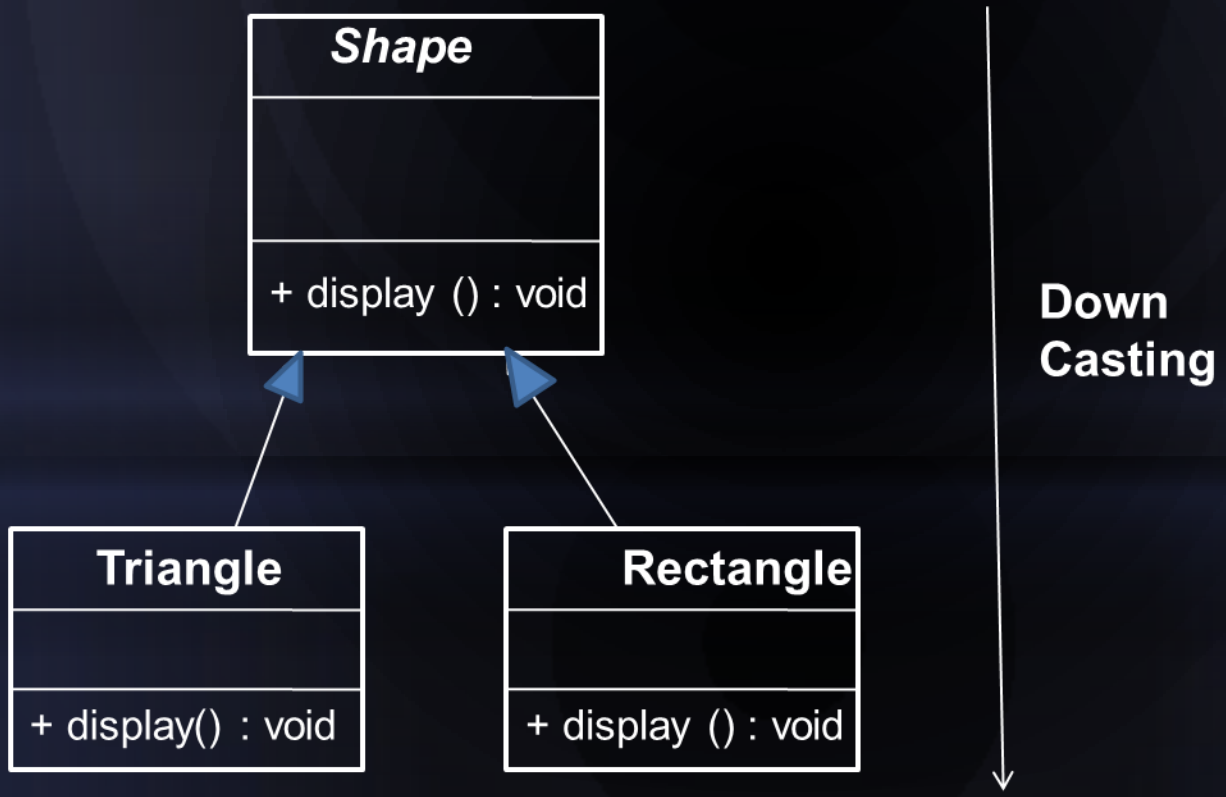
Advanced Java Programming

Review Java Concepts – Polymorphism

Eriq Muhammad Adams J
eriq.adams@ub.ac.id | http://eriq.lecture.ub.ac.id

What's Polymorphism ?

* is an ability that object can have several form.
* an object can be change to another type of object according it hierarchical (casting).



Using Parent Reference to Access Child Members (cont.)

```
public class Parent{
        // properties are omitted …
        public void method1(){
            System.out.prinln("Method1 in parent");
        }
}
public class Child extends Parent{
        // properties are omitted …
        public void method1(){
            System.out.prinln("Method1 di child");
        }
        public void method2(){
            System.out.prinln("Method2 di child");
        }}
```

Using Parent Reference to Access Child Members (cont.)

```
public class PolimorphismDemo{
    public static void main(String [] args){
        Parent parent = new Child();
         // it's fine … virtual method invocation.
         parent.method1();
         parent.method2();  // doesn't compile
         ((Child) parent).method2(); // Yup, it's fine …
    }
}
```

instanceOf

* is used to check type of reference
* Example :

```
if(triangle instanceOf Shape) {
}
```

Polymorphic Parameters Demo (Employee.java)

```
public class Employee
{
   private String name;
   private String address;
   private int number;
   public Employee(String name, String address, int number)
   {
            System.out.println("Constructing an Employee");
            this.name = name;
            this.address = address;
            this.number = number;
   }
   public void mailCheck()
   {
            System.out.println("Mailing a check to " + this.name + " " + this.address);
   }
```

Polymorphic Parameters Demo (Employee.java)

```
public String toString()
{
        return name + " " + address + " " + number;
}
public String getName()
{
        return name;
}
public String getAddress()
{
        return address;
}
public void setAddress(String newAddress)
{
        address = newAddress;
}
```

Polymorphic Parameters Demo (Employee.java)

```
    public int getNumber()
    {
            return number;
    }
}
```

Polymorphic Parameters (Salary.java)

```
public class Salary extends Employee
{
   private double salary; //Annual salary
   public Salary(String name, String address, int number, double salary)
   {
            super(name, address, number);
            setSalary(salary);
   }
   public double getSalary()
   {
            return salary;
   }
```

Polymorphic Parameters Demo (Salary.java)

```
    public void setSalary(double newSalary)
    {
            if(newSalary >= 0.0)
            {
                    salary = newSalary;
            }
    }
    public double computePay()
    {
            System.out.println("Computing salary pay for " + getName());
            return salary/52;
    }
}
```

Polymorphic Parameters Demo (Hourly.java)

```
public class Hourly extends Employee
{
   private double hourlyRate, hoursWorked;
   public Hourly(String name, String address, int number, double hourlyRate)
   {
            super(name, address, number);
            setHourlyRate(hourlyRate);
   }
   public double getHourlyRate()
   {
            return hourlyRate;
   }
```

Polymorphic Parameters Demo (Hourly.java)

```
    public void setHourlyRate(double newRate) {
            if(newRate >= 0.0 && newRate <= 200.00)
            {
                    hourlyRate = newRate;
            }
    }
    public double getHoursWorked() {
            return hoursWorked;
    }
    public void setHoursWorked(double h)
    {
            if(h >= 0 && h <= 80)
            {
            hoursWorked = h;
            }}
}
```

Polymorphic Parameters (Hourly.java)

```
    public double computePay()
    {
            System.out.println("Computing hourly pay for " + getName());
            if(hoursWorked <= 40)
            {
                    return hourlyRate * hoursWorked;
            }
            else
            {
                    return hourlyRate * 40.0 + hourlyRate * 1.5 * (hoursWorked - 40);
            }
    }
}
```

Polymorphic Parameters Demo (Boss.java)

```
public class Boss
{
   public void payEmployee(Employee e) {
            double pay = 0.0;
            if(e instanceof Salary)
            {
                     pay = ((Salary) e).computePay();
            }
            else if(e instanceof Hourly)
            {
                     pay = ((Hourly) e).computePay();
            }
            System.out.println("Pay = " + pay);
            e.mailCheck();
    }
}
```

Next Chapter

* Continue reading about Collections , Generics, & Exceptions