

CSDN博客 (<http://blog.csdn.net>) 移动开发 (<http://blog.csdn.net/mobile/newarticle.html>)
Web前端 (<http://blog.csdn.net/web/newarticle.html>) 架构设计 (<http://blog.csdn.net/enterprise/newarticle.html>)
编程语言 (<http://blog.csdn.net/code/newarticle.html>) 互联网 (<http://blog.csdn.net/www/newarticle.html>) 更多



写博客 (<http://write.blog.csdn.net/postedit>)

0



qq_36596145 (http://blog.csdn.net/qq_36596145) | 退出 (<https://passport.csdn.net/account/logout?ref=toolbar>)

【第二期】观点：人工智能到底用 GPU？还是用 FPGA？ (<http://blog.csdn.net/pk.html?id=9715>)

Java之泛型进阶——泛型通配符

原创 2016年03月15日 21:21:06

877

0

4

Java之泛型进阶——泛型通配符与边界

简介

主要记录泛型的边界与通配符相关，通配符上限，通配符下限。

边界

从上一篇文章中知道，为了保持向后兼容，有了泛型擦除，但是泛型的擦除绝大多数情况实际上是将类型形参用Object代替，但是有时候又想泛型擦除上限固定在一个指定的范围，即某个类或接口的子类，来允许做一些操作，就有了泛型的边界。



Oscar Chen (<http://blog....>)

+关注

(<http://blog.csdn.net/chenghuaying>)

原创

粉丝

喜欢

182

14

3

- > CentOS 集群机器之间ssh免密
(/crave_shy/article/details/72964997)
- > JVM-内存管理-运行时数据区域
(/crave_shy/article/details/56675052)
- > JVM-Blog目录
(/crave_shy/article/details/56675032)
- > JVM-为什么要学JVM
(/crave_shy/article/details/56673439)

更多文章

(<http://blog.csdn.net/chenghuaying>)

在线课程



(http://edu.csdn.net/huiyiCourse/series_detail?utm_source=blog7)

【直播】机器学习&数据挖掘7周实训--韦玮

(http://edu.csdn.net/huiyiCourse/series_detail/54?utm_source=blog7)



(http://edu.csdn.net/combo/detail/471?utm_source=blog7)

【套餐】系统集成项目管理工程师顺利通关--徐朋

(http://edu.csdn.net/combo/detail/471?utm_source=blog7)

```

package org.andy.items.thkinjava.generics.generic2016.bound;

public class GenericBound {
    public static void main(String[] args) {
        Item<GolderFish> item = new Item<>(new GolderFish());
        item.doSomething();
    }
}

interface Animal{
    void speak();
}

interface Fish{
    void bubble();
}

/**
 * 俩接口的实现类，可以符合Item类T的要求，满足其类型参数要求。
 */
class GolderFish implements Animal, Fish {
    @Override
    public void speak() {
        System.out.println("speak something...");
    }
    @Override
    public void bubble() {
        System.out.println("bubble...");
    }
}

class HoldItem<T>{
    T t;
    public HoldItem(T t) {
        this.t = t;
    }
    @SuppressWarnings("unused")
    public T getT() {
        return t;
    }
    @SuppressWarnings("unused")
    public void setT(T t) {
        this.t = t;
    }
}

//泛型的extends后如果有多个接口则使用&分开
class Item<T extends Animal & Fish> extends HoldItem<T>{
    public Item(T t) {
        super(t);
    }

    /**
     * 此时的T可以视为Animal和Fish接口的实现类，可以使用其方法。
     */
    public void doSomething() {
        super.t.speak();
        super.t.bubble();
    }
}

```

通配符相关

主要包括无界通配符，通配符上界，通配符下界。

通配符

通配符不是用来定义泛型的，而是用来代表任何一种类型实参！自己一直困在一种误区中，就是以为通配符是可以在定义泛型类、泛型方法或者泛型接口时使用的。如 `class Generic<?>{}` 这种语法是错误的。

泛型中没有逻辑上的父子关系，如 `List<Number>` 并不是 `List<Integer>` 的父类。两者擦除之后都是`List`，所以形如

```
/**
 * 两者并不是方法的重载。擦除之后都是同一方法，所以编译不会通过。
 * 擦除之后：
 *
 * void m(List numbers) {}
 * void m(List strings) {} //编译不通过，已经存在相同方法签名
 */
void m(List<Number> numbers) {

}

void m(List<String> strings) {

}
```

如果想让 `List<Number>` 逻辑上成为 `List<Integer>` 的父类（实际的应用场景中就是向方法传入的实际参数是方法声明的参数的子类），则可以使用泛型的通配符“`?`”，它表示任何一种不确定的类型。如：

```
/**
 * 可以传入泛型为任何类型的List实现类
 * 但是因为list并不知道你传入的具体会是什么类型，所以只可以使用每个元素从Object继承的方法
 */
static void genericWildcard(List<?> list) {
    list.forEach(java.lang.Object::toString);
}

public static void main(String[] args) {
    List<Number> list = new ArrayList<>();
    genericWildcard(list);

    List<Integer> list1 = new ArrayList<>();
    genericWildcard(list1);

    //还可以传入泛型为String类型
    List<String> list2 = new ArrayList<>();
    genericWildcard(list2);
}
}
```

通配符边界

有时候希望传入的类类型有一个指定的范围，从而可以进行一些允许的操作，这时候就是通配符边界登场的时候了。泛型的边界分两种：上界和下界。

对于通配符的部分，可以从三方面理解（以`List`为例，方便理解）：

含义

查询

与泛型有关的操作

先看三个很简单的类：

```

public abstract class Animal {

    public abstract void animalMethod();

    @Override
    public String toString() {
        return "Animal";
    }
}

public class Dog extends Animal {
    @Override
    public void animalMethod() {
        System.out.println("DOG method");
    }

    @Override
    public String toString() {
        return "Dog";
    }
}

public class Fish extends Animal {
    @Override
    public void animalMethod() {
        System.out.println("Fish method");
    }

    @Override
    public String toString() {
        return "Fish";
    }
}

```

Animal是一个抽象类，Fish，Dog是其实现。

通配符上界

extends关键字声明了类型的上界，表示参数化的类型可能是所指定的类型，或者是此类型的子类。

示例：

```

//list中所有的元素都是Animal或者是其子类
List<? extends Animal> list;

-----

/**
 * List<? extends Animal> animals;
 * 含义: 表示animals集合中所有的元素都是animal或者是animal的子类。
 * 查询: 如方法体中所写，因为animals中所有元素都是其子类，所以可以调用其子类从animal中变
 * 增加: 见方法体
 */
public static void genericUpperWildcard(List<? extends Animal> animals){
    animals.forEach(Animal::animalMethod);

    /*
     * 下面两行都是编译错误，如果了解前面文章中提到的擦除，则很好理解。
     * 编译后的方法签名参数list所用的泛型类会被转换成Animal。假如此时向list中添加的
     * 肯定是不存在的。所以出于类型安全问题，不允许向含有通配符下界的泛型类中添加元
     */
    // animals.add(new Dog());
    // animals.add(new Animal());
}

public static void main(String[] args) {
    List<Dog> dogs = new ArrayList<>();
    dogs.add(new Dog());
    genericUpperWildcard(dogs);
}

```

通配符的下界

super关键字声明了类型的下界，表示参数化的类型可能是所指定的类型，或者是此类型的父类型，直至Object。

```
/**
 * List<? super Animal> list;
 * 含义：表示list中所有的元素都是Animal类或者是其父类
 * 查询：因为返回的结果并不能保证是那个当初添加的Animal类或则其父类，返回的查询结果的类
 * 增加：可以向list中添加任何Animal实例或者Animal子类，因为list在编译之后泛型先被擦除然
 * 此时向里添加的任何类型都是Animal类，所以调用Animal中的任何非私有方法都是允许的。
 *
 */

public static void genericLowerWildcard(List<? super Animal> list) {
    list.add(new Animal());
    list.add(new Dog());

    //      Animal animal1 = list.get(0); 编译出错，
    Object animal = list.get(0);
}
```


泛型的通配符，尤其是边界，比较难理解，但是明白其原理之后很多问题就迎刃而解了。了解泛型的擦除，以及编译之后泛型转换成的普通Java代码是什么样的很重要。

版权声明：本文为博主原创文章，未经博主允许不得转载。



标签：java (<http://so.csdn.net/so/search/s.do?q=java&t=blog>) /
泛型 (<http://so.csdn.net/so/search/s.do?q=泛型&t=blog>) /
通配符 (<http://so.csdn.net/so/search/s.do?q=通配符&t=blog>) /
通配符上界 (<http://so.csdn.net/so/search/s.do?q=通配符上界&t=blog>) /
通配符下界 (<http://so.csdn.net/so/search/s.do?q=通配符下界&t=blog>) /

0条评论

 qq_36596145 (http://my.csdn.net/qq_36596145)
(http://my.csdn.net/qq_36596145)



发表评论

暂无评论

相关文章推荐

Java 泛型通配符？解惑 (/baple/article/details/25056169)

一、通配符的上界 既然知道List并不是List的子类型，那就需要去寻找替他解决的办法，是AnimalTrianer.act()方法变得更为通用（既可以接受List类型，也可以接受List等参...



Baple 2014-05-05 15:53 26626

Java泛型-- 通配符 (/flfna/article/details/6576394)

通配符在本文的前面的部分里已经说过了泛型类型的子类型的不相关性。但有些时候，我们希望能够像使用普通类

型那样使用泛型类型：◆ 向上造型一个泛型对象的引用◆ 向下造型一个泛型对象的引用向上造型一个泛型对象...



flfna 2011-06-30 09:46 19248

AlarmManager研究 (/thinkinwm/article/details/40119033)

<http://my.oschina.net/youranhongcha/blog/149564> 目录[-] 1.概述 2.AlarmManager 2.1 ...



thinkinwm 2014-10-15 21:41 657

C语言的基本文件操作 (/lan74_/article/details/53981296)

C语言的基本文件操作序列1：基本的txt文件读入读出来段代码看看#include int main() { char a[] = "hellow\n"; char buf[6]; ...



LAN74_ 2017-01-02 18:56 104

MBProgressHUD的简单封装 (/anywhereIOS/article/details/50813485)

下面是依据MBProgressHUD提供的分类方法：/** * 显示一般信息 */ + (void)show:(NSString *)text icon:(NSString *)icon vi...



anywhereIOS 2016-03-06 15:09 1414

《深入理解java虚拟机》学习笔记9——并发编程（一） (/chjttony/article/details/7982231)

随着多核CPU的高速发展，为了充分利用硬件的计算资源，操作系统的并发多任务功能正变得越来越重要，但是CPU在进行计算时，还需要从内存读取输出，并将计算结果存放到内存中，然而由于CPU的运算速度比内存高...



chjttony 2012-09-15 15:11 3251

think in java interview-高级开发人员面试宝典(二) (/lifetragedy/article/details/9751079)

从现在开始，以样题的方式——例出各种面试题以及点评，考虑到我在前文中说的，对于一些大型的外资型公司，你将会面临全程英语面试，因此我在文章中也会出现许多全英语样题。这些题目来自于各个真实的公司，公司名我...



lifetragedy 2013-08-05 00:43 38487

就 3 点，提升工作效率 (/foruok/article/details/76950020)

要想提高工作效率，不论你看什么书，看什么文章，用什么工具，只有下面这三点最重要



foruok 2017-08-09 09:10 102006

计算机视觉、机器学习相关领域论文和源代码大集合--持续更新..... (/yingchunhua365/article/details/14043609)

计算机视觉、机器学习相关领域论文和源代码大集合--持续更新..... zouxy09@qq.com
<http://blog.csdn.net/zouxy09> 注：下面有project网站的大部分都有pap...



diaoguangqiang 2013-12-03 10:41 608

图像拼接-硬拼接 (/gukewee/article/details/53606575)

功能：两幅图A和B，将A的左侧和B的右侧按照6个经验offset参数拼接在一起； 算法库：halcon； 注：算法前提是比较精准的机械精度；



gukewee 2016-12-13 09:54 507

java再复习——泛型的通配符与扩展 (/sinat_31311947/article/details/59111116)

上次说了泛型的语法，发现有个问题，就是泛型是什么就得是什么，但这样岂不是程序就得写死了，没有一点可预判断性了吗，那么泛型的好处也就光体现在不用进行强制类型转换上了吗？泛型还给我们提供了一种符号：...



sinat_31311947 2017-03-01 17:07 96

Java程序员从笨鸟到菜鸟之（十）枚举，泛型详解 (http://qcycom.iteye.com/blog/1489841)

一：首先从枚举开始说起 <p style="font-family: Arial; font-size: 14px



qcycom 2012-04-19 22:48 622

Java基础之——泛型(二) 通配符 (/justxiaosha/article/details/51123569)

说在前头的话如果迷茫，请做好当下的事！上一篇讲了泛型的基本知识如果你没用看过，请参考Java基础之——泛型(一)这篇我们来讲一下泛型高级之通配符吧！一、通配符是个什么玩意...



justXiaoSha 2016-04-11 17:34 106

Java泛型通配符 extends 与 super (http://gcc2ge.iteye.com/blog/2267225)

Java 泛型 关键字说明？通配符类型 <? extends T> 表示类型的上界，表示参数化类型的可能是T 或是 T的子类 <? super T> 表示类型下界（Java Core中叫超类型限定），表示参数化类型是此类型的超类型（父类型），直至Object



gcc2ge 2015-12-28 16:57 63

java泛型 (http://huanglz19871030.iteye.com/blog/1127033)

在Java SE1.5中，增加了一个新的特性：泛型（日语中的总称型）。何谓泛型呢？通俗的说，就是泛泛的指定对象所操作的类型，而不像常规方式一样使用某种固定的类型去指定。<span style="color:



huanglz19871030 2011-07-18 15:42 394

Java泛型——通配符和Object的区别 (/uusad/article/details/7898358)

通配符(?)和Object是有区别的：void function(List){} void function(List){} function(new List()); //对于第一种方法编译错误...



uusad 2012-08-23 09:39 5721

java中泛型学习1之类型通配符(?) (http://wsq.iteye.com/blog/1824357)

实体类 package cn.xy.model; /** * 生物类 * @author xy */ public class Living { private String na



wsq 2012-10-17 20:27 243

Java基础进阶_day07_(泛型,Collection集合,迭代器,增强for循环) (/l631106040120/article/details/69055820)



【转】java 泛型 通配符 (<http://roomfourteen224.iteye.com/blog/2193764>)

通配符 在本文的前面的部分里已经说过了泛型类型的子类型的不相关性。但有些时候,我们希望能够像使用普通类型那样使用泛型类型: <p style="col



java泛型程序设计——无限定通配符+通配符捕获 ([/pacosonswjtu/article/details/50224335](http://pacosonswjtu/article/details/50224335))

【0】README0.1) 本文描述+源代码均 转自 core java volume 1 , 旨在理解 java泛型程序设计 的 无限定通配符+通配符捕获 的相关知识; 【1】无限定通配符相关1.1) 无...



Java 理论与实践: 使用通配符简化泛型使用 (<http://fengyanzhang.iteye.com/blog/1944366>)

自从泛型被添加到 JDK 5 语言以来,它一直都是一个颇具争议的话题。一部分人认为泛型简化了编程,扩展了类型系统从而使编译器能够检验类型安全;另外一些人认为泛型添加了很多不必要的复杂性。对于泛型我们都经历过一些痛苦的回忆,但毫无疑问通配符是最棘手的部分。通配符基本介绍 泛型是一种表示类或方法行为对于未知类型的类型约束的方法,比如 “不管这个方法的参数 x 和 y 是哪种类型,它们必须是相同的类型”, “必须为这些方法提供同一类型的参数” 或者 “foo() 的返回值和 bar() 的参数是同一类型的”。通配符 — 使用一个奇怪的问号表示类型参数 — 是一种表示未知类型的类型约束的方法



java进阶（五）：Java泛型 ([/u012228718/article/details/39271577](http://u012228718/article/details/39271577))

一、泛型简介



Java 泛型 通配符理解 (<http://hacksin.iteye.com/blog/2147558>)

泛型, 通配符 (1) 传参时要求类型为B, 那么可以传入B或B的子类型 获取返回值时, 如果返回的类型是B, 那么接收他的类型可以是B或者B的父类型。 (2) 表示其定义的泛型范围为T及其子类型 表示其定义的泛型范围为T及其父类型 (3) object是所有类的父类, NULL代表所有类型 (4) 举例: 设: B extends A, C extends B List list = new ArrayList();//表示list存放的是B或者B的子类。 //list.add(new A()); //



黑马程序员——JAVA基础之泛型和通配符 ([/run_wind/article/details/41620661](http://run_wind/article/details/41620661))



----- android培训、java培训、期待与您交流! ----- 泛型: JDK1.5版本以后出现新特性。用于解决安全问题, 是一个类型安全机制。 ...



Java泛型中通配符的几点理解（转） (<http://bioubiou.iteye.com/blog/1881577>)



置换原则 结合Java本身的一些面向对象的特性, 我们很容易理解这么一个置换原则: 一个指定类型的变量可以被赋值为该类型的任何子类; 一个指定某种类型参数的方法可以通过传入该类型的子类来进行调用。 总的来说, 就

是说我们使用的任何类型变量都可以用该类型的子类型来替换。泛型中一种错误的继承关系 在泛型的编程中，我们考虑到子类型关系的时候，容易把一种关系给弄混淆，并错误的采用置换原则。比如说：List ints = new ArrayList(); ints.add(1); ints.add(2)

 wxb880114 2013-06-03 15:27  401



Java泛型——类型通配符<?> 与 类型通配符上限<? extends Number> (/zhanghaor/article/details/51895306)

原文链接：http://www.cnblogs.com/lwbqqyumidi/p/3837629.html 一. 泛型概念的提出（为什么需要泛型）？首先，我们看下下面这段简短的代码：...

 zhanghaor 2016-07-13 10:14  853



Java泛型-- 通配符（转载） (http://langgufu.iteye.com/blog/2190212)

通配符 在本文的前面的部分里已经说过了泛型类型的子类型的不相关性。但有些时候，我们希望能够像使用普通类型那样使用泛型类型：
• 向上造型一个泛型对象的引用
• 向下造型一个泛型对象的引用
向上造型一个泛型对象的引用

 langgufu 2015-03-06 17:33  1362




夯实JAVA基本之一——泛型详解(2): 高级进阶 (/xuepeng0728119/article/details/50705014)

一、类型绑定 1、引入 我们重新看上篇写的一个泛型：[java] view plain copy class Point { ...

 xuepeng0728119 2016-02-20 15:59  105




java 泛型以及通配符 (http://ywzqzhangjiawei.iteye.com/blog/549043)

泛型（目标是类型安全性，主要用于集合）：参数化类型：[code="java"]import java.util.ArrayList; import java.util.Iterator; import java.util.List; public class TestDemo { public static void main(String[] args) { testList(); testGenList(); } public static void testList(){ List list = new ArrayList(

 ywzqzhangjiawei 2009-12-15 16:29  753
 目录

夯实JAVA基本之二 —— 反射（2）：泛型相关周边信息获取 (/harvic880925/article/details/50085595)

前言：坚信自己坚信的，坚持自己坚持的，永远选择相信自己。在上篇中，我们简单给大家讲解了如何利用反射来获取普通类型的类的使用，今天给大家讲解下，有关如何使用反射来获取泛型中的信息。提前提醒，本篇文章内...

 harvic880925 2015-11-28 17:24  3755
 分享

java 泛型通配符 (http://zhphappy.iteye.com/blog/1947135)

泛型是Java SE 1.5的新特性，泛型的本质是参数化类型，也就是说所操作的数据类型被指定为一个参数。这种参数类型可以用在类、接口和方法的创建中，分别称为泛型类、泛型接口、泛型方法。泛型的好处是在编译的时候检查类型安全，并且所有的强制转换都是自动和隐式的，提高代码的重用率。其中通配符“?”表示未知类型，可以是任何引用类型。通配符一般会结合extend或super使用，来约束泛型的类型上界和类型下界。<

 zhphappy 2013-09-25 23:25  390

夯实JAVA基本之一——泛型详解(2): 高级进阶 (/harvic880925/article/details/49883589)

前言：被温水煮惯了，梦想的东西总是不敢去尝试，失败了又怎样，最多从头来过。上一篇给大家初步讲解了泛型变量的各种应用环境，这篇将更深入的讲解一下有关类型绑定，通配符方面的知识。一、类型绑定1、引入我们重...



harvic880925 2015-11-17 10:24 7127

java泛型之通配符的使用 (<http://happyprince.iteye.com/blog/2256372>)

转自：<http://blog.csdn.net/lonelyroamer/article/details/7927212> 通配符有三种：
<p style="font-family: Arial; line-height:



美丽的小岛 2015-11-12 12:15 122

Java泛型进阶 (/carmelo_z/article/details/71189639)

原文链接 1.在泛型代码内部，无法获得任何有关泛型参数类型的信息。“在泛型代码内部，无法获得任何有关泛型参数类型的信息”，这并不是说使用泛型代码时给定的类型参数会消失（要不然泛型就没有存在的意...



Carmelo_Z 2017-05-04 22:46 84

Java泛型通配符extends与super (<http://sharewind.iteye.com/blog/1622164>)

Java 泛型 关键字说明？通配符类型 <? extends T> 表示类型的上界，表示参数化类型的可能是T 或是 T的子类 <? super T> 表示类型下界（Java Core中叫超类型限定），表示参数化类型是此类型的超类型（父类型），直至Object extends 示例 static class Food{} static class Fruit extends Foo



sharewind 2012-08-06 11:47 25884

黑马程序员——Java泛型通配符的总结 (</tbmacb/article/details/38761143>)

----- android培训、java培训、期待与您交流！ -----



tbmacb 2014-08-22 20:07 160

java进阶之路 (<http://m635674608.iteye.com/blog/2232516>)

啥也不说了，都在图里了。希望可以给大家的职业规划一些提示，尤其是写了几年程序，却越来越迷茫的同学。
泛型: 泛型指代了参数的类型化类型, 一般被用在接口, 类, 方法中 >作用: 用来确定参数的范围, 在书写代码的时候提前检查代码的错误性 >泛型的声明, 以下给出类声明, 依此类推: class Class...



qq285016127 2014-10-14 18:04 741

java泛型的通配符的上边界和下边界 (<http://gaoquanyang.iteye.com/blog/1518466>)

限定通配符的上边界： `Vector<? extends Number> x = new Vector<Integer>();` 说明他能存放 `Number`或者是`Number`的子类。 限定通配符的下边界： `Vector<? super Integer> x = new Vector<Number>();` 说明它能存放的是`Integer`或者`Integer`的超类. 注意： 限定通配符总是包括自己。



bsszds 2012-05-09 09:35 505

JAVA高新技术——JDK1.5新特性 (</wknIm0001/article/details/20297759>)

静态导入 一般导入import是导入一个类或某个包中的所有类。而静态导入是import static 导入的类名或者类名的某个静态方法。其作用是简化书写，调用静态方法时不用再指定包名了，前提是方法名...



wknIm0001 2014-03-02 19:20 345

Java 泛型之通配符 (<http://zds420.iteye.com/blog/1107568>)

```
package com.study.generics; //介绍泛型的通配符 ? 怎么使用
class GenericsCommon<T> { private T variable;
public T getVariable() { return variable; }
public void setVariable(T variable) { this.variable = variable; }
} //泛型中重写toString方法
public Strin
```



zds420 2011-06-28 14:02 518
