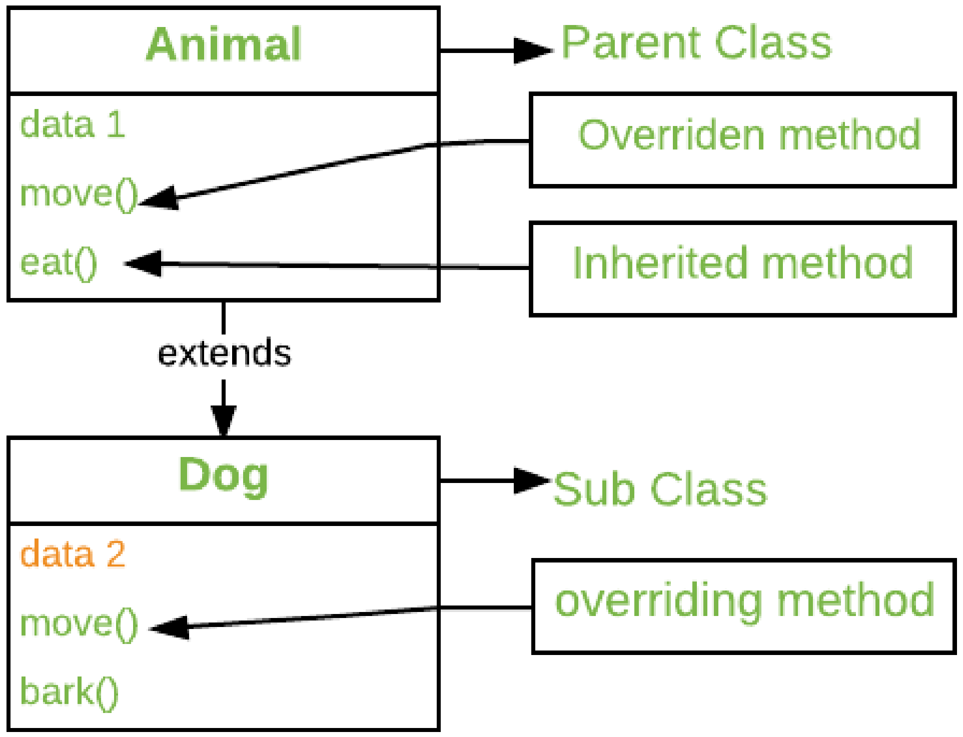
# Method Overriding

- Declaring a method in sub class which is already present in parent class is known as method overriding.
- Overriding is done so that a child class can give its own implementation to a method which is already provided by the parent class.
- In this case the method in parent class is called overridden method and the method in child class is called overriding method.
- Method overriding is one of the way by which java achieve Run Time Polymorphism.

# Method Overriding



## Method Overriding

- we call a method as overriding if it shares these features with one of its superclass' method:

1. The method must have the same name as in the parent class
2. The method must have the same parameter as in the parent class.
3. There must be an IS-A relationship (inheritance).
4. he same or covariant return type

## Example: Hiding with Overriding

-
  ```
  class A {
      int i, j;
      A(int a, int b) {
          i = a; j = b;
      }
      void show() {
          System.out.println("i and j: " + i + " " + j);
      }
  }
  ```

## Example: Hiding with Overriding

-
  ```
  class B extends A {
          int k;
          B(int a, int b, int c) {
          super(a, b);
                  k = c;
          }
          void show() {
                  System.out.println("k: " + k);
          }
  }
  ```

## Example: Hiding with Overriding

- When show() is invoked on an object of type B, the version of show()
- defined in B is used:

```
class Override {
      public static void main(String args[]) {
            B subOb = new B(1, 2, 3);
            subOb.show();
      }
}
```

- The version of show() in A is hidden through overriding.

## Rules For Method Overriding

- The access modifier can only allow more access for the overridden method.
- A final method does not support method overriding.
- A static method cannot be overridden.
- Private methods cannot be overridden.
- The return type of the overriding method must be the same.
- We can call the parent class method in the overriding method using the super keyword.
- A constructor cannot be overridden because a child class and a parent class cannot have the constructor with the same name.

# Overriding versus Overloading

- The show() method in B takes a String parameter, while the show() method in A takes no parameters:

```
class B extends A {
        int k;
        B(int a, int b, int c) {
                super(a, b); k = c;
        }
        void show(String msg) {
                System.out.println(msg + k);
        }
}
```

# Super and Method Overriding

- The hidden super-class method may be invoked using super:
- class B extends A {

```
        int k;
        B(int a, int b, int c) {
                super(a, b);
        k = c;
    }
        void show() {
                super.show();
                System.out.println("k: " + k);
        }
    }
```

- The super-class version of show() is called within the sub-class's version.

## Overriding versus Overloading

- Method overriding occurs only when the names and types of the two methods (super-class and sub-class methods) are identical.
- If not identical, the two methods are simply overloaded:

```
•  class A {
        int i, j;
        A(int a, int b) {
                i = a; j = b;
        }
   void show() {
        System.out.println("i and j: " + i + " " + j);
        }
•  }
```

## Overriding versus Overloading

- The two invocations of show() are resolved through the number of arguments (zero versus one):

```
• class Override {
        public static void main(String args[]) {
              B subOb = new B(1, 2, 3);
              subOb.show("This is k: ");
              subOb.show();
        }
• }
```

# Overriding versus Overloading

| Overloading | Overriding |
|---|---|
| 1. Whenever same method or Constructor is existing multiple times within a class either with different number of parameter or with different type of parameter or with different order of parameter is known as Overloading. | Whenever same method name is existing multiple time in both base and derived class with same number of parameter or same type of parameter or same order of parameters is known as Overriding. |
| 2. Arguments of method must be different at least arguments. | Argument of method must be same including order. |

# Overriding versus Overloading

| Overloading | Overriding |
|---|---|
| 3. Method signature must be different. | Method signature must be same. |
| 4. Private, static and final methods can be overloaded. | Private, static and final methods can not be override. |
| 5. Access modifiers point of view no restriction. | Access modifiers point of view not reduced scope of Access modifiers but increased. |

# Overriding versus Overloading

| Overloading | Overriding |
|---|---|
| 6. Also known as compile time polymorphism or static polymorphism or early binding. | Also known as run time polymorphism or dynamic polymorphism or late binding. |
| 7. Overloading can be exhibited both are method and constructor level. | Overriding can be exhibited only at method label. |
| 8. The scope of overloading is within the class. | The scope of Overriding is base class and derived class. |
| 9. Overloading can be done at both static and non-static methods | Overriding can be done only at non-static method. |