

The logo features the word "NEW" in a small, white, sans-serif font, positioned vertically to the left of the word "HardSkills". "HardSkills" is written in a large, white, serif font. A green dot is placed above the final 's' in "Skills", and a green horizontal line is drawn beneath the entire word "HardSkills".

NEW HardSkills

Java



# План занятия

- ООП
- Интерфейсы
- Практика

# ООП

- Что такое ООП
- Класс
- Объект (экземпляр класса)
- Принципы ООП

# ООП

Объектно-ориентированное программирование - это **парадигма** программирования, в которой программное обеспечение рассматривается с точки зрения **совокупности объектов** и **взаимодействия** между ними, а классы образуют **иерархию наследования**.

# Класс

+ Employee
<ul style="list-style-type: none"><li>- position : String</li><li>- card : IdCard</li><li>- room : Room [1..*]</li><li>- department : Department</li><li>- pastPosition : PastPosition [0..*]</li></ul>
<ul style="list-style-type: none"><li>+ Employee(n:String,s:String,p:String)</li><li>+ setPosition(newPosition:String)</li><li>+ getPosition() :String</li><li>+ setIdCard(newIdCard:IdCard)</li><li>+ getIdCard() :IdCard</li><li>+ setRoom(newRoom:Room)</li><li>+ getRoom() :Room[1..*]</li><li>+ deleteRoom(r:Room)</li><li>+ setDepartment(d:Department)</li><li>+ getDepartment() :Department</li><li>+ setPastPosition(p:PastPosition)</li><li>+ getPastPosition() :PastPosition[1..*]</li><li>+ deletePastPosition(p:PastPosition)</li></ul>

Класс - это набор **свойств** и связанных с ними **методов**.

# Объект

Объект (экземпляр класса) - это **производная** единица от класса.

# Принципы ООП

- Наследование
- Инкапсуляция
- Полиморфизм
- Абстракция

# Наследование

Наследование — возможность **перенимать** некоторые **свойства** и **методы** у базовых классов и использовать их в классах-наследниках.

Чтобы наследоваться от какого-либо класса необходимо использовать ключевое слово **extends**.

```
public class Child extends Father {  
}
```



# Наследование

Наследование — концепция объектно-ориентированного программирования, согласно которой абстрактный тип данных может наследовать **данные** и **функциональность** некоторого существующего типа, способствуя **повторному использованию** компонентов программного обеспечения.

# Инкапсуляция

Инкапсуляция — это **сокрытие реализации**.

Инкапсуляция предоставляет **доступ** только к **общим контрактам** и **не позволяет напрямую** влиять на **логику исполнения** какого-либо процесса.

# Инкапсуляция

Инкапсуляция в информатике, программировании — обеспечение **доступности главного**, выделение основного содержания путём помещения всего мешающего, второстепенного в некую условную капсулу (чёрный ящик).

# Полиморфизм

Полиморфизм — это возможность задавать **разную логику** (обработки, поведения и т.п.) в зависимости от **конкретной реализации**.

# Полиморфизм

Полиморфизм — возможность объектов с **одинаковой спецификацией** иметь различную реализацию.

# Абстракция

Абстракция — это **выделение общего** у некоторых связанных сущностей.

Благодаря этому можно не акцентировать внимание на конкретной реализации и проектировать максимально независимые программные модули и т.п.

# Абстракция

Абстракция — в объектно-ориентированном программировании это придание объекту **характеристик**, которые отличают его от всех объектов, четко определяя его **концептуальные границы**.

# Интерфейсы

- Что такое интерфейсы и зачем они нужны
- Интерфейсы в контексте инкапсуляции
- default методы в интерфейсах



# Что такое интерфейсы и зачем они нужны

interface - «*между* лицами»

Интерфейсы нужны для того, чтобы описать **контракты** (публичные методы) и делегировать их реализацию на классы, которые в дальнейшем будут **реализовывать** этот интерфейс.

# Интерфейсы в контексте инкапсуляции

Интерфейсы дают возможность **скрывать реализацию** какой-либо логики. Конечному пользователю предоставляются только **контракты** (публичные методы), а детали реализации остаются за кадром. Причём таких реализаций может быть **множество**.

# default методы в интерфейсах

default методы - это методы, имеющие **реализацию** прямо в интерфейсах

Реализация **по умолчанию** может использоваться для обеспечения **обратной совместимости**.

Вопросы