**IT-Колледж “Сириус”**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**ДОКЛАД**

по дисциплине “**Введение в специальность**”

на тему “**ООП**”

Выполнил:  
Студент группы

1.9.7.2

Маланин Дмитрий Алексеевич

Принял:

Старший преподаватель  
Тенигин Альберт Андреевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

IT-Колледж “Сириус”  
2022

# Оглавление

Оглавление

[Оглавление 2](#_Toc122457496)

[Введение 2](#_Toc122457497)

[Объектно-ориентированное программирование 3](#_Toc122457498)

[Классы и объекты 3](#_Toc122457499)

[Принципы ООП 4](#_Toc122457500)

[Инкапсуляция: 4](#_Toc122457501)

[Наследование: 5](#_Toc122457502)

[Полиморфизм 5](#_Toc122457503)

[Как и где используют ООП 6](#_Toc122457504)

[Отличие ООП и функционального программирования 7](#_Toc122457505)

[История ООП 8](#_Toc122457506)

[Примеры ООП на различных языках программирования 9](#_Toc122457507)

[Заключение 11](#_Toc122457508)

[Список Литературы 12](#_Toc122457509)

# Введение

Современное программное обеспечение требуют от разработчиков владения лучшими технологиями по его созданию. Одной из таких технология является объектно-ориентированное программирование (ООП). Чтобы технологически грамотно использовать ООП, необходимо хорошо понимать его основные концепции и научиться мыслить при разработке по его принципам.

# Объектно-ориентированное программирование

**Объектно-ориентированное программирование** — это основная методология программирования 1990-х годов и начала XXI века. Она представляет собой продукт более 35 лет практики и опыта, которые восходят к использованию языка Simula. Идея, которую предложили разработчики этого языка оказалась судьбоносной в истории развития языков программирования. Позже на этих идеях были построены самые известные объектно-ориентированные языки, такие как Java, C++, C# и многие другие. В этой методологии программа разбивается на множество взаимодействующих элементов — объекты.

В методологии ООП программа разбивается на множество взаимодействующих элементов — **объекты**. Объект — это то, что может быть индивидуально описано и рассмотрено. Так же в этой методологии реализуются **классы.** Класс – это способ описания сущности, определяющий состояние и поведение.

# Классы и объекты

В реальном мире нас окружают объекты. Мы учимся в аудиториях, перемещаемся на транспорте, работаем на компьютерах, звоним по телефонам и т. д. Все это — задачи и объекты реального мира. В ООП важно понимать, что описание задачи нужно вести в объектах.

Класс представляет собой макет или шаблон, который определяет общее поведение объекта. У каждого класса есть свои определённые поля и методы, которые он реализует.

Для примера рассмотрим класс “Персонаж(маг) в Компьютерной игре”

Поля:

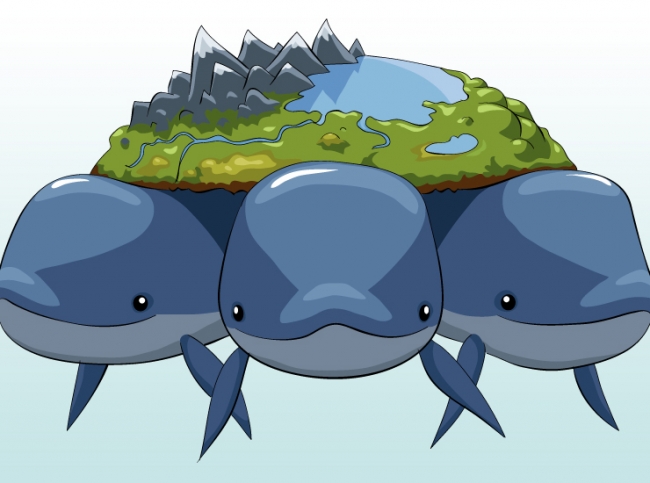
* Имя: Wizard
* 100 hp (health points)
* 200 mp (mana points)
* 0 ammo

Методы:

* Атаковать
* Защищаться
* Использовать заклинания (расходовать mp)

Можно сделать вывод, что класс — это абстрактный тип данных, который имеет собственные поля и методы, а объект – это экземпляр класса. Если представить это в более формальном виде, то можно сказать, что класс – чертёж дома, экземпляр – дом, построенный по чертежу.

# Принципы ООП

 Основных принципа 3, которые иногда называют “три кита ооп”

* инкапсуляция
* наследование
* полиморфизм

Далее все примеры кода будут приводится на языке “Java”

## Инкапсуляция:

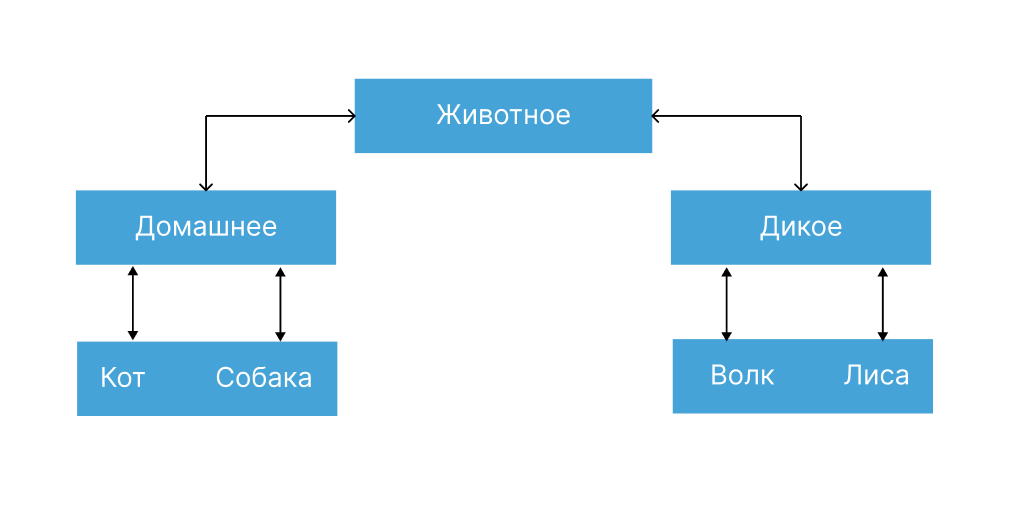
— это механизм, который объединяет данные и код, манипулирующий этими данными, а также защищает и то, и другое от внешнего вмешательства или неправильного использования. Это происходит при помощи модификаторов класса, таких как Default, Public, Private, Protected. Инкапсуляция означает, что методы и поля класса рассматриваются в качестве единого целого.

* Default модификатором доступа по умолчанию , он используются когда другой модификатор не указан.
* Public – это модификатор, который указывает, что метод или переменная может использоваться любым другим классом или программой, имеющей доступ к файлу.
* Private используется для сокрытия методов или переменных класса от других классов и программ, то есть они доступны только внутри класса.
* Protected для определения видимости членов класса в самом классе и в его наследниках.

## Наследование:

— механизм языка, позволяющий описать новый класс на основе уже существующего родительского. Другими словами, при помощи наследования можно построить иерархию классов.

Это будет выглядеть примерно так:



## Полиморфизм

— реализация одинакового по смыслу действия различными способами в зависимости от типа объекта. Простым языком, один интерфейс — множество реализаций.

На примере экземпляра класса можно привести пример для определения полиморфизма:

Рассмотрим на методе “атака”:

|  |  |  |
| --- | --- | --- |
| Wizard | Robot | Knight |
| Использует заклинание | Стреляет Лазером | Бьёт мечом |

# Как и где используют ООП

ООП используется во многих языках программирования, таких как C++, Java, Python и многие другие и может быть применена в различных областях компьютерных технологий, таких как игры, мобильные приложения, веб-приложения, научные приложения и многое другое.

Объектно-ориентированный подход позволяет сделать код более структурированным, в нем легко разобраться стороннему человеку. Благодаря инкапсуляции объектов уменьшается количество ошибок и ускоряется разработка. Так же ускоряется и командная разработка, так как каждый человек может отвечать именно за свой класс. Но такой подход надо использовать не всегда.

Однако стоит отметить, что ООП не является решением для всех проблем и не подходит для всех типов программ. В некоторых случаях другие технологии программирования, такие как функциональное программирование, могут быть более подходящими.

Некоторые примеры областей, где можно встретить ООП:

* Игры:
  + ООП широко используется в разработке компьютерных игр, где она помогает структурировать игровые объекты, такие как игроки, враги, оружие и т.д.
* Веб-приложения:
  + сайты интернет-магазинов
  + социальные сети
  + управление проектами.
* Мобильная разработка:

Вся мобильная разработка выполняется при помощи классов и особенно при помощи методов, например:

* + onCreate - вызывается для первоначального создания фрагмента.
  + onCreateView - для создания компонентов приложения.
  + onActivityCreated - сообщает фрагменту, что его активность завершилась.
  + onStart - делает фрагмент видимым для пользователя.
  + onResume - фрагмент начал взаимодействовать с пользователем
  + onPause - фрагмент больше не взаимодействует с пользователем либо потому, что его активность приостановлена
  + onStop - фрагмент больше не виден пользователю потому, что его активность остановлена
  + onDestroy – Вызывается для окончательной очистки состояния фрагмента.
* Научные приложения:
  + моделирование физических систем
  + анализ данных
  + проектирование
* Базы данных:
  + управление информацией
  + хранение информации
* Операционные системы:
  + Структурирование системы
  + Управление функциями системы

# Отличие ООП и функционального программирования

Объектно-ориентированное программирование (ООП) и функциональное программирование - это две различные технологии программирования, которые различаются по тому, как они организуют и структурируют код.

ООП сфокусирована на моделировании реальных объектов, их свойств и поведений. В ООП код разбивается на объекты, каждый из которых содержит свои поля и методы. ООП также использует концепции, такие как наследование, полиморфизм, инкапсуляция, чтобы улучшить структуру и организацию кода.

В функциональном программировании функции являются "чистыми" - то есть они не имеют побочных эффектов и всегда возвращают одинаковый результат при одинаковых входных данных. Это позволяет функциональным программам быть более масштабируемыми и предсказуемыми. Функциональное программирование также часто использует концепции, такие как “каррирование”, частичное применение функций и функции высшего порядка.

ООП и функциональное программирование - это разные подходы к организации кода, и они могут быть использованы вместе или отдельно. В конечном итоге выбор технологии зависит от требований к программе и предпочтений разработчика.

# История ООП

Объектно-ориентированное программирование является одной из самых распространенных парадигм программирования, которая была разработана ещё в 1960-х годах.

История ООП начинается с работ американских ученых Алана Кея и Джона Маккарти, которые изучали возможности использования компьютеров в образовательных целях. Они предложили использовать объекты для моделирования реальных сущностей и их взаимодействий.

В 1970-х годах был разработан язык программирования Smalltalk, который стал первым языком, полностью основанным на ООП. Он также интродуцировал концепции, такие как наследование, полиморфизм и абстракция. В это же время были разработаны также языки программирования Simula и Smalltalk-76, которые также были основаны на ООП.

В 1980-х годах были разработаны языки программирования C++ и Java, которые стали очень популярными и широко используемыми в различных областях компьютерных технологий по сегодняшний день. Можно сказать, что эти два языка являются гигантами в ООП.

C++ был разработан Бьерном Страуструпом и является расширением языка С, который был популярен в 1970-х годах. C++ добавил в язык С концепции ООП, то есть добавил тех самых “китов” инкапсуляция, наследование, полиморфизм.

Java был разработан компанией Sun Microsystems в 1995 году и стал одним из самых популярных языков программирования в мире. Он обладает множеством функций, которые делают его привлекательным для разработчиков, в том числе автоматическое управление памятью, переносимость кода, удобный синтаксис и богатая библиотека классов. По наше время он остаётся популярным, но всё чаще про него забывают.

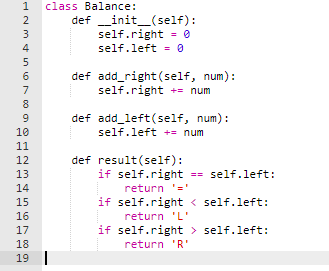
В современное время ООП остаётся значимой и популярной парадигмой, которая используется практически во всех сферах общества.

# Примеры ООП на различных языках программирования





На скриншотах показан код по текстовой игре на языке Java.



На этом скриншоте показан код по реализации весов на Python

Смотря на эти два примера ООП, можно сказать, что каждый язык использует ООП для разных целей. В Java классы создаются для последующей реализации экземпляров. В python ООП в основном реализуются для облегчения каких либо задач.

# Заключение

Объектно-ориентированное программирование (ООП) является одной из основных моделей программирования, которая основана на понятиях объектов, классов, наследования и полиморфизма. Она стремится к тому, чтобы упростить разработку и обслуживание программного обеспечения, обеспечивая решение общих задач с помощью классов и объектов.

Одним из основных преимуществ ООП является то, что она позволяет разработчикам мыслить в терминах объектов реального мира, что упрощает понимание и разработку сложных систем. ООП также позволяет разработчикам легко изменять и расширять существующий код, что существенно упрощает обслуживание программного обеспечения.

В целом, ООП является важной и мощной моделью программирования, которая широко используется в различных областях и на различных платформах. Она позволяет разработчикам строить сложные системы, упрощая их разработку и обслуживание. Однако также существует ряд недостатков ООП, таких как сложность из-за большого количества концепций и терминов, а также более высокая стоимость разработки из-за необходимости реализации дополнительных механизмов ООП.

В заключение, ООП является важным инструментом для разработки программного обеспечения, но ее использование должно быть рассматриваться в соответствии с конкретными требованиями и целями разработки.

# Список Литературы

1. Роберт Сэджвик, Кевин Уэйн Основы программирования на Java, ООП, Алгоритмы и структура данных. - 2017. - 1072 с.
2. Объектно-ориентированное проектирование // Samsung Innovation Campus URL: <https://myitschool.ru/edu/mod/book/view.php?id=551&chapterid=660>
3. Концепции объектно-ориентированного программирования // JAVARUSH URL: <https://javarush.com/groups/posts/1265-koncepcii-obhhektno-orientirovannogo-programmirovanija-java>
4. ООП, «святая троица» и SOLID: некоторый минимум знаний о них // ХАБР URL: <https://habr.com/ru/post/446816/>
5. Что такое классы в объектно-ориентированном программировании // r\_d media URL: https://habr.com/ru/post/446816/
6. Для чего придумали объектно-ориентированное программирование // php.zone URL: <https://php.zone/post/dlya-chego-pridumali-oop>
7. Фрагменты // URL: <https://developer.android.com/reference/android/app/Fragment>
8. Коротко об истории объектно-ориентированного программирования // habr URL: <https://habr.com/ru/post/107940/>
9. История объектно-ориентированного программирования // Виртуальный компьютерный музей URL: <https://www.computer-museum.ru/histsoft/oophist.htm>
10. StackOverFlow URL: <https://survey.stackoverflow.co/2022/#technology-most-loved-dreaded-and-wanted>