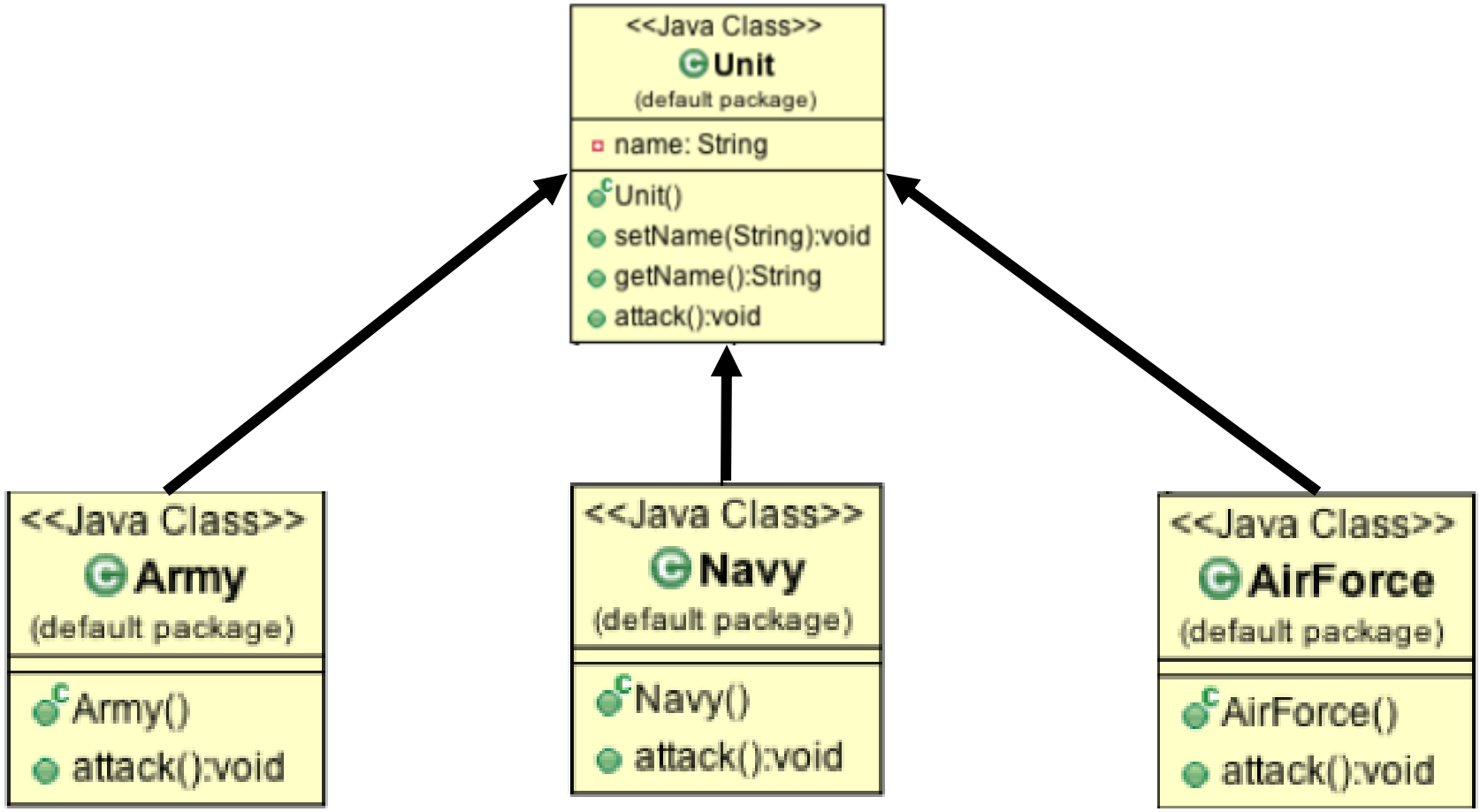
Override

1. 다형성 구현


<<Java Class>>
Unit
(default package)
name: String
Unit()
setName(String):void
getName():String
attack():void
<<Java Class>>
Army
(default package)
Army()
attack():void
<<Java Class>>
Navy
(default package)
Navy()
attack():void
<<Java Class>>
AirForce
(default package)
AirForce()
attack():void

2. Override

```
class Hello {
  public void say() {
    System.out.println("Hello");
  }
}

class English extends Hello {

}

class Korean extends Hello {
  public void say() {
    System.out.println("안녕하세요.");
  }
}

public class Foo {
  public static void main(String[] args) {
    English e = new English();

    e.say();

    Korean k = new Korean();

    k.say();
  }
}
```

3. super(1/4)

```
1 class Hello {
2   public void say() {
3     System.out.println("Hello");
4   }
5 }
6
7 class Korean extends Hello {
8   public void say() {
9     System.out.println("안녕하세요.");
10  }
11
12  public void sayHello() {
13    super.say();
14    this.say();
15  }
16 }
```

3. super(2/4)

```
1  class Hello {
2    public void say() {
3      System.out.println("Hello");
4    }
5  }
6
7  class Korean extends Hello {
8    public void say() {
9      System.out.println("안녕하세요.");
10   }
11 }
12
13 public class Foo {
14   public static void main(String[] args) {
15     Korean k = new Korean();
16     k.say();
17   }
18 }
```

3. super(3/4)

```
1 class Hello {
2   public void say() {
3     System.out.println("Hello");
4   }
5 }
6
7 class Korean extends Hello {
8   public void say() {
9     System.out.println("Hello");
0     System.out.println("안녕하세요.");
1   }
2 }
3
4 public class Foo {
5   public static void main(String[] args) {
6     Korean k = new Korean();
7     k.say();
8   }
9 }
```

3. super(4/4)

```
1 class Hello {
2   public void say() {
3     System.out.println("Hello");
4   }
5 }
6
7 class Korean extends Hello {
8   public void say() {
9     super.say();
10    System.out.println("안녕하세요.");
11  }
12 }
13
14 public class Foo {
15   public static void main(String[] args) {
16     Korean k = new Korean();
17     k.say();
18   }
19 }
```

4. 생성자가 있는 부모 클래스

```java
class Hello {
  public Hello(String msg) {
    System.out.println(msg);
  }
}

class Korean extends Hello {

}
```

```java
class Hello {
  public Hello(String msg) {
    System.out.println(msg);
  }
}

class Korean extends Hello {
  // 부모와 동일한 파라미터를
  // 받도록 생성자를 정의하고,
  // 전달받은 파라미터를 부모에게
  // 재전달 한다.
  public Korean(String msg) {
    super(msg);
  }
}
```