Язык программирования Python

Выполнил - Гильмуллин Рафаил 25 группы

ГБПОУ ТПК

Что такое Python

- **Python -** высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Язык является полностью объектно-ориентированным в том плане, что всё является объектами. Необычной особенностью языка является выделение блоков кода пробельными отступами. Синтаксис ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. Сам же язык известен как интерпретируемый и используется в том числе для написания скриптов. Недостатками языка являются зачастую более низкая скорость работы и более высокое потребление памяти написанных на нём программ по сравнению с аналогичным кодом, написанным на компилируемых языках, таких как C или C++.

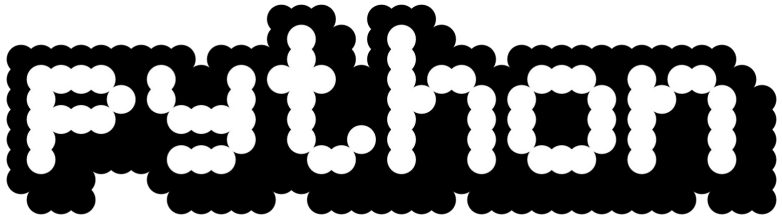
- **Python** - является мультипарадигмальным языком программирования, поддерживающим императивное, процедурное, структурное, объектно-ориентированное программирование, метапрограммирование и функциональное программирование. Задачи обобщённого программирования решаются за счёт динамической типизации. Аспектно-ориентированное программирование частично поддерживается через декораторы, более полноценная поддержка обеспечивается дополнительными фреймворками. Такие методики как контрактное и логическое программирование можно реализовать с помощью библиотек или расширений. Основные архитектурные черты — динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений с глобальной блокировкой интерпретатора (GIL), высокоуровневые структуры данных. Поддерживается разбиение программ на модули, которые, в свою очередь, могут объединяться в пакеты.

Реализация

- Эталонной реализацией Python является интерпретатор CPython, который поддерживает большинство активно используемых платформ и являющийся стандартом де-факто языка. Он распространяется под свободной лицензией Python Software Foundation License, позволяющей использовать его без ограничений в любых приложениях, включая проприетарные. CPython компилирует исходные тексты в высокоуровневый байт-код, который исполняется в стековой виртуальной машине. К другим трём основным реализациям языка относятся Jython (для JVM), IronPython (для CLR/.NET) и PyPy. PyPy написан на подмножестве языка Python (RPython) и разрабатывался как альтернатива CPython с целью повышения скорости исполнения программ, в том числе за счёт использования JIT-компиляции. Поддержка версии Python 2 закончилась в 2020 году. На текущий момент активно развивается версия языка Python 3. Разработка языка ведётся через предложения по расширению языка **PEP** (англ. *Python Enhancement Proposal*), в которых описываются нововведения, делаются корректировки согласно обратной связи от сообщества и документируются итоговые решения.

- **Стандартная библиотека** включает большой набор полезных переносимых функций, начиная с возможностей для работы с текстом и заканчивая средствами для написания сетевых приложений. Дополнительные возможности, такие как математическое моделирование, работа с оборудованием, написание веб-приложений или разработка игр, могут реализовываться посредством обширного количества сторонних библиотек, а также интеграцией библиотек, написанных на Си или C++, при этом и сам интерпретатор Python может интегрироваться в проекты, написанные на этих языках. Существует и специализированный репозиторий программного обеспечения, написанного на Python, — PyPI. Данный репозиторий предоставляет средства для простой установки пакетов в операционную систему и стал стандартом де-факто для Python. По состоянию на 2019 год в нём содержалось более 175 тысяч пакетов.

История



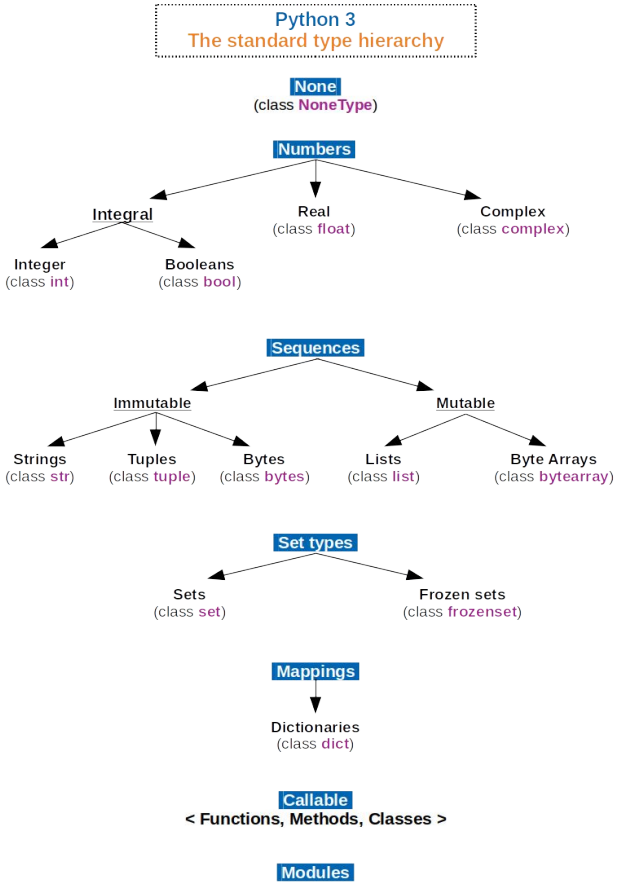
Логотип, использовавшийся с 1990-х до 2006 года

- Задумка по реализации языка появилась в конце 1980-х годов, а разработка его реализации началась в 1989 году сотрудником голландского института CWI Гвидо ван Россумом. Для распределённой операционной системы Amoeba требовался расширяемый скриптовый язык, и Гвидо начал разрабатывать Python на досуге, позаимствовав некоторые наработки для языка ABC (Гвидо участвовал в разработке этого языка, ориентированного на обучение программированию). В феврале 1991 года Гвидо опубликовал исходный текст в группе новостей alt.sources. С самого начала Python проектировался как объектно-ориентированный язык.

Портируемость

- Python портирован и работает почти на всех известных платформах — от КПК до мейнфреймов. Существуют порты под Microsoft Windows, практически под все варианты UNIX (включая FreeBSD и Linux), Android, Plan 9, Mac OS и macOS, iPhone OS (iOS) 2.0 и выше, iPadOS, Palm OS, OS/2, Amiga, HaikuOS, AS/400, OS/390, Windows Mobile и Symbian.
- По мере устаревания платформы её поддержка в основной ветви языка прекращается. Например, с версии 2.6 прекращена поддержка Windows 95, Windows 98 и Windows ME. В версии 3.5 перестала поддерживаться Windows XP. В версии 3.9 перестала поддерживаться Windows Vista и Windows 7.
- При этом, в отличие от многих портируемых систем, для всех основных платформ Python имеет поддержку характерных для данной платформы технологий (например, Microsoft COM/DCOM). Более того, существует специальная версия Python для виртуальной машины Java — Jython, что позволяет интерпретатору выполняться на любой системе, поддерживающей Java, при этом классы Java могут непосредственно использоваться из Python и даже быть написанными на Python. Также несколько проектов обеспечивают интеграцию с платформой Microsoft.NET, основные из которых — IronPython и Python.Net.

Типы и структуры данных



- Python поддерживает динамическую типизацию, то есть тип переменной определяется только во время исполнения. Поэтому вместо «присваивания значения переменной» лучше говорить о «связывании значения с некоторым именем». К примитивным типам в Python относятся булевый, целое число произвольной точности, число с плавающей запятой и комплексное число. Из контейнерных типов в Python встроены: строка, список, кортеж, словарь и мно жество. Все значения являются объектами, в том числе функции, методы, модули, классы.

Операторы

Набор операторов достаточно традиционен.

- Условный оператор **if** (если). При наличии нескольких условий и альтернатив применяется необязательный блок **elif** (сокр. от else if) который может повторяться в коде неограниченное число раз. Если ни одно из условий не было соблюдено, то выполняется необязательный блок **else** (иначе).
- Оператор цикла **while**.
- Оператор цикла **for**.
- Операторы обработки исключений **try — except — else — finally**.
- Оператор определения класса **class**.
- Оператор определения функции, метода или генератора **def**. Внутри возможно применение **return** (возврат) для возврата из функции или метода, а в случае генератора — **yield** (давать).
- Оператор pass ничего не делает. Используется для пустых блоков кода.

Имена

- Имя (идентификатор) может начинаться с буквы любого алфавита в Юникоде любого регистра или подчёркивания, после чего в имени можно использовать и цифры. В качестве имени нельзя использовать ключевые слова (их список можно узнать по import keyword; print(keyword.kwlist)) и нежелательно переопределять встроенные имена. Имена, начинающиеся с символа подчёркивания, имеют специальное значение.
- В каждой точке программы интерпретатор имеет доступ к трём пространствам имён (то есть отображениям имён в объекты): локальному, глобальному и встроенному.
- *Области видимости* имён могут быть вложенными друг в друга (внутри определяемой функции видны имена из окружающего блока кода). На практике с областями видимости и связыванием имён связано несколько правил «хорошего тона», о которых можно подробнее узнать из документации.

Строки документации

- Python предлагает механизм документирования кода pydoc. В начало каждого модуля, класса, функции вставляется строка документации — *docstring (англ.)*. Строки документации остаются в коде на момент времени исполнения, и в язык встроен доступ к документации(переменная __doc__), что используется современными IDE (Интегрированная среда разработки) (например, Eclipse).
- В интерактивном режиме можно получить помощь, сгенерировать гипертекстовую документацию по целому модулю или даже применить *doctest (англ.)* для автоматического тестирования модуля.

Парадигмы программирования

- Python — мультипарадигмальный язык программирования. Полностью поддерживаются объектно-ориентированное, структурное, обобщённое, функциональное программирование и метапрограммирование. Базовая поддержка аспектно-ориентированного программирования реализуется за счёт метапрограммирования. Множество других методик, в том числе контрактное и логическое программирование можно реализовать с помощью расширений.

Функциональное программирование

Несмотря на то, что Python изначально не задумывался как язык функционального программирования, Python поддерживает программирование в стиле функционального программирования, в частности:

- функция является объектом первого класса,
- функции высших порядков,
- рекурсия,
- фокус на работу со списками,
- аналог замыканий,
- частичное применение функции с помощью метода partial(),
- возможность реализации других средств на самом языке (например, карринг).

Однако, в отличие от большинства языков, непосредственно ориентированных на функциональное программирование, Python не является чистым языком программирования и код не защищён от побочных эффектов.

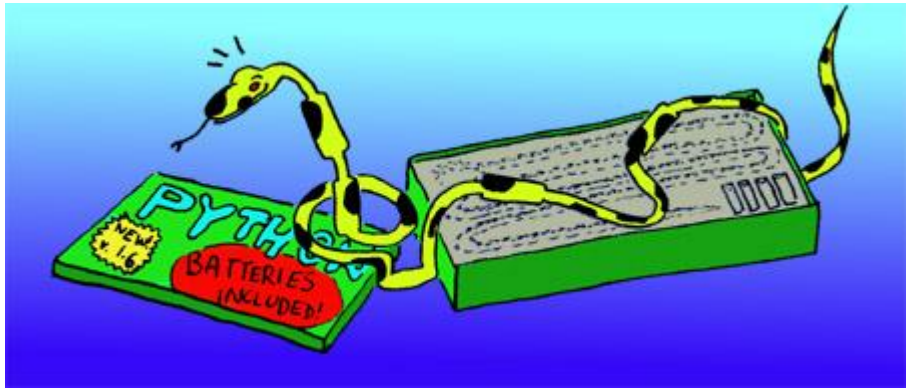
В стандартной библиотеке Python существуют специальные пакеты operator и functools для функционального программирования.

Управление контекстом выполнения

В Python 2.5 появились средства для управления контекстом выполнения блока кода — оператор with и модуль contextlib.

Оператор может применяться в тех случаях, когда до и после некоторых действий должны обязательно выполняться некоторые другие действия, независимо от возбуждённых в блоке исключений или операторов return: файлы должны быть закрыты, ресурсы освобождены, перенаправление стандартного ввода вывода закончено и т. п. Оператор улучшает читаемость кода, а значит, помогает предотвращать ошибки.

Стандартная библиотека



Python поставляется «с батарейками в комплекте». Такую метафору использовали разработчики, чтобы подчеркнуть богатую стандартную библиотеку языка

Богатая стандартная библиотека является одной из привлекательных сторон Python. Здесь имеются средства для работы со многими сетевыми протоколами и форматами Интернета, например, модули для написания HTTP-серверов и клиентов, для разбора и создания почтовых сообщений, для работы с XML и т. п. Набор модулей для работы с операционной системой позволяет писать кросс-платформенные приложения. Существуют модули для работы с регулярными выражениями, текстовыми кодировками, мультимедийными форма тами, криптографическими протоколами, архивами, сериализации данных, поддержка юнит-тестирования и др.

Графические библиотеки

С Python поставляется библиотека tkinter на основе Tcl/Tk для создания кроссплатформенных программ с графическим интерфейсом.

Существуют расширения, позволяющие использовать все основные библиотеки графических интерфейсов — wxPython, основанное на библиотеке wxWidgets, PyGObject для GTK, PyQt и PySide для Qt и другие. Некоторые из них также предоставляют широкие возможности по работе с базами данных, графикой и сетями, используя все возможности библиотеки, на которой основаны.

Для создания игр и приложений, требующих нестандартного интерфейса, можно использовать библиотеку Pygame. Она также предоставляет обширные средства работы с мультимедиа: с её помощью можно управлять звуком и изображениями, воспроизводить видео. Предоставляемое pygame аппаратное ускорение графики OpenGL имеет более высокоуровневый интерфейс по сравнению с PyOpenGL, копирующей семантику C-библиотеки для OpenGL. Есть также PyOgre, обеспечивающая привязку к Ogre — высокоуровневой объектно-ориентированной библиотеке 3D-графики. Кроме того, существует библиотека pythonOCC, обеспечивающая привязку к среде 3D-моделирования и симуляции OpenCascade.

Языки, на которые повлиял Python

Python, как весьма популярный язык программирования, повлиял на следующие языки:

- CoffeeScript имеет синтаксис, вдохновлённый Python.
- ECMAScript/JavaScript заимствовал итераторы и генераторы из Python.
- Go, при сильнейших идеологических различиях, заимствовал у динамических языков, таких как Python, встроенные словари, динамические массивы, срезы.
- Groovy был создан с мотивацией привнести философию Python на Java.
- Julia была задумана как «такая же пригодная для общего программирования, как и Python».
- Nim использует систему отступов и аналогичный синтаксис.
- Ruby — Юкихиро Мацумото, создатель языка, сказал: «Я хотел скриптовый язык, который был бы более мощным, чем Perl, и более объектно-ориентированным, чем Python. Вот почему я решил создать свой собственный язык».
- Swift во время разработки брал идеи структуры языка из Python, а также из Objective-C, Rust, Haskell, Ruby, C#, CLU.

Благодарю за внимание!
Теперь вы знаете что такое язык программирования Python

Поставьте 5 пожалуйста)

Я старался