1 重载

如果多个方法有相同的名字、不同的参数就构成了重载。调用方法时，编译器用各个方法定义时的参数类型与方法调用中所使用的值类型进行匹配，来选出正确的方法。如果编译器找不到匹配的参数，就产生编译错误。

Java 允许重载任何方法，而不只是构造器方法。要完整地描述一个方法，需要指定方法名以及参数类型，这叫做方法的**签名**。例如，`String`类有 4 个名为`indexOf`的方法，它们的签名是：

```
indexOf(int)
indexOf(int, int)
indexOf(String)
indexOf(String, int)
```

返回类型不是方法类型的一部分，也就是说，不能有两个名字相同、参数类型也相同却有不同返回类型的方法。

类可以有多个构造器，以满足构造对象的多种需求。

2 默认字段初始化

如果在构造器中没有显式地为字段设置初始值，就会自动赋为默认值：数值为 0、布尔值为`false`、对象引用为`null`。不过这种做法会影响程序代码的可读性，因此不推荐使用。

3 无参数的构造器

如果写一个类时没有编写构造器，编译器就会提供一个无参数构造器，这个构造器将所有实例字段设置为默认值。

如果类中提供了至少一个构造器，就不会自动提供这个无参数的构造器，无参数的构造器需要自己编写。此时如果不编写无参数的构造器，构造对象时不提供参数就是不合法的。

4 显式字段初始化

通过重载类的构造器方法，可以采用多种形式设置类的实例字段的初始状态。不管怎样调用构造器，每个实例字段都要设置为一个有意义的初值。

可以在类定义中直接为任何字段赋值。例如：

```
class Employee
{
    private String name = "";
    // ...
}
```

在执行构造器之前先完成这个赋值操作。如果一个类的所有构造器都希望把某个特定的实例字段设置为同一个值，这个语法就特别有用。

初始值不一定是常量值，也可以利用方法调用初始化一个字段。例如：

```
class Employee
{
    private static int nextId;
    private int id = assignId();
    // ...

    private static int assignId()
    {
        int r = nextId;
        nextId++;
        return r;
    }

    // ...
}
```

5 调用另一个构造器

如果构造器的第一条语句形如`this(...)`，这个构造器将调用同一个类的另一个构造器。例如：

```
public Employee(double s)
{
    this("Employee #" + nextId, s); // 调用 Employee(String, double)
    nextId++;
}
```

采用这种方式非常有用，这样对公共的构造器代码只需要编写一次。

6 初始化块

在一个类的声明中，可以包含任意多个代码块，只要构造这个类的对象，这些块就会被执行。例如：

```
class Employee
{
    private static int nextId;

    private int id;
    private String name;
    private double salary;

    // 初始化块
    {
        id = nextId;
        nextId++;
    }

    public Employee(String n, double s)
    {
        name = n;
        salary = s;
    }

    public Employee()
    {
        name = "";
        salary = 0;
    }

    // ...
}
```

在这个例子中，无论使用哪个构造器构造对象，`id`字段都会在对象初始化块中初始化。首先运行初始化块，然后才运行构造器的主体部分。

这种机制不是必需的，也不常见。通常会直接将初始化代码放在构造器中。

对于静态字段的初始化，可以在类定义中直接赋值，也可以使用静态的初始化块。如果类的静态字段需要很复杂的初始化代码，就可以使用静态的初始化块。例如：

```
static
{
    var generator = new Random(); // 构造一个随机数生成器
    nextId = generator.nextInt(10000); // 返回一个 0~n-1 之间的随机数
}
```

在类第一次加载的时候，将会进行静态字段的初始化。所有的静态字段初始化方法以及静态初始化块都将按照类声明中出现的顺序执行。

调用构造器的具体处理步骤如下：

1. 如果构造器的第一行调用了另一个构造器，则基于所提供的参数执行第二个构造器。
2. 否则，
   （1）所有数据字段初始化为其默认值。
   （2）按照在类声明中出现的顺序，执行所有字段初始化方法和初始化块。
3. 执行构造器主体代码。