

1. Polymorphism
2. Polymorphism with interfaces
3. instanceof

# Polymorphism

# Polymorphism

## polymorphism **noun**

poly·mor·phism | \,pä-lē-'môr-fi-zəm 

### **Definition of *polymorphism***

: the quality or state of existing in or assuming different forms: such as

**a(1)** : existence of a species in several forms independent of the variations of sex

**(2)** : existence of a gene in several allelic forms

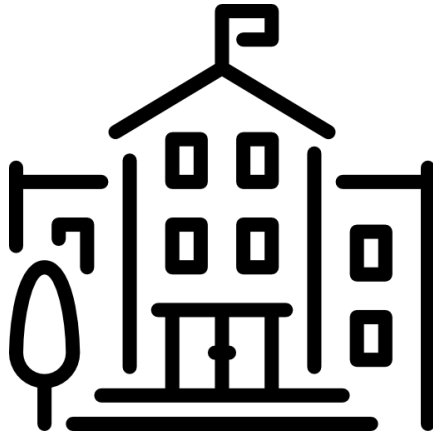
*also* : a variation in a specific DNA sequence


**(3)** : existence of a molecule (such as an enzyme) in several forms in a single species

Inheritance allow an object to become polymorphic

## SCHOOL LEARNING PLATFORM

---



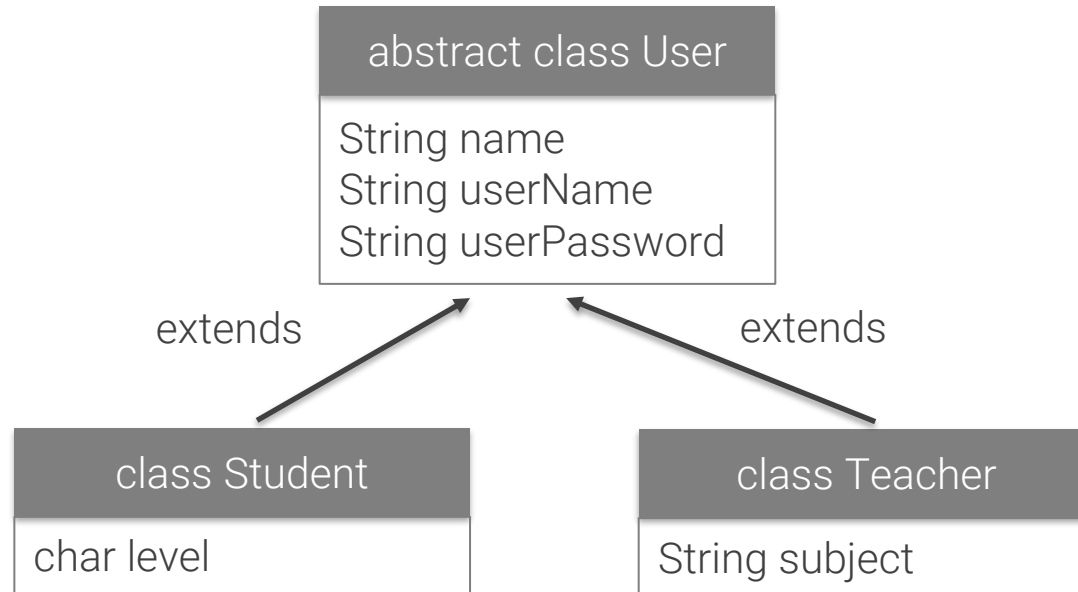


Elena Roig Pas

username: elenaroigpas

password: elenaroigpas73

## SCHOOL LEARNING PLATFORM



## SCHOOL LEARNING PLATFORM

---

```
public abstract class User{
    protected String name;
    protected String userName;
    protected String userPassword;

    public User(String name) {
        this.name = name;
        this.userName = createUser_name();
        this.userPassword = createUserPassword();
    }

    private String createUser_name(){ ... }
    private String createUserPassword(){ ... }
    public abstract String display();
}
```

User.java

## SCHOOL LEARNING PLATFORM

---

```
public class Student extends User{
    private char level; //P:Primary, M:Middle School, H: High School

    public Student(String name, char level) {
        super(name);
        this.level = level;
    }

    public String display() {
        return "Student{" +
            "name=" + name + "\" +
            ", userName=" + userName + "\" +
            ", userPassword=" + userPassword + "\" +
            ", level=" + level + "\" +
            '}'";
    }

    public char getLevel(){return this.level;}
}
```

Student.java



## SCHOOL LEARNING PLATFORM

---

```
public class Teacher extends User{
    private String subject; //Maths, Science, Music

    public Teacher(String name, String subject) {
        super(name);
        this.subject = subject;
    }

    public String display() {
        return "Teacher{" +
            "name=" + name + "\" +
            ", userName=" + userName + "\" +
            ", userPassword=" + userPassword + "\" +
            ", subject =" + subject + "\" +
            '}'";
    }

    public String getSubject(){return this.subject;}
}
```

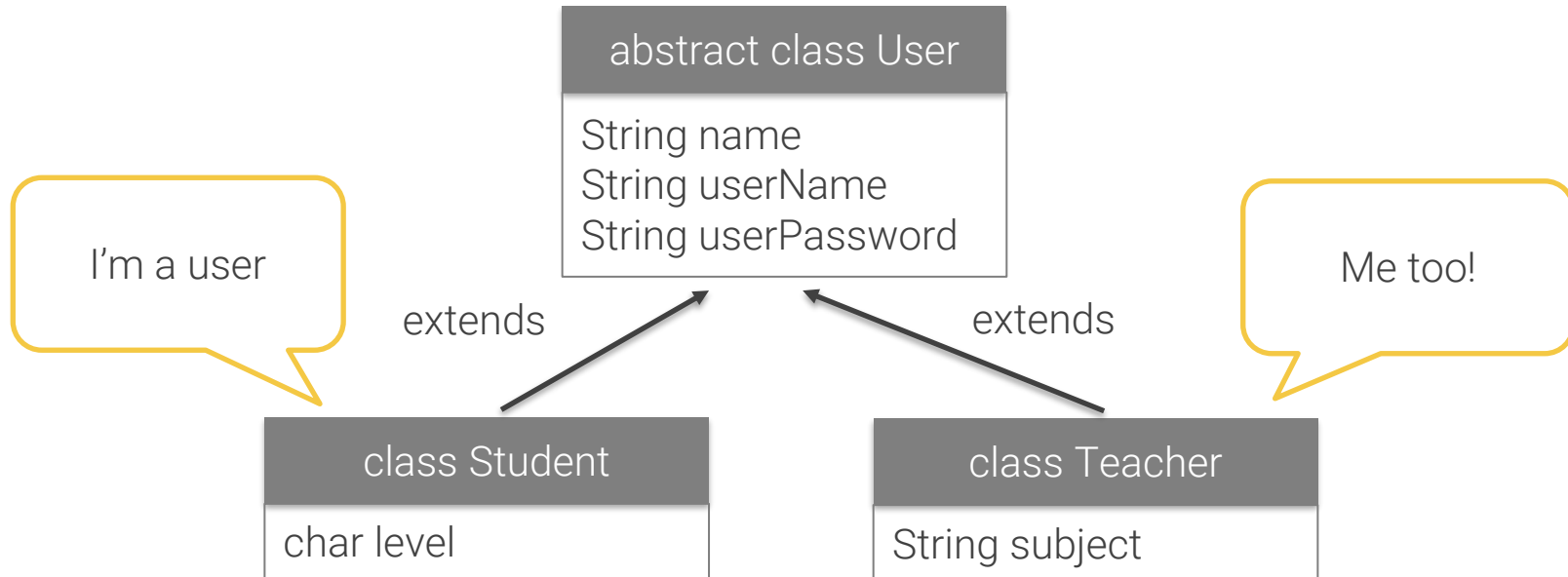
Teacher.java

## SCHOOL LEARNING PLATFORM

```
public class Main {  
  
    public static void main(String[] args) {  
  
        Student student = new Student("Elena Pérez Roig", 'P');  
  
        Teacher teacher= new Teacher("Manuel Díaz Velasco", "Science");  
  
    }  
}
```

Main.java

## SCHOOL LEARNING PLATFORM



## SCHOOL LEARNING PLATFORM

---

```
public class Main {  
  
    public static void main(String[] args) {  
  
        Student student = new Student("Elena Pérez Roig", 'P');  
  
        Teacher teacher= new Teacher("Manuel Díaz Velasco", "Science");  
  
    }  
}
```

Main.java

## SCHOOL LEARNING PLATFORM

---

```
public class Main {  
  
    public static void main(String[] args) {  
  
        User student = new Student("Elena Pérez Roig", 'P');  
  
        User teacher= new Teacher("Manuel Díaz Velasco", "Science");  
  
    }  
}
```

Main.java

## SCHOOL LEARNING PLATFORM

---



## SCHOOL LEARNING PLATFORM

---

```
public class Main {  
    public static void main(String[] args) {  
        ArrayList<User> primaryClassroom = new ArrayList<>();  
        Student student = new Student("Elena Pérez Roig", 'P');  
        primaryClassroom.add(student); //Elena  
        student = new Student("Mercedes Requena Sanz", 'P');  
        primaryClassroom.add(student); //Elena, Mercedes  
        Teacher teacher= new Teacher("Manuel Díaz Velasco", "Science");  
        primaryClassroom.add(teacher); //Elena, Mercedes, Manuel  
        ...  
    }  
}
```

Main.java

## SCHOOL LEARNING PLATFORM

---

```
public class Main {  
  
    public static void main(String[] args) {  
        ArrayList<User> primaryClassroom = new ArrayList<>();  
        Student student = new Student("Elena Pérez Roig",'P');  
        primaryClassroom.add(student); //Elena  
        student = new Student("Mercedes Requena Sanz",'P');  
        primaryClassroom.add(student); //Elena, Mercedes  
        Teacher teacher= new Teacher("Manuel Díaz Velasco", "Science");  
        primaryClassroom.add(teacher); //Elena, Mercedes, Manuel  
        ...  
    }  
}
```

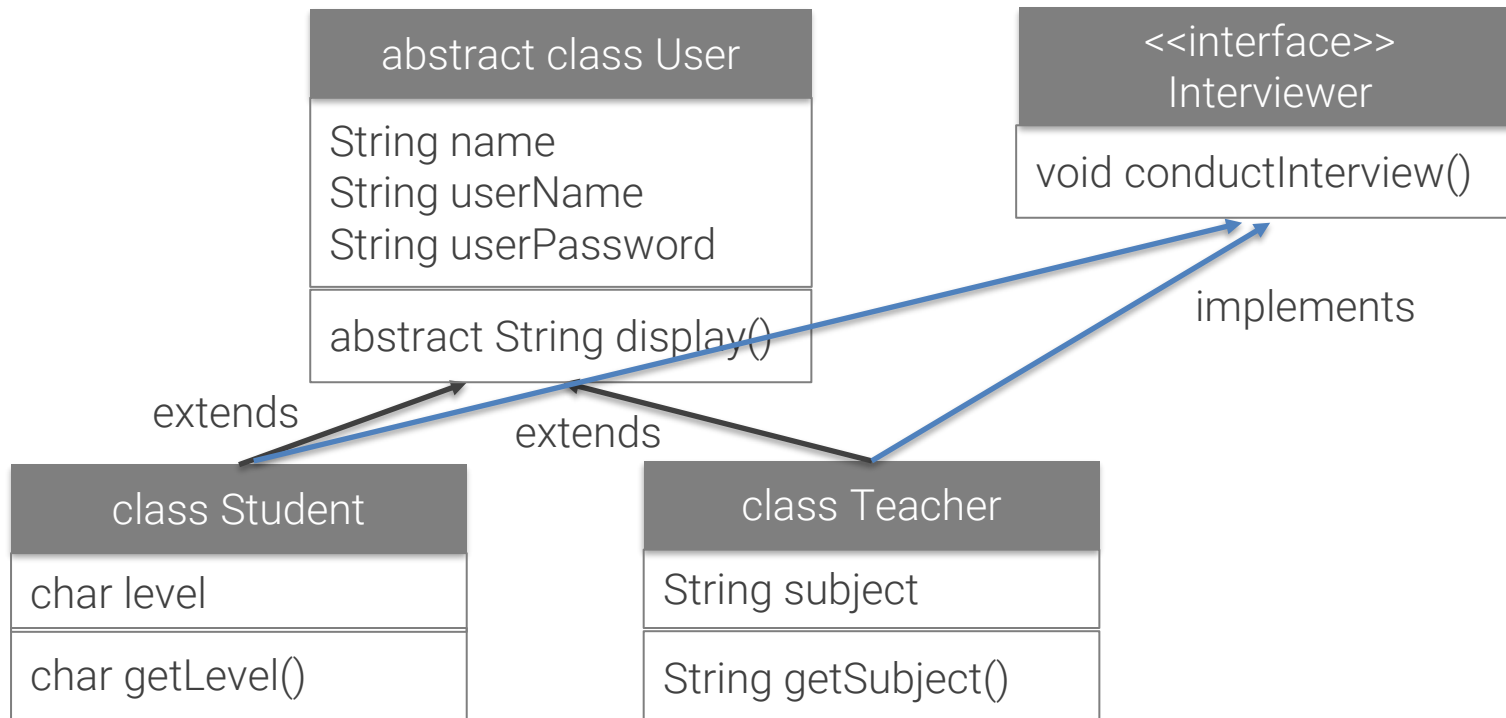
Main.java



# Polymorphism with interfaces

## 2. Polymorphism with interfaces

### SCHOOL LEARNING PLATFORM



### SCHOOL LEARNING PLATFORM

---

```
public class Main {  
  
    public static void main(String[] args) {  
  
        Student student = new Student("Elena Pérez Roig", 'P');  
  
        Teacher teacher= new Teacher("Manuel Díaz Velasco", "Science");  
  
    }  
}
```

Main.java

### SCHOOL LEARNING PLATFORM

---

```
public class Main {  
  
    public static void main(String[] args) {  
  
        Interviewer student = new Student("Elena Pérez Roig", 'P');  
  
        Interviewer teacher = new Teacher("Manuel Díaz Velasco", "Science");  
  
    }  
}
```

Main.java

### SCHOOL LEARNING PLATFORM

---

```
public class Main {  
  
    public static void main(String[] args) {  
  
        Student student = new Student("Elena Pérez Roig", 'P');  
        Teacher teacher= new Teacher("Manuel Díaz Velasco", "Science");  
  
        createInterview(student);  
        createInterview(teacher);  
  
    }  
  
    public static void createInterview(Interviewer interviewer){  
        interviewer.conductInterview();  
    }  
}
```



Main.java

### SCHOOL LEARNING PLATFORM

---

```
public class Main {  
    public static void main(String[] args) {  
        Student student = new Student("Elena Pérez Roig", 'P');  
        Teacher teacher= new Teacher("Manuel Díaz Velasco", "Science");  
  
        createInterview(student);  
        createInterview(teacher);  
    }  
  
    public static void createInterview(Student interviewer){  
        interviewer.conductInterview();  
    }  
  
    public static void createInterview(Teacher interviewer){  
        interviewer.conductInterview();  
    }  
}
```

Main.java

instanceof

## SCHOOL LEARNING PLATFORM

---

```
public class Main {  
  
    public static void main(String[] args) {  
  
        User student = new Student("Elena Pérez Roig", 'P');  
  
    }  
}
```

Super class type  
Interface type

Main.java



## SCHOOL LEARNING PLATFORM

---

```
public class Main {  
  
    public static void main(String[] args) {  
  
        User student = new Student("Elena Pérez Roig", 'P');  
        System.out.println(student.display());  
  
    }  
}
```

Main.java

## SCHOOL LEARNING PLATFORM

---

```
public class Main {  
  
    public static void main(String[] args) {  
  
        User student = new Student("Elena Pérez Roig", 'P');  
        System.out.println(student.display());  
        System.out.println(student.getLevel());  
  
    }  
}
```

Main.java

## SCHOOL LEARNING PLATFORM

```
public class Main {  
  
    public static void main(String[] args) {  
  
        User student = new Student("Elena Pérez Roig", 'P');  
        System.out.println(student.display());  
        System.out.println(((Student)student).getLevel());  
  
    }  
}
```



Casting

Main.java

## SCHOOL LEARNING PLATFORM

---

```
public class Main {  
  
    public static void main(String[] args) {  
        ArrayList<User> primaryClassroom = new ArrayList<>();  
        Student student = new Student("Elena Pérez Roig",'P');  
        primaryClassroom.add(student); //Elena  
        student = new Student("Mercedes Requena Sanz",'P');  
        primaryClassroom.add(student); //Elena, Mercedes  
        Teacher teacher= new Teacher("Manuel Díaz Velasco", "Science");  
        primaryClassroom.add(teacher); //Elena, Mercedes, Manuel  
        ...  
    }  
}
```

Main.java

## SCHOOL LEARNING PLATFORM

---

```
public class Main {  
  
    public static void main(String[] args) {  
        ArrayList<User> primaryClassroom = new ArrayList<>();  
        ...  
        for (User user : primaryClassroom){  
            System.out.println(user.display());  
            //Is Student? -> System.out.println(((Student)user).getLevel());  
        }  
    }  
}
```

Main.java

## SCHOOL LEARNING PLATFORM

---

```
public class Main {  
  
    public static void main(String[] args) {  
        ArrayList<User> primaryClassroom = new ArrayList<>();  
        ...  
        for (User user : primaryClassroom){  
            System.out.println(user.display());  
  
            //Is Student?  
            if (user instanceof Student){  
                System.out.println(((Student) user).getLevel());  
            }  
        }  
    }  
}
```

Main.java

## SCHOOL LEARNING PLATFORM

---

```
public class Main {  
  
    public static void main(String[] args) {  
        ArrayList<User> primaryClassroom = new ArrayList<>();  
        ...  
        int countTeachers = 0, countStudents = 0;  
        for (User user : primaryClassroom){  
            /*insert code here*/  
        }  
    }  
}
```

Main.java

## SCHOOL LEARNING PLATFORM

---

```
public class Main {  
  
    public static void main(String[] args) {  
        ArrayList<User> primaryClassroom = new ArrayList<>();  
        ...  
        int countTeachers = 0, countStudents = 0;  
        for (User user : primaryClassroom){  
  
            if (user instanceof Student) countStudents++;  
            else if (user instanceof Teacher) countTeachers++;  
  
        }  
    }  
}
```

Main.java



“Nunca he perdido un juego, solo no tuve el tiempo suficiente para lograr la victoria.”

*Vince Lombardi, considerado uno de los mejores entrenadores de futbol americano*

