Java.lang

1. **Process类**

抽象类Process是用来封装线程的，Process主要用作 由Runtime类的exec()方法创建的对象类型或由ProcessBuilder类的start()方法创建的对象类型的超类。

1. **Runtime类**

Runtime封装了运行时环境，不能实例化Runtime对象，但可以调用静态的Runtime.getRuntime()方法来获取当前Runtime对象的引用，继而可以调用JVM的状态和行为。

1. **内存管理**

获取内存相关信息可以使用totalMemory()方法和freeMemory()方法。

先调用gc()方法，再调用freeMemory()方法获取可用的内存基准。

1. **arraycopy(obj, index, target, index, length)**

将任意类型的数组从一个地方复制到另一个地方；比循环较快

第一个参数是源目标数组，第二个参数源目标数组复制的位置，第三是目标数组，第四是复制到目标数组的索引，第五是复制的长度。

1. **环境属性**

Java通过System.getProperty()方法获取环境变量的值

1. **Object**

Object类是所有类的超类

1. **clone()与Cloneable接口**

clone（）方法产生调用对象的副本，只有实现了Cloneable接口的类才可以被赋值。

Cloneable接口没有定义接口成员。复制可能会导致问题。

1. **Class类 (一般用于反射)**

Class类封装类或接口运行时的状态，Class类型的对象是加载时自动创建的，不能显式的声明Class对象。通常，通过Object类定义的getClass()方法获取Class对象；Class是泛型类。其声明如：

class Class<T>

T代表类或接口的类型对象。

1. **ClassLoader类**
2. **Math类**

Math类中有两个double常量E 和PI；

有三角函数、指数函数、舍入函数

1. **Thread类、ThreadGroup类、Runnable接口支持多线程**
   * + Runnable接口
       - 启动某个单独线程执行的任何类都必须实现Runnable接口，该接口只定义了抽象方法run();该方法是线程的入口点。
     + Thread类
       - Thread类创建新的线程。线程具有优先级，

有三个常量：MAX-PRORITY、MIN-PRORITY、NORM-PRORITY(默认值)。

* + - ThreadGroup创建一组线程

1. **ThreadLocal和InheritableThreadLocal**
   * + ThreadLocal用于创建线程局部变量，每个线程都将具有线程局部变量的一个副本。
     + InheritableThreadLocal用于创建可以被继承的线程局部变量。
2. **Throwable类、SecurityManager类、StackTrackElement类**
3. **Enum枚举类**

枚举是已命名常量的列表，枚举是使用enum关键字创建的‘所有枚举自动继承自Enum。其中Enum是泛型类，

其声明如下：class Enum<E extends Enum<E>>;

其中，E代表枚举类型，Enum没有共用构造函数。

1. **Iterable接口（后续）**

Iterable迭代器，该接口是泛型类，声明如下：interface Iterable<T>{},其中，T代表将进行迭代查的对象的类型。该接口定义了iterator（）方法，Iterable<T> iterator();该方法返回一个包含调用元素的迭代器。

只有实现了此**Iterable**接口的类才能够使用for循环遍历。

# 集合框架

集合接口允许某些方法是可选的；可选方法允许修改集合的内容。

支持可选方法的集合称为可修改的；

不允许修改内容的集合称为不可修改的，如果试图在不可修改的集合上进行修改，则会抛出UnsupportedOperationException异常。

## 1.1 Collection接口

Collection接口是构建集合框架的基础，定义集合的所有类都必须实现该接口。Collection是泛型接口声明如下：

Interface Collection<E>

其中，E制定了集合存储的对象的类型。

Collection拓展了Iterable接口，则所有集合都可以使用for-each风格的for循环遍历；

### 1.1.1 Collection接口方法

Iterator<E> iterator(); 返回调用集合的迭代器；

default Stream<E> parallelStream();返回一个使用调用集合作为元素来源的流，该流能够支持并行操作。（JDK1.8新增）

default Spliterator<E> spliterator();返回调用集合的Spliterator（JDK1.8新增）；

default Stream<E> stream();返回一个使用调用集合作为元素来源的流，是顺序流（JDK1.8新增）

## 1.2 List接口

List接口拓展了Collection接口列表可以包含重复的数据。List是泛型接口。

声明形式：

interface List<E>

E 为存储在列表中对象的类型。

方法：set（）；get（）；

subList（）；指定子列表的开始索引和结束索引。

sort（）方法排序

Collection的sort()方法排序分为两部分：一是排序规则；二是排序算法。

方式为两种：

1. java.lang下面的一个接口：Comparable。可以让自定义对象实现一个Comparable接口，这个接口只有一个方法comparableTo（Object o）

其规则是当前对象与o对象进行比较，返回一个int值，系统根据此值进行排序。

如当前对象>o对象，则返回值>0；

如当前对象=o对象，则返回值=0；

如当前对象<o对象，则返回值<0；

2. java.util下有一个Comparator（比较器）。它拥有compare( )方法，用来比较两个对象

## 1.3 Set接口

同样拓展了Collection接口，其中，集合中不允许有重复的数据。Set是泛型接口。形式如下：

interface Set<E>

## 1.4 SortedSet接口

拓展了Set接口并声明了以升序进行排序的组行为。同样是泛型

## 1.5 Queue接口

拓展了Collection接口，声明队列行为。队列是先进先出的列表。

## 1.6 Deque接口

拓展了Queue接口，并声明了双端队列的行为。既可以像标准队列一样先进先出，又可以像堆栈那样后进先出。