

Day x - Notes

Java Training



April 1, 2023

Dpoint

[Company address]

# Inheritance:

Mechanism of obtaining properties of parents.

Idea is to create classes that are built on top of existing classes, increase reusability, extending functionality.

Inheritance = parent-child relationship = IS-A relationship

Sub class = child class = derived class

Super class = parent class = base class

**Types of Inheritance:**

1. **Single:** class inheriting another class
2. **Multilevel:** chain of inheritance, Animal > Dog > Baby Dog
3. **Hierarchical:** Two or more class inherit single class

Dog extends Animal, Cat extends Animal.

1. **Multiple:** Class inheriting multiple classes. Not supported in java, to avoid ambiguity of calling a method provided it is present in both the parent class.
2. **Hybrid:** hierarchial + multiple = since multiple is not supported in java, hybrid is not feasible.

# Aggregation In Java:

Class having entity reference of another class in known as Aggregation.

Aggregation = HAS-A relationship.

Example: Employee has the entity reference of Address class.

HelloWorld:

public class HelloWorld {

public static void main(String[] args) {

Person person = new Person("Alice", 25);

Student student = new Student("Bob", 20, "Computer Science");

Employee employee = new Employee("Charlie", 30, "Acme Inc.");

person.sayHello();

student.sayHello();

employee.sayHello();

}

}

Person:

public class Person {

private String name;

private int age;

public Person(String name, int age) {

this.name = name;

this.age = age;

}

public String getName() {

return name;

}

public int getAge() {

return age;

}

public void sayHello() {

System.out.println("Hello, my name is " + name + " and I'm " + age + " years old.");

}

}

Student and person:

public class Student extends Person {

private String major;

public Student(String name, int age, String major) {

super(name, age);

this.major = major;

}

public String getMajor() {

return major;

}

@Override

public void sayHello() {

System.out.println("Hi, my name is " + getName() + " and I'm a student majoring in " + major + ".");

}

}

class Employee extends Person {

private String company;

public Employee(String name, int age, String company) {

super(name, age);

this.company = company;

}

public String getCompany() {

return company;

}

@Override

public void sayHello() {

System.out.println("Hello, my name is " + getName() + " and I work for " + company + ".");

}

}

Final cut all in one with this and super:

public class Person {

private String name;

private int age;

public Person(String name, int age) {

this.name = name;

this.age = age;

}

public String getName() {

return name;

}

public int getAge() {

return age;

}

public void sayHello() {

System.out.println("Hello, my name is " + name + " and I'm " + age + " years old.");

}

}

public class Student extends Person {

private String major;

public Student(String name, int age, String major) {

super(name, age);

this.major = major;

}

public String getMajor() {

return major;

}

@Override

public void sayHello() {

System.out.println("Hi, my name is " + getName() + " and I'm a student majoring in " + major + ".");

}

public void introduce() {

System.out.println("My name is " + getName() + " and I'm " + getAge() + " years old. " +

"I'm majoring in " + major + ".");

}

}

public class Employee extends Person {

private String company;

public Employee(String name, int age, String company) {

super(name, age);

this.company = company;

}

public String getCompany() {

return company;

}

@Override

public void sayHello() {

System.out.println("Hello, my name is " + getName() + " and I work for " + company + ".");

}

public void introduce() {

System.out.println("My name is " + getName() + " and I'm " + getAge() + " years old. " +

"I work for " + company + ".");

}

}