首页 HTML CSS **JAVASCRIPT JQUERY** VUE2 VUE3 **BOOTSTRAP** PYTHON3 PYTHON2 JAVA



Java 教程

Java 简介

Java 开发环境配 置

Java 基础语法

Java 对象和类

Java 基本数据类 型

Java 变量类型

Java 修饰符

Java 运算符

Java 循环结构

Java 条件语句

Java switch case

Java Number & Math 类

Java Character 类

Java String 类

Java StringBuffer

Java 数组

Java 日期时间

Java 正则表达式

Java 方法

Java Stream,

File, IO

Java Scanner 类

Java 异常处理

### Java 面向对象

Java 继承

Java

Override/Overload

#### ← Java HashSet

#### Java Iterator ->

# Java HashMap



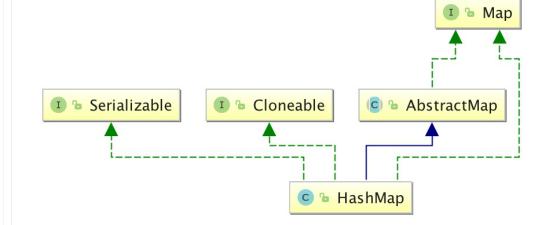
# 多 Java 集合框架

HashMap 是一个散列表,它存储的内容是键值对(key-value)映射。

HashMap 实现了 Map 接口,根据键的 HashCode 值存储数据,具有很快的访问速度,最 多允许一条记录的键为 null , 不支持线程同步。

HashMap 是无序的,即不会记录插入的顺序。

HashMap 继承于AbstractMap,实现了 Map、Cloneable、java.io.Serializable 接口。



HashMap 的 key 与 value 类型可以相同也可以不同,可以是字符串(String)类型的 key 和 value, 也可以是整型 (Integer)的 key 和字符串 (String)类型的 value。

Map<String, String> map = Map.of("google", "google.com", "runoob", "runoob.com");

key	value
google	google.com
runoob	runoob.com

Map<Integer, String> map = Map.of(1, "google", 2, "runoob");

key	value
1	google
2	runoob

### Ⅲ 分类导 航

HTML / CSS

**JavaScript** 

服务端

数据库

移动端

XML 教程

**ASP.NET** 

**Web Service** 

开发工具

网站建设

Advertisement







Java 多态 Java 抽象类 Java 封装 Java 接口 Java 枚举 Java 包(package) Java 高级教程 Java 数据结构 Java 集合框架 Java ArrayList Java LinkedList Java HashSet Java HashMap Java Iterator Java Object Java 泛型 Java 序列化 Java 网络编程 Java 发送邮件 Java 多线程编程 Java Applet 基础 Java 文档注释 Java 实例 Java 8 新特性 Java MySQL 连 接 Java 9 新特性 Java 测验

HashMap 中的元素实际上是对象,一些常见的基本类型可以使用它的包装类。

基本类型对应的包装类表如下:

基本类型	引用类型	
boolean	Boolean	
byte	Byte	
short	Short	
int	Integer	
long	Long	
float	Float	
double	Double	
char	Character	

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

```
import java.util.HashMap; // 引入 HashMap 类
```

以下实例我们创建一个 HashMap 对象 Sites ,整型 (Integer ) 的 key 和字符串 (String ) 类型的 value :

```
HashMap<Integer, String> Sites = new HashMap<Integer, String>();
```

## 添加元素

HashMap 类提供类很多有用的方法,添加键值对(key-value)可以使用 put()方法:

## 实例

```
| // 引入 HashMap 类
| import java.util.HashMap;

| public class RunoobTest {
| public static void main(String[] args) {
| // 创建 HashMap 对象 Sites
| HashMap<Integer, String> Sites = new HashMap<Integer, String
| >();
| // 添加键值对
| Sites.put(1, "Google");
| Sites.put(2, "Runoob");
| Sites.put(3, "Taobao");
| Sites.put(4, "Zhihu");
| System.out.println(Sites);
| }
| }
```



信息来源: superof



믦



执行以上代码,输出结果如下:

```
{1=Google, 2=Runoob, 3=Taobao, 4=Zhihu}
```

以下实例创建一个字符串(String)类型的 key 和字符串(String)类型的 value:

```
实例
```

```
import java.util.HashMap;

public class RunoobTest {
    public static void main(String[] args) {
        // 创建 HashMap 对象 Sites
        HashMap<String, String> Sites = new HashMap<String, String>(
);

    // 添加键值对
    Sites.put("one", "Google");
    Sites.put("two", "Runoob");
    Sites.put("three", "Taobao");
    Sites.put("four", "Zhihu");
    System.out.println(Sites);
    }
}
```

执行以上代码,输出结果如下:

```
{four=Zhihu, one=Google, two=Runoob, three=Taobao}
```

## 访问元素

我们可以使用 get(key) 方法来获取 key 对应的 value:

#### 实例

```
import java.util.HashMap;

public class RunoobTest {
    public static void main(String[] args) {
        // 创建 HashMap 对象 Sites
        HashMap<Integer, String> Sites = new HashMap<Integer, String
>();

// 添加键值对
    Sites.put(1, "Google");
    Sites.put(2, "Runoob");
    Sites.put(3, "Taobao");
    Sites.put(4, "Zhihu");
    System.out.println(Sites.get(3));
}
```

执行以上代码,输出结果如下:

Taobao

出结果如下:

믦

## 删除元素

我们可以使用 remove(key) 方法来删除 key 对应的键值对(key-value):

```
实例
```

```
// 引入 HashMap 类
import java.util.HashMap;
public class RunoobTest {
    public static void main(String[] args) {
        // 创建 HashMap 对象 Sites
        HashMap<Integer, String> Sites = new HashMap<Integer, String</pre>
>();
        // 添加键值对
        Sites.put(1, "Google");
        Sites.put(2, "Runoob");
        Sites.put(3, "Taobao");
        Sites.put(4, "Zhihu");
        Sites.remove(4);
        System.out.println(Sites);
    }
}
```

执行以上代码,输出结果如下:

```
{1=Google, 2=Runoob, 3=Taobao}
```

删除所有键值对(key-value)可以使用 clear 方法:

#### 实例

```
// 引入 HashMap 类
import java.util.HashMap;
public class RunoobTest {
    public static void main(String[] args) {
        // 创建 HashMap 对象 Sites
        HashMap<Integer, String> Sites = new HashMap<Integer, String</pre>
>();
        // 添加键值对
        Sites.put(1, "Google");
        Sites.put(2, "Runoob");
        Sites.put(3, "Taobao");
        Sites.put(4, "Zhihu");
        Sites.clear();
        System.out.println(Sites);
   }
}
```

执行以上代码,输出结果如下:

```
{}
```







如果要计算 HashMap 中的元素数量可以使用 size() 方法:

### 实例

```
import java.util.HashMap;

public class RunoobTest {
    public static void main(String[] args) {
        // 创建 HashMap 对象 Sites
        HashMap<Integer, String> Sites = new HashMap<Integer, String
>();

// 添加键值对
    Sites.put(1, "Google");
    Sites.put(2, "Runoob");
    Sites.put(3, "Taobao");
    Sites.put(4, "Zhihu");
    System.out.println(Sites.size());
}
```

执行以上代码,输出结果如下:

```
4
```

## 迭代 HashMap

可以使用 for-each 来迭代 HashMap 中的元素。

如果你只想获取 key,可以使用 keySet() 方法,然后可以通过 get(key) 获取对应的 value,如果你只想获取 value,可以使用 values() 方法。

#### 实例

```
// 引入 HashMap 类
import java.util.HashMap;
public class RunoobTest {
    public static void main(String[] args) {
        // 创建 HashMap 对象 Sites
        HashMap<Integer, String> Sites = new HashMap<Integer, String</pre>
>();
        // 添加键值对
        Sites.put(1, "Google");
        Sites.put(2, "Runoob");
        Sites.put(3, "Taobao");
        Sites.put(4, "Zhihu");
        // 输出 key 和 value
        for (Integer i : Sites.keySet()) {
            System.out.println("key: " + i + " value: " + Sites.get(
i));
        }
        // 返回所有 value 值
        for(String value: Sites.values()) {
          // 输出每一个value
          System.out.print(value + ", ");
        }
```





```
}
```

### 执行以上代码,输出结果如下:

```
key: 1 value: Google
key: 2 value: Runoob
key: 3 value: Taobao
key: 4 value: Zhihu
```

Google, Runoob, Taobao, Zhihu,

# Java HashMap 方法

hashmap

Java HashMap 常用方法列表如下:

方法	描述			
<u>clear()</u>	删除 hashMap 中的所有键/值对			
clone()	复制一份 hashMap			
isEmpty()	判断 hashMap 是否为空			
size()	计算 hashMap 中键/值对的数量			
put()	将键/值对添加到 hashMap 中			
putAll()	将所有键/值对添加到 hashMap 中			
putlfAbsent()	如果 hashMap 中不存在指定的键 , 则将指定的键/值对插入到 hashMap 中。			
remove()	删除 hashMap 中指定键 key 的映射关系			
containsKey()	检查 hashMap 中是否存在指定的 key 对应的映射关系。			
containsValue()	检查 hashMap 中是否存在指定的 value 对应的映射关系。			
replace()	替换 hashMap 中是指定的 key 对应的 value。			
replaceAll()	将 hashMap 中的所有映射关系替换成给定的函数所执行的结果。			
get()	获取指定 key 对应对 value			
getOrDefault()	获取指定 key 对应对 value,如果找不到 key ,则返回设置的默认值			
forEach()	对 hashMap 中的每个映射执行指定的操作。			
entrySet()	返回 hashMap 中所有映射项的集合集合视图。			



믦믢



keySet()	返回 hashMap 中所有 key 组成的集合视图。
<u>values()</u>	返回 hashMap 中存在的所有 value 值。
merge()	添加键值对到 hashMap 中
compute()	对 hashMap 中指定 key 的值进行重新计算
computelfAbsent()	对 hashMap 中指定 key 的值进行重新计算,如果不存在这个 key,则添加到 hasMap 中
computeIfPresent()	对 hashMap 中指定 key 的值进行重新计算,前提是该 key 存在于 hashMap 中。

更多 API 方法可以查看: <a href="https://www.runoob.com/manual/jdk11api/java.base/java/util/Has">https://www.runoob.com/manual/jdk11api/java.base/java/util/Has</a> hMap.html



<u></u> Java 集合框架



Java Iterator >

# 🕝 点我分享笔记

在线实例	字符集&工	最新更新	站点信息
· HTML 实例	具	· PHP	· 意见反
· CSS 实例	· HTML 字符 集设置	array_key_l  · PHP	· 免责声
· JavaScript 实例	· HTML	array_key_f	· 关于我
	ASCII 字符集	· 9.2 Verilog 可	· 文章归
· Ajax 实例	· HTML ISO- 8859-1	باریا 9.1 Verilog	
· jQuery 实例	· HTML 实体	逻	
· XML 实例	符号	· 8.5 Verilog	
· Java 实例	· HTML 拾色	ACC	<u> </u>
	器	· 8.4 Verilog ACC	关注
	· JSON 格式 化工具	· 8.3 Verilog TF	回激 <b>3</b>

#### 息

- 馈
- 調
- 训
- 档

## 微信



Copyright © 2013-2021 菜鸟 runoob.com All Rights Reserv 号:闽ICP备15012807号









