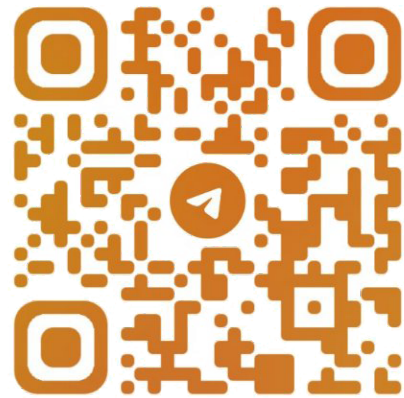
ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ С ПОМОЩЬЮ PYTHON

ИРВ КАЛЬБ

БОМБОРА
ИЗДАТЕЛЬСТВО

no starch press



IRV KALB

OBJECT-ORIENTED PYTHON

MASTER OOP BY BUILDING GAMES AND GUIS

ИРВ КАЛЬБ

ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ С ПОМОЩЬЮ PYTHON

БОМБОРА
ИЗДАТЕЛЬСТВО
Москва 2024

УДК 004.42
ББК 32.973.26-018.2
К17

Кальб, Ирв.

К17 Объектно-ориентированное программирование с помощью Python / Ирв Кальб ; [перевод с английского М. А. Райтмана]. — Москва : Эксмо, 2024. — 512 с. — (Мировой компьютерный бестселлер).

ISBN 978-5-04-186627-3

Объектно-ориентированное программирование (ООП) — это метод, основанный на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования, что позволяет по-другому думать о вычислительных задачах и решать их с возможностью многократного использования. «Объектно-ориентированное программирование с помощью Python» предназначено для программистов среднего уровня и представляет собой практическое руководство, которое глубоко изучает основные принципы ООП и показывает, как использовать инкапсуляцию, полиморфизм и наследование для написания игр и приложений с использованием Python.

Книга начинается с рассказа о ключевых проблемах, присущих процедурному программированию, затем вы познакомитесь с основами создания классов и объектов в Python.

Затем вы научитесь создавать графические интерфейсы с помощью pygame, благодаря чему вы сможете писать интерактивные игры и приложения с виджетами графического пользовательского интерфейса (GUI), анимацией, различными сценами и многоразовой игровой логикой.

УДК 004.42
ББК 32.973.26-018.2

ISBN 978-5-04-186627-3

Моей замечательной жене Дорин.
Ты клей, который держит нашу семью вместе.
Много лет назад я сказал: «Я делаю», но я имел в виду: «Я сделаю».

ОБ АВТОРЕ

Ирв Кальб — профессор в UCSC Silicon Valley Extension и Университете Кремниевой долины (ранее Политехнический колледж Когсвелла), где он преподает вводные курсы программирования и курсы объектно-ориентированного программирования на языке Python. Ирв имеет степени бакалавра и магистра в области компьютерных наук, более 30 лет занимается объектно-ориентированным программированием на различных языках и более 10 лет преподает. У него десятилетний опыт разработки программного обеспечения с акцентом на образовательное ПО. Как Furry Pants Productions он и его жена создали и выпустили два обучающих диска с персонажем — далматинцем Дарби в главной роли. Ирв также является автором Learn to Program with Python 3: A Step-by-Step Guide to Programming («Учимся программировать на Python 3. Пошаговое руководство по программированию»).

Ирв активно участвовал в раннем развитии спорта Ultimate Frisbee®. Он возглавил создание многих версий официального сборника правил и стал соавтором и издателем первой книги об этом виде спорта — Ultimate: Fundamentals of the Sport («Ultimate: Основы спорта»).

О ТЕХНИЧЕСКОМ АВТОРЕ

Монте Давидофф — независимый консультант по разработке программного обеспечения. Его области специализации включают DevOps и Linux. Монте программирует на Python уже более 20 лет. Он использовал Python для разработки разнообразного программного обеспечения, включая критически важные для бизнеса приложения и встроенное ПО.

КРАТКОЕ СОДЕРЖАНИЕ

ПОДРОБНОЕ СОДЕРЖАНИЕ

Часть III. Инкапсуляция, полиморфизм и наследование 233

БЛАГОДАРНОСТИ

Я хотел бы поблагодарить людей, которые помогли создать эту книгу:

- Эла Свейгарта — за то, что приучил меня к `pygame` (особенно за его код Pygbutton), и за то, что позволил мне использовать концепцию его игры Dodger;
- Монте Давидоффа, который помог мне получить исходный код и документацию к нему для правильной сборки с помощью GitHub, Sphinx и ReadTheDocs. Этот человек творил чудеса, используя бесчисленное множество инструментов, чтобы управляться с файлами;
- Монте Давидоффа (да, это все тот же парень) — за то, что оказался выдающимся техническим рецензентом. Монте дал отличные технические и авторские предложения по всей книге, и многие примеры кода стали более Pythonic и более ООП-ориентированными именно после его комментариев;
- Тепа Сатья Кхиеу, который проделал потрясающую работу, нарисовав все оригинальные схемы для этой книги. Я не художник (я даже не играю ни одного на ТВ). Теп смог взять мои примитивные карандашные наброски и превратить их в четкие, последовательные произведения искусства;

- Харрисона Янга, Кевина Лая и Эмили Эллис — за их вклад в художественное оформление некоторых игр;
- ранних рецензентов: Илью Кацюка, Джейми Калб, Гергану Анджелову и Джо Лангмюра, которые нашли и устранили много опечаток и внесли отличные предложения по исправлениям и уточнениям;
- всех редакторов, которые работали над этой книгой: Лиз Чедвик (редактор по развитию), Рейчел Хед (редактор) и Кейт Камински (производственный редактор). Все они внесли огромный вклад, задавая вопросы и часто переписывая и реорганизуя некоторые из моих объяснений. Они также были чрезвычайно полезны в расставлении и удалении запятых [нужна ли она здесь?] и удлинении моих предложений, как здесь, чтобы убедиться, что точка встречается там, где нужна (ОК, я остановлюсь!). Я боюсь, что никогда не пойму, где следует использовать «который», а не «что», или как расставлять запятые и тире, но я рад, что они знают! Спасибо также Морин Форис (верстальщице) за ее ценный вклад в готовый продукт;
- всех студентов, которые побывали на моих занятиях в течение многих лет в UCSC Silicon Valley Extension и в Университете Кремниевой долины (ранее Политехнический колледж Когсвелла). Их отзывы, предложения, улыбки, недовольство, моменты озарения, разочарование, понимающие кивки и даже большие пальцы вверх (в классах Zoom в эпоху ковида) были чрезвычайно полезны в формировании содержания этой книги и моего общего стиля преподавания;
- наконец, мою семью, которая поддерживала меня в длительном процессе написания, тестирования, редактирования, переписывания, редактирования, отладки, редактирования, переписывания, редактирования (и так далее) этой книги и связанного с ней кода. Без вас я бы не справился. Я не был уверен, достаточно ли у нас книг в библиотеке, поэтому написал еще одну!

ВВЕДЕНИЕ

Эта книга о технике разработки программного обеспечения под названием *объектно-ориентированное программирование* (ООП) и о том, как его можно использовать с Python.

До ООП программисты применяли подход, известный как *процедурное* или *структурированное программирование*, который включает в себя построение набора функций (процедур) и передачу данных через вызовы к этим функциям. Парадигма ООП дает эффективный способ объединить код и данные в связные единицы, которые подходят для повторного использования.

В процессе подготовки к написанию этой книги я подробно познакомился с существующей литературой и видеоматериалами, изучив конкретные подходы, используемые для объяснения этой важной и широкомасштабной темы. Я обнаружил, что учителя и писатели обычно начинают с определения некоторых ключевых терминов: *класс, переменная экземпляра, метод, инкапсуляция, наследование, полиморфизм* и так далее.

Хотя все это важные понятия и мы их подробно здесь рассмотрим, я все же начну с другого — с вопроса: «Какую проблему мы решаем?» То есть если решением является ООП, то в чем

проблема? Чтобы ответить, я покажу несколько примеров программ, построенных с использованием процедурного программирования, а затем выявлю сложности, присущие этому стилю. А после покажу вам, как объектно-ориентированный подход может упростить разработку и облегчить поддержку таких программ.

Для кого эта книга?

Эта книга предназначена для людей, которые уже знакомы с Python и используют основные функции из его стандартной библиотеки. Я предполагаю, что вы понимаете основной синтаксис языка и можете писать небольшие и средние программы с помощью переменных, инструкций присваивания, операторов `if/elif/else`, а также циклы, функции и вызовы функций, списки, словари и так далее. Если вы не знакомы со всеми этими понятиями, то я предлагаю вам сначала ознакомиться с моей предыдущей работой, Learn to Program with Python 3*.

Издание, которое вы сейчас читаете, среднего уровня, поэтому есть ряд более продвинутых тем, их я не стану затрагивать. Например, чтобы книга оставалась практичной, я не часто буду вдаваться в подробности о внутренней реализации Python. Для простоты и ясности, а также для того, чтобы сосредоточиться на освоении методов ООП, примеры написаны с использованием ограниченного подмножества языка. Есть более продвинутые и лаконичные способы программирования на Python, которые выходят за рамки этой книги.

Я рассмотрю главные детали ООП, не зависящие от языка, но обозначу области, где существуют различия между Python и другими языками ООП. Ознакомившись с основами программирования в стиле ООП в этой книге, при желании вы сможете легко применить те же методы к другим языкам ООП.

Версия(-и) Python и установка

Все примеры кода в этой книге были написаны и протестированы с использованием версий Python с 3.6 по 3.9 и должны хорошо работать с версией 3.6 или новее.

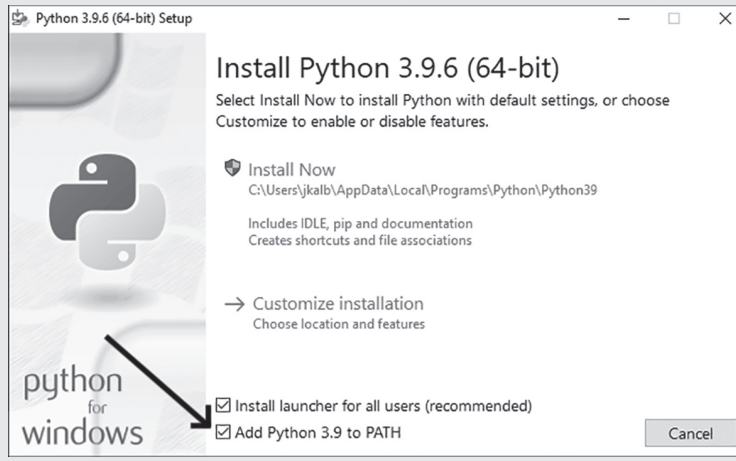
---

* Не издавалась на русском языке. — *Прим. ред.*

Python доступен бесплатно по адресу **https://www.python.org**. Если у вас не установлен Python или вы хотите обновиться до последней версии, перейдите на этот сайт, найдите вкладку Downloads и нажмите кнопку **Download**. Так вы загрузите установочный файл на свой компьютер. Дважды щелкните по нему для установки Python.

УСТАНОВКА НА WINDOWS

При установке на систему Windows необходимо обратить внимание на один важный параметр. При выполнении шагов по установке вы увидите экран, как на рисунке ниже.



В нижней части диалогового окна находится флажок **Add Python 3.x to PATH** (Добавить Python 3.x в переменную PATH). Пожалуйста, не забудьте установить его (по умолчанию он не установлен). Данная настройка позволит правильно инсталлировать пакет `pygame` (который будет представлен в книге позже).

ПРИМЕЧАНИЕ Мне известен «PEP 8 — Руководство по стилю для кода Python» и его конкретная рекомендация использовать Snake Case (змеиный регистр) для имен переменных и функций. Тем не менее я использовал конвенцию об именовании Camel Case (горбатый регистр) в течение многих лет до того, как был написан

документ PEP 8, и мне было комфортно с ним на протяжении моей карьеры. Поэтому все имена переменных и функций в этой книге написаны с использованием горбатого регистра.

Как я объясняю ООП?

Примеры в первых нескольких главах используют текстовый Python; в них программы получают входные данные от пользователя и выводят ему информацию исключительно в виде текста. Я представлю ООП, показывая, как моделировать физические объекты в коде на основе текста. Для начала мы представим в виде объектов выключатели света, диммеры и телевизионные пульты дистанционного управления. Затем я покажу вам, как с помощью ООП моделировать банковские счета и банк в целом.

Как только мы рассмотрим основы ООП, я представлю модуль *pygame*, который позволяет программистам писать игры и приложения, использующие *графический пользовательский интерфейс* (GUI). В программах на основе графического интерфейса пользователь интуитивно взаимодействует с кнопками, флажками, полями ввода и вывода текста и другими удобными виджетами.

Я решил использовать `pygame` с Python, потому что эта комбинация позволяет мне демонстрировать концепции ООП визуально, используя элементы на экране. `Pygame` чрезвычайно портативна и работает практически на всех платформах и операционных системах. Все образцы программ, использующих пакет `pygame`, были протестированы с недавно выпущенной версией `pygame` 2.0.

Я создал пакет под названием `pygwidgets`, который работает с `pygame` и реализует ряд базовых виджетов, все они построены с использованием подхода ООП. Я представлю этот пакет позже в книге и дам пример кода, с которым вы можете работать и экспериментировать. Такой подход позволит увидеть реальные, практические примеры ключевых объектно-ориентированных концепций, при этом внедряя эти приемы для создания забавных игр, в которые можно поиграть. Я также представлю мой пакет `pyghelpers`, содержащий код, что помогает написать более сложные игры и приложения.

Весь приведенный в книге код доступен для скачивания как один файл с сайта **https://addons.eksmo.ru/it/OOP-Code.zip**.

Структура книги

Эта книга состоит из четырех частей. Часть I знакомит с объектно-ориентированным программированием.

- В главе 1 представлен обзор кода с использованием процедурного программирования. Я покажу вам, как реализовать текстовую карточную игру и смоделировать банк, выполняющий операции по одному или нескольким счетам. Попутно я обсуждаю общие проблемы процедурного подхода.
- Глава 2 повествует о классах и объектах и показывает, как вы можете представлять реальные вещи, такие как выключатели света или пульт дистанционного управления телевизором, на Python с помощью классов. Вы увидите, как объектно-ориентированный подход решает проблемы, обозначенные в первой главе.
- В главе 3 представлены две ментальные модели, которые вы можете использовать, чтобы думать о том, что происходит за кулисами, когда вы создаете объекты на Python. Мы будем использовать Python Tutor, чтобы просмотреть код и увидеть, как создаются объекты.
- В главе 4 показан стандартный способ обработки нескольких объектов одного типа путем введения понятия объекта диспетчера объектов. Мы расширим моделирование банковского счета с помощью классов, и я покажу вам, как обрабатывать ошибки с помощью исключений.

Часть II посвящена созданию графических интерфейсов с помощью `pygame`.

- Глава 5 представляет пакет `pygame` и управляемую событиями модель программирования. Мы создадим несколько простых программ, чтобы вы могли начать с размещения графики в окне и обработки ввода с клавиатуры и мыши, а затем разработаем более сложную программу с прыгающими мячами.

- Глава 6 подробнее описывает использование ООП с программами `pygame`. Мы перепишем приложение с прыгающим мячом в стиле ООП и разработаем некоторые простые элементы графического интерфейса.
- В главе 7 представлен модуль `pygwidgets`, который содержит полные реализации многих стандартных элементов графического интерфейса (кнопок, флажков и т. д.), каждый из которых разрабатывается как класс.

Часть III посвящена основным принципам ООП.

- В главе 8 обсуждается инкапсуляция, которая включает в себя сокрытие деталей реализации от внешнего кода и размещение всех связанных методов в одном месте — классе.
- Глава 9 вводит полиморфизм — идею о том, что несколько классов могут иметь методы с одинаковыми именами, — и показывает, как он позволяет вызывать методы в нескольких объектах, не зная типа каждого из них. Мы создадим программу Shapes, чтобы продемонстрировать эту концепцию.
- Глава 10 описывает наследование, которое позволяет создавать набор подклассов, использующих общий код, встроенный в базовый класс, а не «изобретать колесо» с похожими классами. Мы рассмотрим несколько реальных примеров, когда пригодится наследование, например реализацию поля ввода, которое принимает только числа, а затем перепишем наш пример программы Shapes, чтобы использовать эту функциональность.
- Глава 11 завершает эту часть книги обсуждением некоторых дополнительных важных тем ООП, в основном связанных с управлением памятью. Мы посмотрим на время жизни объекта и в качестве примера построим небольшую игру с шариками.

Часть IV посвящена нескольким темам, связанным с использованием ООП в разработке игр.

- Глава 12 демонстрирует, как мы можем перестроить карточную игру, разработанную в главе 1, в качестве программы с графическим интерфейсом на основе `pygame`. Я также

покажу вам, как создавать многоразовые классы колод и карт, которые вы можете использовать при создании других карточных игр.

- В главе 13 рассматривается вопрос о времени. Мы разработаем различные классы таймеров, которые позволят программе продолжать работу, одновременно проверяя заданный лимит времени.

- В главе 14 описываются классы анимации, которые можно использовать для отображения последовательностей изображений. Мы рассмотрим два метода: создание анимации из коллекции отдельных файлов изображений и извлечение и использование нескольких изображений из одного файла листа спрайта.

- Глава 15 объясняет концепцию конечного автомата, представляющего и контролирующего поток ваших программ, и менеджер сцен, который вы можете задействовать для создания программы с несколькими сценами. Чтобы продемонстрировать использование каждого из них, мы построим две версии игры «Камень, ножницы, бумага».

- В главе 16 обсуждаются различные типы модальных диалогов — еще одна важная функция взаимодействия с пользователем. Затем мы пройдемся по созданию полнофункциональной видеоигры на основе ООП под названием Dodger, которая демонстрирует многие методы, описанные в книге.

- В главе 17 представлена концепция шаблонов проектирования на примере шаблона контроллера представления модели, а затем показана программа бросания костей, в которой используется этот шаблон, чтобы позволить пользователю визуализировать данные различными способами. Завершается глава кратким подведением итогов работы над книгой.

Среды разработки

При работе с этой книгой вы будете использовать командную строку только для установки программного обеспечения. Все инструкции по установке четко прописаны, поэтому вам

не потребуется изучать дополнительный синтаксис командной строки.

Я твердо верю, что для разработки лучше использовать не командную строку, а интерактивную среду разработки (IDE). Интегрированная среда обрабатывает многие детали базовой операционной системы за вас и позволяет писать, редактировать и запускать код с помощью одной программы. Интегрированные среды, как правило, являются кросс-платформенными, что позволяет программистам легко перемещаться с Mac на компьютер под управлением Windows или наоборот.

Краткие примеры программ в книге могут быть запущены в среде разработки IDLE, которая устанавливается вместе с Python. IDLE очень проста в использовании и хорошо работает для программ, которые состоят из одного файла. Когда мы перейдем к более сложным программам, использующим несколько файлов Python, лучше применять что-то посложнее; лично я использую среду разработки JetBrains PyCharm, которая легче обрабатывает проекты с несколькими файлами. Community Edition доступен бесплатно по адресу **https://www.jetbrains.com/**, и я настоятельно рекомендую его. В PyCharm входит полностью интегрированный отладчик, который может быть чрезвычайно полезен при написании более крупных программ. Дополнительные сведения об использовании отладчика см. в моем видео на YouTube «Отладка Python 3 с помощью PyCharm» по адресу **https://www.youtube.com/watch?v=cxAOSQQwDJ4&t=43s/**.

Виджеты и примеры игр

В книге представлены и доступны два пакета Python: `pygwidgets` и `pyghelpers`. Используя их, вы сможете создавать полноценные программы с графическим интерфейсом пользователя, но, что более важно, вы получите представление о том, как каждый из виджетов написан как класс и используется как объект.

Примеры игр в книге, включающие различные виджеты, начинаются с относительно простых и становятся все более сложными. Глава 16 рассказывает о разработке и внедрении полнофункциональной видеоигры с таблицей результатов, которая сохраняется в файл.

К концу этой книги вы должны уметь писать собственные игры — карточные или видеоигры в стиле Pong, Hangman, Breakout, Space Invaders и так далее. Объектно-ориентированное программирование дает вам возможность писать программы, которые могут легко отображать и контролировать несколько элементов одного типа, что регулярно требуется при построении пользовательских интерфейсов и часто необходимо в игре.

Объектно-ориентированное программирование — это общий стиль, который можно применять во всех аспектах программирования, далеко за пределами игровых примеров, которые я использую для демонстрации техники ООП. Надеюсь, вам понравится этот подход к изучению ООП.

Итак, начнем.

ЧАСТЬ I

ВВЕДЕНИЕ В ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

Эта часть книги знакомит вас с объектно-ориентированным программированием. Мы обсудим проблемы, присущие процедурному коду, а затем посмотрим, как объектно-ориентированное программирование решает их. Мышление в объектах (с состоянием и поведением) даст вам новое представление о том, как писать код.

В главе 1 представлен обзор процедурного Python. Я начинаю с презентации текстовой карточной игры Higher or Lower («Больше-меньше»), затем прорабатываю несколько все более сложных реализаций банковского счета на Python, чтобы помочь вам лучше понять распространенные проблемы программирования в процедурном стиле.

Глава 2 показывает, как мы можем представлять объекты реального мира в Python с помощью классов. Мы напишем программу, имитирующую выключатель света, изменим ее, чтобы добавить возможности диммера (плавного затемнения), а затем перейдем к более сложному моделированию пульта от телевизора.

Глава 3 дает вам два разных способа думать о том, что происходит за кулисами, когда вы создаете объекты на Python.

Глава 4 демонстрирует стандартный способ обработки нескольких объектов одного типа (например, в такой простой игре, как шашки, где вы должны отслеживать много похожих игровых фигур). Мы расширим программу банковского счета из главы 1 и изучим, как обрабатывать ошибки.