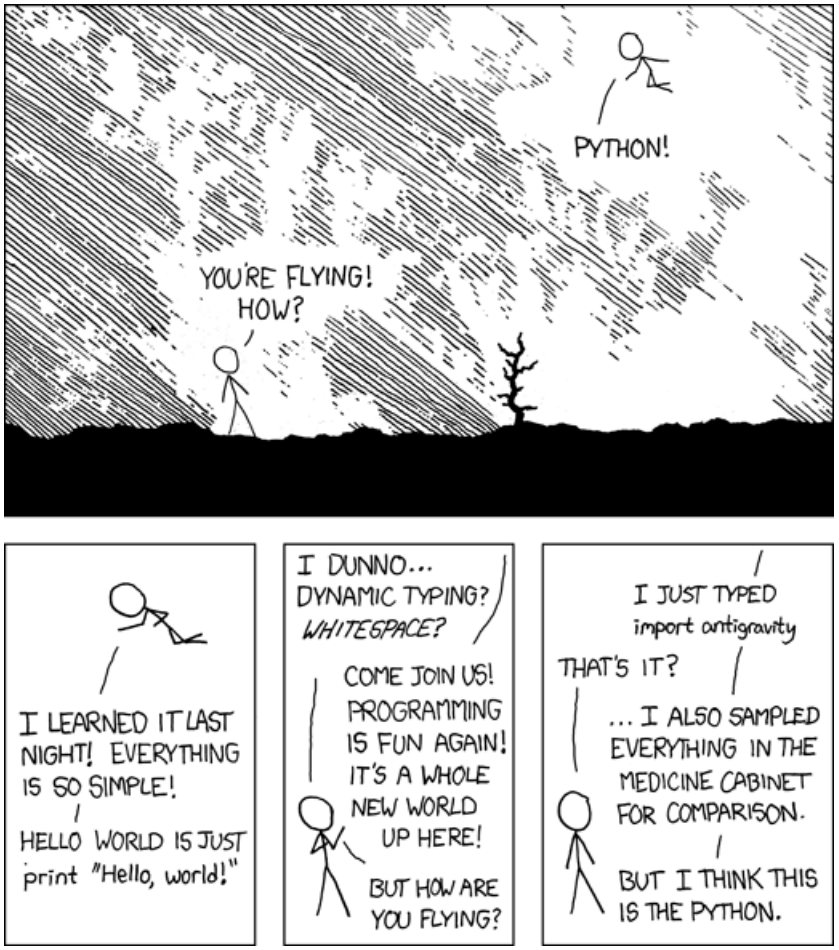
- Python достаточно сложен для изучения, особенно если разбираться во всех тонкостях
- Невозможность качественного статического анализа
- Слабая поддержка инструментальными средствами (по сравнению с C++/C#/Java)
- Несколько десятков способов официальных способов выстрелить себе в ногу (и попасть в голову соседу)
- Низкая скорость исполнения (по сравнению с компилируемыми языками)
- Высокая сложность написания JIT (по сравнению с Javascript)
- Поддержку многих высокоуровневых возможностей сложно сделать эффективной == она скорее всего всегда будет медленной (перегрузка функций, поиск по образцу, оптимизацию хвостовой рекурсии)
- В дизайне языка допускаются ошибки

Почему Python (+)

- Второй по лаконичности язык (уступает Perl) при этом имея строгий минималистичный синтаксис. Требует значительно меньше кода, чем компилируемые языки.
- Ядро языка чрезвычайно компактно. Нет декларируемых конструкций. Большая часть конструкция - синтаксический сахар
- В несколько десятков строк можно реализовать большинство отсутствующих языковых возможностей (но скорость...)
- Все это + интроспекция = pythonic библиотеки (документация по библиотеке умещается на заглавной странице сайта)
- Батарейки в комплекте
- Переносимость, обратная совместимость, прогнозируемая и формализованная модель разработки, ...

import antigravity

- import antigravity (python 2.7+)



Python

- Язык программирования ориентированный на скорость работы программиста и быстрое освоение библиотек/легкое использование
- Мультипарадигменный - процедурный, ООП, функциональный. Позволяет легко реализовывать другие парадигмы
- Opensource - hg clone http://hg.python.org/cpython
- С-подобный минималистичный (учитывая уровень языка) строгий синтаксис
- Разрабатывается с конца 80х. В 2001 выходит v2.1 и создается PSF.
- Развивается открытым сообществом под руководством Гвидо Ван Россума
- Роль стандарта выполняет CPython

Monty Python's flying circus

Распространение

- 4-8й язык по популярности (Javascript/VisualBasic/ObjectiveC/PHP)
- Научные расчеты и постобработка данных
- Web
- GUI, Системы управления, встраиваемый язык
- Склеивание компонентов, написанных на C/C++
- Xen, apt, mercurial, Trac, youtube, GAE, ...

Версии и реализации

- Две ветви 2.X(2.7.3) и 3.X(3.3.0rc1) 3.3.0 запланирована на 22 сентября
- 2.8 не будет
- Внутри каждой ветви поддерживается полная обратная совместимость (для py файлов)
- 3.X (Python 3k) достаточно близка к 2.X, содержит несовместимые исправления архитектурных ошибок, внесенных в язык на ранних стадиях
- print стал функцией, переработка юникод подсистемы, ввода-вывода и др
- Тем не менее любая нетривиальная программа на 2.X должна быть изменена для запуска на 3.X
- Все реализации в значительноы мере - интерпретаторы
- jython, PyPy, IronPython, Stackless Python, ...
- Работает на всех распространенных платфрмах - Intel win/lin/mac/bsd/.., Sun, Power, ARM(Android), Symbian, ...

Процесс разработки языка

- Формализованный и бюрократический подход к изменениям в языке
- Новые версии каждые 1.5 года
- Разработка ведется через python-dev & python-ideas списки рассылки
- Все рассылки открытые
- Все изменения и предложения описаны в PEP's (Python Enhancement Proposals) в т.ч. и отклоненные
- python-ideas -> python-dev -> PEP XXXX -> ... -> accepted/rejected by Guido

Библиотеки

- Очень широкий спектр библиотек
- Web, Сети, DB, Визуализация, Научные расчеты, XML, GUI,...
- Есть привязки почти для всех крупных C/C++ библиотек
- Cython, SWIG, SIP,...

Установка windows

- Windows: 2.7.3 с http://www.python.org/download/ или ActivePython с http://www.activestate.
- Почти все библиотеки - http://pypi.python.org/pypi
- Ручная установка библиотеки - exe или распаковать zip, и python setup.py install
- Пакетные менеджеры - pip, setuptools
  pip или easy_install install имя_пакета==версия
- virtualenv - создание изолированных окружений
- C:\Python2.7\lib\site-packages

Установка linux

- Linux: apt-get install python python-setuptools python-pip python-virtualenv
- Почти все библиотеки - http://pypi.python.org/pypi
- Ручная установка библиотеки - exe или распаковать zip, и python setup.py install
- Пакетные менеджеры - pip, setuptools
  pip или easy_install install имя_пакета==версия
- virtualenv - создание изолированных окружений
- /usr/lib/python2.7/dist-packages

IDE & Co

- Eclipse + pydev, PyCharm, PyScripter, Python for VS, ...
- Sublime Text 2, Notepad++, Texmate, Vim, Emacs,....
- ipython (pyreadline, http://ipython.org/pyreadline.html)
- ipython notebook (pyzmq + tornado)
- ipython qtconsole (PyQt4)
- PEP8, pylint >= pychecker >= pyflakes
- winpdb (требует wxpython)

Интерпретатор

- python
- ipython
- ipython notebook

Программа на Python

- Набор файлов на Python
- Каждый файл рассматривается как набор строк

Заголовок программы

```
1 #!/usr/bin/env python
2 # -*- coding:utf8 -*-
```

- Часть строки после # - комментарий
- Длинные строки можно переносить, поставив в конце "\". После него не должно идти пробелов

Пример программы

```
1       #!/usr/bin/env python
2       # -*- coding:utf8 -*-
3       x = 1
4       print "Hello, world!"
5       print "x =", x
6       print "This is definitelly " + \
7               "too long line"
```

Исполнение программы

- При первой загрузке программа компилируется в байтокод для виртуального стекового процессора, встроенного в CPython
- .py -> .pyc (python compiled)
- python -o .py -> .pyo. Удаление assert, etc
- .pyd, .so - бинарные модули

Print

- Вывод набора значений или переменных на экран
- Автоматически вставляет пробелы между значениями и перенос строки в конце

```
1       print var1, var2
2       print 1, 2, "34"
3       print x, y, x + y
4       print x, y, x + y, # no new line
```

Внешние модули

- Библиотеки на Python называются модулями или пакетами
- import module подключает модуль "module" в программу. После этого его элементы доступны как "module.name"
- from module import * напрямую включает все элементы "module" в программу.
- from module import xxx,yyy напрямую включает выбранные элементы "module" в программу.

```
1       import os
2       from os import listdir
3       from os import *
4
5       print os.listdir(".")
```

Ошибки

- При возникновении ошибки python порождает исключение, передающееся вверх по стеку до первого обработчика.
- Если в программе не определен ни один обработчик ошибок этого типа, то исключение передается в обработчик по умолчанию, печатающий информацию о исключении и завершающий программу.

```
1    def f1(a, b):
2        return a / b
3
4    def f2(m):
5        return f1(2, m)
6
7    f2(0)
```

```
1    Traceback (most recent call last):
2      File "/tmp/m.py", line 7, in <module>
3        f2(0)
4      File "/tmp/m.py", line 5, in f2
5        return f1(2, m)
6      File "/tmp/m.py", line 2, in f1
7        return a / b
8    ZeroDivisionError: integer division or modulo by zero
```

Справка и исследование объектов

- help(obj)
- obj? -- help
- obj?? -- help + source
- obj.<tab> -- extension
- dir(obj)

```
In [1]: import antigravity

In [2]: antigravity??
Type:           module
String Form:<module 'antigravity' from 'C:\Dev\Python\Python2
File:           c:\dev\python\python27_x86\lib\antigravity.py
Source:

import webbrowser

webbrowser.open("http://xkcd.com/353/")
```