

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 多态
Java 抽象类
Java 封装
Java 接口
Java 枚举
Java 包(package)

枚举(enum)

是一个特殊的类，一般表示一组常量，比如一年的 4 个季节，一个年的 12 个月，一个星期的 7 天，方向有东南西北等。

类使用 enum 关键字来定义，各个常量使用逗号，来分割。
一个颜色的枚举类。

```
Color
```

```
RED, GREEN, BLUE;
```

枚举 Color 颜色常量有 RED, GREEN, BLUE，分别表示红色，绿色，蓝色。

```
Color
```

```
RED, GREEN, BLUE;
```

```
class Test
```

执行输出结果

```
public static void main(String[] args)
```

```
Color c1 = Color.RED;
System.out.println(c1);
```

代码输出结果为：

如何使用枚举

可以声明在内部类中：

```
class Test
```

```
enum Color
```



<div><div><div><div><div></div><div>Java 教程</div></div><div><div></div><div></div><div></div></div></div><div><div>Java 教程</div><div>Java 简介</div><div>Java 开发环境配置</div><div>Java 基础语法</div><div>Java 对象和类</div><div>Java 基本数据类型</div><div>Java 变量类型</div><div>Java 修饰符</div><div>Java 运算符</div><div>Java 循环结构</div><div>Java 条件语句</div><div>Java switch case</div><div>Java Number & Math 类</div><div>Java Character 类</div><div>Java String 类</div><div>Java StringBuffer</div><div>Java 数组</div><div>Java 日期时间</div><div>Java 正则表达式</div><div>Java 方法</div><div>Java Stream、File、IO</div><div>Java Scanner 类</div><div>Java 异常处理</div><div>Java 面向对象</div><div>Java 继承</div><div>Java Override/Overload</div><div>Java 多态</div><div>Java 抽象类</div><div>Java 封装</div><div>Java 接口</div><div><div><div></div><div>Java 枚举</div></div></div><div>Java 包(package)</div></div></div></div>	<div>RED, GREEN, BLUE;</div> <div>执行输出结果</div> <div><pre>public static void main(String[] args) Color c1 = Color.RED; System.out.println(c1);</pre></div> <div>代码输出结果为:</div> <div></div> <div>是通过 Class 在内部实现的，且所有的枚举值都是 public static final 的。</div> <div>枚举类 Color 转化在内部类实现:</div> <div><pre>Color public static final Color RED = new Color(); public static final Color BLUE = new Color(); public static final Color GREEN = new Color();</pre></div> <div>枚举元素</div> <div>for 语句来迭代枚举元素:</div> <div><pre>Color), GREEN, BLUE; class MyClass { public static void main(String[] args) { for (Color myVar : Color.values()) { System.out.println(myVar); } } }</pre></div> <div>代码输出结果为:</div> <div></div> <div>在 Java 中使用枚举类</div> <div>枚举用于 switch 语句中:</div>
--	---



<div><div><div><div><div></div><div>Java 教程</div></div><div><div></div><div></div><div></div></div></div><div><div></div><div></div><div></div></div></div></div>	<div>Color</div> <div>RED, GREEN, BLUE;</div> <div><pre>class MyClass { public static void main(String[] args) { Color myVar = Color.BLUE; switch (myVar) { case RED: System.out.println("红色"); break; case GREEN: System.out.println("绿色"); break; case BLUE: System.out.println("蓝色"); break; } } }</pre></div>
---	--



反馈/建议

<div><div><div><div><div></div><div>Java 教程</div></div><div><div></div><div></div><div></div></div></div><div><div>Java 教程</div><div>Java 简介</div><div>Java 开发环境配置</div><div>Java 基础语法</div><div>Java 对象和类</div><div>Java 基本数据类型</div><div>Java 变量类型</div><div>Java 修饰符</div><div>Java 运算符</div><div>Java 循环结构</div><div>Java 条件语句</div><div>Java switch case</div><div>Java Number & Math 类</div><div>Java Character 类</div><div>Java String 类</div><div>Java StringBuffer</div><div>Java 数组</div><div>Java 日期时间</div><div>Java 正则表达式</div><div>Java 方法</div><div>Java Stream、File、IO</div><div>Java Scanner 类</div><div>Java 异常处理</div><div>Java 面向对象</div><div>Java 继承</div><div>Java Override/Overload</div><div>Java 多态</div><div>Java 抽象类</div><div>Java 封装</div><div>Java 接口</div><div><div><div></div><div>Java 枚举</div></div></div><div>Java 包(package)</div></div></div><td><div><div><div><div>}</div></div><div><div>// 使用 <code>valueOf()</code> 返回枚举常量，不存在的会报错 <code>IllegalArgumentException</code></div><div><code>System.out.println(Color.valueOf("RED"));</code></div><div><code>// System.out.println(Color.valueOf("WHITE"));</code></div></div></div><div>代码输出结果为:</div><div><div>index 0</div><div>at index 1</div><div>at index 2</div></div><div><div>枚举</div><div>枚举类一样可以用自己的变量、方法和构造函数，构造函数只能使用 <code>private</code> 访问，所以外部无法调用。</div><div>枚举类可以包含具体方法，也可以包含抽象方法。如果枚举类具有抽象方法，则枚举类的所有子类都必须实现它。</div></div><div><div><code>Color</code></div><div><code>RED, GREEN, BLUE;</code></div><div><div>构造函数</div><div><code>private Color()</code></div><div><code>System.out.println("Constructor called for : " + this.toString());</code></div></div><div><code>public void colorInfo()</code></div><div><code>System.out.println("Universal Color");</code></div></div><div><div><code>class Test</code></div><div>输出</div><div><code>public static void main(String[] args)</code></div><div><code>Color c1 = Color.RED;</code></div><div><code>System.out.println(c1);</code></div><div><code>c1.colorInfo();</code></div></div><div>代码输出结果为:</div><div><div>Constructor called for : RED</div><div>Constructor called for : GREEN</div></div></div></td></div>	<div><div><div><div>}</div></div><div><div>// 使用 <code>valueOf()</code> 返回枚举常量，不存在的会报错 <code>IllegalArgumentException</code></div><div><code>System.out.println(Color.valueOf("RED"));</code></div><div><code>// System.out.println(Color.valueOf("WHITE"));</code></div></div></div><div>代码输出结果为:</div><div><div>index 0</div><div>at index 1</div><div>at index 2</div></div><div><div>枚举</div><div>枚举类一样可以用自己的变量、方法和构造函数，构造函数只能使用 <code>private</code> 访问，所以外部无法调用。</div><div>枚举类可以包含具体方法，也可以包含抽象方法。如果枚举类具有抽象方法，则枚举类的所有子类都必须实现它。</div></div><div><div><code>Color</code></div><div><code>RED, GREEN, BLUE;</code></div><div><div>构造函数</div><div><code>private Color()</code></div><div><code>System.out.println("Constructor called for : " + this.toString());</code></div></div><div><code>public void colorInfo()</code></div><div><code>System.out.println("Universal Color");</code></div></div><div><div><code>class Test</code></div><div>输出</div><div><code>public static void main(String[] args)</code></div><div><code>Color c1 = Color.RED;</code></div><div><code>System.out.println(c1);</code></div><div><code>c1.colorInfo();</code></div></div><div>代码输出结果为:</div><div><div>Constructor called for : RED</div><div>Constructor called for : GREEN</div></div></div>
--	---



反馈/建议

Java 教程

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 多态

Java 抽象类

Java 封装

Java 接口

Java 枚举

Java 包(package)

ctor called for : BLUE

sal Color

Java 接口

Java 包(package) →

1 篇笔记

写笔记

枚举类中的抽象方法实现，需要枚举类中的每个对象都对其进行实现。

```
enum Color{
    RED{
        public String getColor(){//枚举对象实现抽象方法
            return "红色";
        }
    },
    GREEN{
        public String getColor(){//枚举对象实现抽象方法
            return "绿色";
        }
    },
    BLUE{
        public String getColor(){//枚举对象实现抽象方法
            return "蓝色";
        }
    };
    public abstract String getColor();//定义抽象方法
}
```

```
public class Test{
    public static void main(String[] args) {
        for (Color c:Color.values()){
            System.out.print(c.getColor() + "、");
        }
    }
}
```

doggy 1年前 (2020-11-30)

例	字符集&工具	最新更新	站点信息
实例	· HTML 字符集设置	· Julia 数据类型	· 意见反馈
实例	· HTML ASCII 字符集	· Julia 元组	· 免责声明
cript		· Python2 与 Pyth...	· 关于我们
例			· 文章归档

Java 教程

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 多态

Java 抽象类

Java 封装

Java 接口

Java 枚举

Java 包(package)

实例

实例

实例

· HTML ISO-8859-1

· PNG/JPEG 图片压缩

· HTML 拾色器

· JSON 格式化工具

Java 枚举(enum) | 菜鸟教程

· Julia 数组

· Python3 reload(...)

· Julia 基本语法

· Julia 交互式命令

关注微信



-2022 菜鸟教程 runoob.com All Rights Reserved. 备案号：闽ICP备15012807号-1

反馈/建议