**Java Iterator（迭代器）**

Java迭代器（Iterator）是 Java 集合框架中的一种机制，是一种用于遍历集合（如列表、集合和映射等）的接口。

它提供了一种统一的方式来访问集合中的元素，而不需要了解底层集合的具体实现细节。

Java Iterator（迭代器）不是一个集合，它是一种用于访问集合的方法，可用于迭代 ArrayList 和 HashSet 等集合。

Iterator 是 Java 迭代器最简单的实现，ListIterator 是 Collection API 中的接口， 它扩展了 Iterator 接口。

迭代器接口定义了几个方法，最常用的是以下三个：

next() - 返回迭代器的下一个元素，并将迭代器的指针移到下一个位置。

hasNext() - 用于判断集合中是否还有下一个元素可以访问。

remove() - 从集合中删除迭代器最后访问的元素（可选操作）。

Iterator 类位于 java.util 包中，使用前需要引入它，语法格式如下：

import java.util.Iterator; // 引入 Iterator 类

通过使用迭代器，我们可以逐个访问集合中的元素，而不需要使用传统的 for 循环或索引。这种方式更加简洁和灵活，并且适用于各种类型的集合。

获取一个迭代器

集合想获取一个迭代器可以使用 iterator() 方法:

实例

// 引入 ArrayList 和 Iterator 类

import java.util.ArrayList;

import java.util.Iterator;

public class RunoobTest {

public static void main(String[] args) {

// 创建集合

ArrayList<String> sites = new ArrayList<String>();

sites.add("Google");

sites.add("Runoob");

sites.add("Taobao");

sites.add("Zhihu");

// 获取迭代器

Iterator<String> it = sites.iterator();

// 输出集合中的第一个元素

System.out.println(it.next());

}

}

执行以上代码，输出结果如下：

Google

使用迭代器遍历集合时，如果在遍历过程中对集合进行了修改（例如添加或删除元素），可能会导致 ConcurrentModificationException 异常，为了避免这个问题，可以使用迭代器自身的 remove() 方法进行删除操作。

循环集合元素

让迭代器 it 逐个返回集合中所有元素最简单的方法是使用 while 循环：

while(it.hasNext()) {

System.out.println(it.next());

}

以下输出集合 sites 中的所有元素：

实例

// 引入 ArrayList 和 Iterator 类

import java.util.ArrayList;

import java.util.Iterator;

public class RunoobTest {

public static void main(String[] args) {

// 创建集合

ArrayList<String> sites = new ArrayList<String>();

sites.add("Google");

sites.add("Runoob");

sites.add("Taobao");

sites.add("Zhihu");

// 获取迭代器

Iterator<String> it = sites.iterator();

// 输出集合中的所有元素

while(it.hasNext()) {

System.out.println(it.next());

}

}

}

执行以上代码，输出结果如下：

Google

Runoob

Taobao

Zhihu

删除元素

要删除集合中的元素可以使用 remove() 方法。

以下实例我们删除集合中小于 10 的元素：

实例

// 引入 ArrayList 和 Iterator 类

import java.util.ArrayList;

import java.util.Iterator;

public class RunoobTest {

public static void main(String[] args) {

ArrayList<Integer> numbers = new ArrayList<Integer>();

numbers.add(12);

numbers.add(8);

numbers.add(2);

numbers.add(23);

Iterator<Integer> it = numbers.iterator();

while(it.hasNext()) {

Integer i = it.next();

if(i < 10) {

it.remove(); // 删除小于 10 的元素

}

}

System.out.println(numbers);

}

}

执行以上代码，输出结果如下：

[12, 23]

注意：Java 迭代器是一种单向遍历机制，即只能从前往后遍历集合中的元素，不能往回遍历。同时，在使用迭代器遍历集合时，不能直接修改集合中的元素，而是需要使用迭代器的 remove() 方法来删除当前元素。