Class **Object**

Class **Object**

Constructor

Constructor

```java
/**
 * Constructs a new object.
 */
@HotSpotIntrinsicCandidate
public Object() {}
```

Methods

getClass(): Class<?>: example

```java
public class Test {
    public static void main(String[] args) {
        Object obj = new String("Hell to world!");
        Class c = obj.getClass();
        System.out.println("Class of Object obj is: " + c.getName());
        // Class of Object obj is: java.lang.String
    }
}
```

hashCode(): int: example

```java
public class Program {
    public static void main(String[] args) {
        Person tom = new Person("Tom");
        System.out.println(tom.hashCode());
    }
}

class Person {
    private String name;

    public Person(String name) {
        this.name = name;
    }

    @Override
    public int hashCode() {
        return 19 * name.hashCode() + 7;
    }
}
```

toString(): String

```java
public String toString() {
    return getClass().getName() + "@" + Integer.toHexString(hashCode());
}
```

toString(): String: example

```java
class Complex {
    private double re;
    private double im;

    public Complex(double re, double im) {
        this.re = re;
        this.im = im;
    }

    @Override
    public String toString() {
        return String.format(re + " + i" + im);
    }
}
```

toString(): String: example

```java
public class DEmo {
    public static void main(String[] args) {
        Complex c1 = new Complex(10, 15);
        System.out.println(c1);
    }
}
```

equals(Object): boolean: example

```java
class Complex {
    private double re;
    private double im;

    public Complex(double re, double im) {
        this.re = re;
        this.im = im;
    }

    @Override
    public boolean equals(Object o) {
        if (o == this) {
            return true;
        } else if (!(o instanceof Complex)) {
            return false;
        } else {
            Complex c = (Complex) o;
            return (Double.compare(re, c.re) == 0) && (Double.compare(im,
        }
    }
}
```

equals(Object): boolean: example

```java
public class Demo {
    public static void main(String[] args) {
        Complex c1 = new Complex(10, 15);
        Complex c2 = new Complex(10, 15);
        if (c1.equals(c2)) {
            System.out.println("Equal ");
        } else {
            System.out.println("Not Equal ");
        }
    }
}
```

finalize(): void: example

```java
public class Test {
    public static void main(String[] args) {
        Test t = new Test();
        System.out.println(t.hashCode());
        t = null;
        System.gc();
        System.out.println("end");
    }

    @Override //@Deprecated(since="9")
    protected void finalize() {
        System.out.println("finalize method called");
    }
}
```

Multithreading methods

- `notify(): void`

- `notifyAll(): void`

- `wait(): void`

- `wait(long): void`

- `wait(long, int): void`

How clone objects?

How clone objects?

- clone(): Object

- Interface Cloneable (**интерфейс-маркер/Marker Interface**)

```
public interface Cloneable {
}
```

clone(): Object

```java
    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }

    @Override
    public String toString() {
        return "Person{" +
                "name='" + name + '\'' +
                ", age=" + age +
                '}';
    }

    @Override
    protected Person clone() throws CloneNotSupportedException {
        super.clone();
        return new Person(this.name, this.age);
    }
}
```

clone(): Object

```java
public class Demo {
    public static void main(String[] args) throws CloneNotSupportedException
        Person lucas = new Person("Lucas", 23);
        System.out.println(lucas);
        Person leo = lucas.clone();
        System.out.println(leo);
        System.out.println(lucas.equals(leo));
    }
}
```