

Лекция 1: Введение в язык Python

Сергей Мыц

кафедра Информатики, БГУИР

предмет “ИСП”, ИиТП, второй курс, весенний семестр
2017

Содержание лекции

- 1 Немного истории
- 2 Что это
- 3 Зачем оно
- 4 Характерные черты
 - Плюсы и минусы
 - Взаимодействие с языками и платформами
 - Расширяемость и библиотеки
- 5 Начало работы с языком
 - Идеология языка и соглашения
 - Хорошо выполняемые задачи
 - Интерпретаторы
 - Вспомогательные инструменты
 - Документация и книги

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Язык программирования Python



Источник названия

Создателю нравилась британская комедийная передача
Monty Python's flying circus.

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Fun to write

Основная идея: на языке должно быть удобно и приятно разрабатывать программы.

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

В каком году появился?

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

В каком году появился?

1991 год

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Создатель

Guido van Rossum

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

BDFL

Benevolent dictator for life

Великодушный пожизненный диктатор

О развитии языка

- Python имеет длинную историю развития
- Более подробно можно почитать самостоятельно в интернетех
- Отдельные интересные статьи
<http://python-history.blogspot.com>
- Сейчас активно используются версии языка 2.7.x и 3.x

Что такое Python?

Высокоуровневый динамический строго типизированный
интерпретируемый язык программирования общего
назначения.

Высокоуровневый

Высокий уровень абстракции от деталей исполняющей системы.

Динамический

Типы выводятся и проверяются во время выполнения
(run-time).

С версии 3.5 можно также использовать compile-time
проверки.

Строго типизированный

Сильно ограничены неявные приведения типов
(строка + число и т.п.).

Интерпретируемый

Команды языка интерпретируются во время работы специальной программой-интерпретатором.

Общего назначения

Нет ограниченной области применения
(в противоположность domain-specific языкам).

О чём могли слышать

- Отступы – определяют структуру программы
- Duck-typing – утиная типизация, характеризует обращение с объектами и вызов методов

Различные языки

Есть множество различных языков программирования.
Почему?

Различные классы задач

Отдельные языки помогают проще решать отдельные классы задач.

Примеры языков

- C\C++
- C#, Java
- Python, Go

У питона также есть своя ниша.

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Плюсы и минусы
Взаимодействие с языками и платформами
Расширяемость и библиотеки
Идеология языка и соглашения
Хорошо выполняемые задачи

Содержание

1 Немного истории

2 Что это

3 Зачем оно

4 Характерные черты

- Плюсы и минусы
- Взаимодействие с языками и платформами
- Расширяемость и библиотеки

- Идеология языка и соглашения
- Хорошо выполняемые задачи

5 Начало работы с языком

- Интерпретаторы
- Вспомогательные инструменты
- Документация и книги

- Кроссплатформенный
- Мультипарадигменный (процедурное, ООП, функциональное, метапрограммирование и др.)
- Автоматическое управление памятью (refcounting + cycle-detecting garbage collector)

Плюсы

- Легко разрабатывать – краткость и выразительность
- Легко читать – читабельность как одна из целей в основе дизайна языка
- Легко отлаживать – интерактивная проверка, можно всё динамически посмотреть

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Плюсы и минусы
Взаимодействие с языками и платформами
Расширяемость и библиотеки
Идеология языка и соглашения
Хорошо выполняемые задачи

Минусы

Относительно медленно работает – частично компенсируется интеграцией с другими языками или использованием альтернативных интерпретаторов.

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Плюсы и минусы
Взаимодействие с языками и платформами
Расширяемость и библиотеки
Идеология языка и соглашения
Хорошо выполняемые задачи

Минусы

Нет статических проверок компилятора – компенсируется качественным автоматизированным тестированием.

Содержание

- 1 Немного истории
- 2 Что это
- 3 Зачем оно
- 4 Характерные черты**
 - Плюсы и минусы
 - **Взаимодействие с языками и платформами**
 - Расширяемость и библиотеки
- 5 Начало работы с языком
 - Интерпретаторы
 - Вспомогательные инструменты
 - Документация и книги

Интеграция с другими языками

- Многие модули написаны на Си и C++
- Проработанные механизмы для создания обёрток к модулям на других языках
- Механизмы использования Python из других языков

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Плюсы и минусы
Взаимодействие с языками и платформами
Расширяемость и библиотеки
Идеология языка и соглашения
Хорошо выполняемые задачи

Python и платформы

- JVM – Jython
- .NET – Iron Python

Содержание

1 Немного истории

2 Что это

3 Зачем оно

4 Характерные черты

- Плюсы и минусы
- Взаимодействие с языками и платформами
- **Расширяемость и библиотеки**

- Идеология языка и соглашения
- Хорошо выполняемые задачи

5 Начало работы с языком

- Интерпретаторы
- Вспомогательные инструменты
- Документация и книги

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Плюсы и минусы
Взаимодействие с языками и платформами
Расширяемость и библиотеки
Идеология языка и соглашения
Хорошо выполняемые задачи

Расширяемость языка

Язык был изначально задуман легко расширяемым:
небольшое ядро + библиотеки.

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Плюсы и минусы
Взаимодействие с языками и платформами
Расширяемость и библиотеки
Идеология языка и соглашения
Хорошо выполняемые задачи

Библиотеки

- Стандартная библиотека
- Внешние модули

Богатая стандартная библиотека

- регулярные выражения
- юнит-тесты
- логгирование
- поддержка различных стандартных форматов и протоколов
- работа с базами данных
- многопоточность и многопроцессность
- и многое другое

Огромное количество внешних модулей

- web-фрэймворки
- web-сервера
- поддержка различных сетевых протоколов
- фрэймворки тестирования, автоматизации, документирования, системного администрирования
- web-crawler-ы
- научные вычисления, обработка текста и изображений

Содержание

1 Немного истории

2 Что это

3 Зачем оно

4 Характерные черты

- Плюсы и минусы
- Взаимодействие с языками и платформами
- Расширяемость и библиотеки

• Идеология языка и соглашения

- Хорошо выполняемые задачи

5 Начало работы с языком

- Интерпретаторы
- Вспомогательные инструменты
- Документация и книги

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Плюсы и минусы
Взаимодействие с языками и платформами
Расширяемость и библиотеки
Идеология языка и соглашения
Хорошо выполняемые задачи

Дзен

Набор основополагающих тезисов в основе развития и
использования языка

The Zen of Python, by Tim Peters

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Complex is better than complicated.
- Flat is better than nested.
- Sparse is better than dense.
- Readability counts.

The Zen of Python, by Tim Peters

- Special cases aren't special enough to break the rules.
- Although practicality beats purity.
- Errors should never pass silently.
- Unless explicitly silenced.

The Zen of Python, by Tim Peters

- In the face of ambiguity, refuse the temptation to guess.
- There should be one– and preferably only one –obvious way to do it.
- Although that way may not be obvious at first unless you're Dutch.
- Now is better than never.
- Although never is often better than **right** now.

The Zen of Python, by Tim Peters

- If the implementation is hard to explain, it's a bad idea.
- If the implementation is easy to explain, it may be a good idea.
- Namespaces are one honking great idea – let's do more of those!

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Плюсы и минусы
Взаимодействие с языками и платформами
Расширяемость и библиотеки
Идеология языка и соглашения
Хорошо выполняемые задачи

The Zen of Python, by Tim Peters

- `import this`
- `cat this.py`

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Плюсы и минусы
Взаимодействие с языками и платформами
Расширяемость и библиотеки
Идеология языка и соглашения
Хорошо выполняемые задачи

PEP – Python Enhancement Proposal

Документы стандартизированного формата в которых
содержатся предложения по развитию языка и описание уже
имеющихся элементов.

Оформление и документирование кода

- PEP-8 – <https://www.python.org/dev/peps/pep-0008/>
- PEP-257 – <https://www.python.org/dev/peps/pep-0257/>

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Плюсы и минусы
Взаимодействие с языками и платформами
Расширяемость и библиотеки
Идеология языка и соглашения
Хорошо выполняемые задачи

Pythonic

Понятие `pythonic` используется для характеристики решений, элементов и практик, которые соответствуют идеям языка Python.

Pythonic

Изучаем самостоятельно:

- <http://python.net/~goodger/projects/pycon/2007/idiomatic/handout.html>
- <http://habrahabr.ru/post/114731/>
- <http://docs.python-guide.org/en/latest/writing/style/>

Содержание

- 1 Немного истории
- 2 Что это
- 3 Зачем оно
- 4 Характерные черты**
 - Плюсы и минусы
 - Взаимодействие с языками и платформами
 - Расширяемость и библиотеки
- 5 Начало работы с языком
 - Интерпретаторы
 - Вспомогательные инструменты
 - Документация и книги

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Плюсы и минусы
Взаимодействие с языками и платформами
Расширяемость и библиотеки
Идеология языка и соглашения
Хорошо выполняемые задачи

Когда функциональность важнее эффективности

Python особенно популярен в задачах, где функциональность гораздо важнее эффективности: исследовательское программирование и написание прототипов.

Вспомогательный инструмент

Даже если нужно быстро, язык будет полезен просто как инструмент для написания небольших и средних серверных и вспомогательных прикладных программ для обработки данных.

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Плюсы и минусы
Взаимодействие с языками и платформами
Расширяемость и библиотеки
Идеология языка и соглашения
Хорошо выполняемые задачи

Веб и сервера

Позволяет просто реализовать весь набор элементов
веб-приложения любой сложности.

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Плюсы и минусы
Взаимодействие с языками и платформами
Расширяемость и библиотеки
Идеология языка и соглашения
Хорошо выполняемые задачи

Где используется

В значительном количестве крупных компаний
разрабатывающих ПО или обрабатывающих данные так или
иначе используется Python.

Немного истории
Что это
Зачем оно
Характерные черты
Начало работы с языком

Плюсы и минусы
Взаимодействие с языками и платформами
Расширяемость и библиотеки
Идеология языка и соглашения
Хорошо выполняемые задачи

Где используется

Пруфы

- [ru.wiki] <http://bit.ly/12ycKDm>
- [en.wiki] <http://bit.ly/1iAj5Gy>.

Содержание

- 1 Немного истории
- 2 Что это
- 3 Зачем оно
- 4 Характерные черты
 - Плюсы и минусы
 - Взаимодействие с языками и платформами
 - Расширяемость и библиотеки
- 5 Начало работы с языком
 - Идеология языка и соглашения
 - Хорошо выполняемые задачи
 - Интерпретаторы
 - Вспомогательные инструменты
 - Документация и книги

Интерпретаторы

- CPython – стандартная реализация
- Jython – JVM
- Iron Python – .NET
- PyPy – JIT-компиляция и написан на языке с python-подобным синтаксисом
- Stackless Python – вместо стека вызовов функций из C использует собственную реализацию
- и др.

Версии языка

- Основной выбор: 2.7.x или 3.x
- Почитать про различия можно тут <http://bit.ly/1gcF18X>
- Мы в этом курсе будем пользоваться 2.7
- <https://www.python.org/downloads/release/python-2713/>

Мотивация выбора 2.7

- Всё ещё широко используется
- Будет меньше проблем с совместимостью tutorиалов, документации, описаний найденных в интернете
- Не будет проблем с совместимостью большинства библиотек
- При желании, несложно самостоятельно перейти на использование версии 3.x

Содержание

- 1 Немного истории
- 2 Что это
- 3 Зачем оно
- 4 Характерные черты
 - Плюсы и минусы
 - Взаимодействие с языками и платформами
 - Расширяемость и библиотеки
- 5 Начало работы с языком
 - Идеология языка и соглашения
 - Хорошо выполняемые задачи
 - Интерпретаторы
 - Вспомогательные инструменты
 - Документация и книги

Интерактивная оболочка IPython

Расширенная по сравнению со стандартной оболочка для интерактивного выполнения программ:

- улучшенная интроспекция
- дополнительные команды
- подсветка кода
- автодополнение

Редакторы

- Расширяемые редакторы Emacs или Vim с множеством различных плагинов
- PyCharm – хорошая IDE для Python
- Sublime Text 3 – популярный расширяемый редактор для тех, кто не осилил Vim :)

PyPi

- Центральный репозиторий модулей для Python
- Позволяет очень просто распространять их
- Подавляющее большинство библиотек устанавливается одной короткой командой
- `pip install <pkg_name>`

Awesome Python

- <https://github.com/vinta/awesome-python>
- И вообще:
<https://github.com/bayandin/awesome-awesomeness>

Содержание

- 1 Немного истории
- 2 Что это
- 3 Зачем оно
- 4 Характерные черты
 - Плюсы и минусы
 - Взаимодействие с языками и платформами
 - Расширяемость и библиотеки
- 5 Начало работы с языком
 - Идеология языка и соглашения
 - Хорошо выполняемые задачи
 - Интерпретаторы
 - Вспомогательные инструменты
 - Документация и книги

Официальная документация

- <https://docs.python.org/>
- Можно выбрать интересующую версию языка (нам 2.7, пожалуйста)
- Практически все детали от базовых до глубоких и редко используемых можно найти здесь или начиная отсюда

Книги

- Марк Пилгрим. Погружение в Python – быстрая вводная в python в виде книги [ru]
- <http://www.diveintopython.net/toc/index.html> – обновляемая онлайн версия предыдущей книги [en]
- Марк Лутц. Программирование на Python – подробное руководство в виде книги [ru]

Тьюториалы

- <http://learnpythonthehardway.org/book/> – подробный пошаговый онлайн тьюториал для начинающих [en]
- <http://www.swaroopch.com/notes/python/> – онлайн введение в python для начинающих [en]

Продвинутый уровень

- <http://docs.python-guide.org/en/latest/> – продвинутое руководство по использованию языка
- <https://speakerdeck.com/pyconslides/transforming-code-into-beautiful-idiomatic-python-by-raymond-hettinger-1>
– продвинутый уровень: советы по написанию кода