Inheritance

- When the child class is inheriting properties from parent class.
- Child class = properties of parent class + unique properties of child class
- `extends` allows child class to inherit variables from parent class.

Constructors

- A constructor in Java is **a special method that is used to initialize objects**. The constructor is called when an object of a class is created. It can be used to set initial values for object attributes. In Java, a constructor is a block of codes similar to the method.

Parent Class

```java
package com.inclass.properties.inheritance;

public class Box {
    double l;
    double h;
    double w;
        // default constructor
    Box() {
        this.l = 0;
        this.h = 0;
        this.w = 0;
    }
    // cube constructor
    Box (double side) {
        this.l = side;
        this.h = side;
        this.w = side;
    }
    // cuboid constructor
    Box(double l, double h, double w) {
        this.l = l;
        this.h = h;
        this.w = w;
    }
    // copy constructor
    Box (Box old) {
        this.h = old.h;
        this.l = old.l;
        this.w = old.w;
    }
    // method
    public void information() {
        System.out.println("Running the box");
    }
}
```

Child Class

```
package com.inclass.properties.inheritance;

public class BoxWeight extends Box{

    double weight;

    public BoxWeight() {
        this.weight = 0;

    }
    public BoxWeight(double l, double h, double w, double weight) {
        // private variables in parent class, cannot be accessed in child class
        super(l, h, w);
        this.weight = weight;

        /*  Error!
        this.weight = weight;
           super(l, h, w);
        */

        System.out.println(this.w);
        System.out.println(super.w);


    }

    BoxWeight(BoxWeight other) {
        super(other);
        // NOTE: This line does not give any error
        // Box(Box old) {} constructor is used in parent class
           // Similar to Box box6 = new BoxWeight(1, 2, 3, 4);
        weight = other.weight;
    }
}
```

Code

```
package com.inclass.properties.inheritance;

public class Main {
    public static void main(String[] args) {
                Box box = new Box();
                System.out.println(box.l + ", " + box.w + ", " + box.h);

                Box box2 = new Box(4);
                System.out.println(box2.l + ", " + box2.w + ", " + box2.h);

                Box box3 = new Box(1, 2, 3);
                System.out.println(box3.l + ", " + box3.w + ", " + box3.h);

                Box box4 = new Box(box2);
                System.out.println(box4.l + ", " + box4.w + ", " + box4.h);
    }
}
```

Output

```
0.0, 0.0, 0.0
4.0, 4.0, 4.0
1.0, 3.0, 2.0
4.0, 4.0, 4.0
```

Code

```
package com.inclass.properties.inheritance;

public class Main {
    public static void main(String[] args) {
                Box box6 = new BoxWeight(1, 2, 3, 4);
                System.out.println(box6.l);
                System.out.println(box6.h);
                System.out.println(box6.w);
                System.out.println(box6.weight);
        }
}
```

Bug

1.0
2.0
3.0

Error

Important

- The type of the reference variable (not the object), that actually detemines what variables can be accessed.
- The parent class cannot access variables in child class.

Code

```
package com.inclass.properties.inheritance;

public class Main {
    public static void main(String[] args) {
            BoxWeight box7 = new Box(2, 3, 4);
    }
}
```

Error

Important

Here the object itself is of parent class and it is not possible to call the constructor of parenct class.

Super keyword

- `super` keyword calls the parent class constructor directly above.
- In Inheritance, variable of parent class has to be initialised first, before the child variables are initialised. Therefore, `super` keyword has to be used first.

Success

```
super(l, h, w);
this.weight = weight;
```

Error

```
this.weight = weight;
super(l, h, w);
```

- `super` can be used instead of `this`, it directly accesses the variable at parent class.

Success

```
System.out.println(this.w);
System.out.println(super.w);
```

- While creating constructor, making a copy of the child class, `super(child_class_object)` calls the `parent(patent_class_object)` (like typecasting float -> integer)

Success

```
BoxWeight(BoxWeight other) {
        super(other);
        // Box(Box old) {} constructor is used in parent class
           // Similar to Box box6 = new BoxWeight(1, 2, 3, 4);
        weight = other.weight;
    }
```

Types of Inheritance

1. **Simple Inheritance**:
   One class extends another class.
2. **Multi Inheritance**:
   One class extends more than one class.
3. **Hierarchical Inheritance**:
   One class is inherited from many class.
4. **Hybrid Inheritance**:
   Both Multi and Hierarchical Inheritance.