

java.util.jar

Attributes.Name

java.lang.Object

Attributes.Name

语法

```
public static class Attributes.Name
```

描述

一个java档案(JAR)文件包含了一个名单文件（见 Manifest），在名单文件中又包含了许多属性。一个属性是一个名字/值对，其中名字和值都是字符串。尽管值可以使任意非空字符串，但是属性名是不区分大小写的，并要遵循一定的语法规则。特别是一个属性名可以是一个任意长度的非空字符串，但是字符串只能由以下的字符组成：

数字：0~9

字母：a~z及A~Z

下划线：_

减号：-

例如“Version_1”和“0-1”是有效的属性名，而“a.b”和“#1”是无效的属性名。

对属性名的语法规则的详细描述请到下列网址中查询：

<http://java.sun.com/products/jdk/1.2/docs/guide/jar/manifest.html>。

使用

Attributes.Name类代表了一个属性名。在与一个Attributes类进行连接时必须要用到这个类。一个Attributes对象实际上是一个关键字为Attributes.Named对象的映射表（见java.util.Map）。为了更新一个Attributes对象的值，或为了从一个Attributes对象中检索值，可以使用一个Attributes.Name对象来指定一个属性名。

成员概述

构造函数

Attributes.Name()

构造一个新的Attributes.Name实例。

对象方法

equals()

判断Attributes.Name对象是否与另一对象相等。

hashCode()

计算Attributes.Name对象的哈希码。

toString()

返回属性名。

预定义属性名

CLASS_PATH

用来指定附加的JAR文件的属性名。

CONTENT_TYPE

用来指定JAR文件的入口类型的属性名。

(续)

成员概述	
MAIN_CLASS	用来指定一个应用程序的入口点的属性名。
IMPLEMENTATION_TITLE	包含了一个包实现标题的属性名。
IMPLEMENTATION_VERSION	包含了一个包实现版本的属性名。
IMPLEMENTATION_VENDOR	包含了一个包实现的提供商的属性名。
MANIFEST_VERSION	用来指定名单版本的属性名。
SEALED	用来封缄一个包的属性名字。
SIGNATURE_VERSION	包含了JAR文件的签名版本好的属性名。
SPECIFICATION_TITLE	包含了一个包的指定标题的属性名。
SPECIFICATION_VERSION	包含了一个包的指定版本号的属性名。
SPECIFICATION_VENDOR	包含了一个包的指定提供商的属性名。

参见

Attributes、Manifest。

示例

参见Attributes类中对此类的使用示例。

Attributes.Name()

目的	构造一个新的Attributes.Name类的实例。
语法	public Name(String aname)
描述	此构造函数创建一个新的包含了 aname的Attributes.Name对象。aname可以是一个任意长度的非空字符串，但是字符串只能由以下的字符组成： 数字：0 ~ 9 字母：a ~ z及A ~ Z 下划线：_ 减号：-
参数	
aname	非空字符串。
异常	
IllegalArgumentException	如果aname是无效的。
示例	

```
Attributes.Name an;

// Some valid attribute names.
an = new Attributes.Name("Version");
an = new Attributes.Name("_-0123456789");

// Some invalid attribute names.
//an = new Attributes.Name("a.b");           // IllegalArgumentException
//an = new Attributes.Name("a b");           // IllegalArgumentException
```

CLASS_PATH

目的	用来指定附加JAR文件的属性的名字。
----	--------------------

语法	<code>public static final Name CLASS_PATH</code>
描述	<p>此常量包含了用来指定附加 JAR 文件的属性的名字。如果一个 JAR 文件的内容依赖于另一个 JAR 文件，那么可以用 CLASS_PATH 属性来给附加的 JAR 文件命名。当一个类加载器搜索一个类或资源时，它也会搜索这些 JAR 文件。</p> <p>CLASS_PATH 属性的值是一个被空格分开的若干个 URL 的集合。每个 URL 必须引用一个 JAR 文件。URL 可以是绝对的也可以是相对的。如果是相对的，那么它是与包含了 CLASS_PATH 属性的 JAR 文件有关的。在一个装载表文件的主属性集合（见 Manifest）中必须指定 CLASS_PATH 属性。</p> <p>此常量的值为 “ Class-Path ”。</p>
示例	见 Package.isSealed()。

CONTENT_TYPE

目的	用来指定 JAR 文件入口类型属性的名字。
语法	<code>public static final Name CONTENT_TYPE</code>
描述	<p>此常量包含了用来指定 JAR 文件入口类型属性的名字。CONTENT_TYPE 属性的值是一个 MIME 类型（RFC 2046）。例如 “ image/gif ” 和 “ audio/basic ” 便是两个 MIME 类型的值。这个信息可被一个浏览器用来决定如何显示 JAR 的入口。关于 MIME 类型请参阅 MimeType。</p> <p>CONTENT_TYPE 属性可被指定为一个主属性。这意味着其值适用于 JAR 文件中的所有入口。因此它可以被各个单独的入口重载。</p> <p>此常量的值为 “ Content-Type ”。</p>

equals()

目的	判断 Attributes.Name 对象是否与另一个对象相等。
语法	<code>public boolean equals(Object obj)</code>
描述	此方法判断 Attributes.Name 对象是否与 obj 相等。只有当 obj 是一个 Attributes.Name 类的非空实例并且 toString().equalsIgnoreCase(obj.toString()) 方法返回 true 时，此方法才返回 true。
参数	
obj	可能是 null 的一个对象。
返回	如果 obj 是一非空对象且与 Attributes.Name 对象相等，则此方法返回 true。
重载	java.lang.Object.equals()。
参见	hashCode()。
示例	

```
Attributes.Name an1 = new Attributes.Name("version");
Attributes.Name an2 = new Attributes.Name("VERSION");

// Show that attribute names are not case-sensitive.
System.out.println(an1.equals(an2));           // true
System.out.println(an1.hashCode());            // 351608024
System.out.println(an2.hashCode());            // 351608024

System.out.println(an1.toString());             // version
System.out.println(an2.toString());             // VERSION
```

hashCode()

目的	计算Attributes.Name对象的哈希码。
语法	public int hashCode()
描述	此方法计算 Attributes.Name对象的哈希码。两个相等的 Attributes.Name对象（见 equals()）有相同的哈希码。两个不同的 Attributes.Name对象也有可能具有相同的哈希码，尽管哈希码的计算算法使得这种可能性非常小。哈希码一般用来作为哈希表的关键字。
返回	返回Attributes.Name对象的哈希码。
重载	java.lang.Object.hashCode()。
参见	equals()、java.lang.Object.equals()。
示例	见equals()。

IMPLEMENTATION_TITLE

目的	包含了一个包的实现标题的属性的名字。
语法	public static final Name IMPLEMENTATION_TITLE
描述	<p>此常量包含了一个包的实现标题的属性的名字。关于 IMPLEMENTATION_TITLE属性的值请参阅Package.getImplementationTitle()。</p> <p>为了把一个IMPLEMENTATION_TITLE用于一个包p.q，例如，把它添加到一个名为“p/q/”的装载表表项上，具体做法如下：</p> <p>Name：p/q/</p> <p>Implementation-Title：Demo API Implementation for Solaris</p> <p>在这个示例中，IMPLEMENTATION_TITLE属性的值为“Demo API Implementation for Solaris”。</p> <p>IMPLEMENTATION_TITLE属性可以被指定为一个主属性。这意味着它的值适用于JAR文件中的所有包。因此它可以被各个单独的包重载。</p> <p>此常量的值为“Implementation-Title”。</p>
参见	java.lang.Package.getImplementationTitle()。
示例	见java.lang.Package类的示例。

IMPLEMENTATION_VERSION

目的	包含了一个包的实现版本号的属性的名字。
语法	public static final Name IMPLEMENTATION_VERSION
描述	<p>此常量包含了一个包的实现版本号的属性的名字。</p> <p>关于IMPLEMENTATION_VERSION属性的值请参阅Package.getImplementationVersion()。</p> <p>为了把一个IMPLEMENTATION_VERSION用于一个包p.q，例如，把它添加到一个名为“p/q/”的名单表项上，具体做法如下：</p> <p>Name：p/q/</p> <p>Implementation-Version：1.2.3</p> <p>在这个示例中，IMPLEMENTATION_VERSION属性的值为“1.2.3”。</p> <p>IMPLEMENTATION_VERSION属性可以被指定为一个主属性。这意味着它的值</p>

适用于JAR文件中的所有包。因此它可以被各个单独的包重载。

此常量的值为“Implementation-Version”。

参见 java.lang.Package.getImplementationVersion()。

示例 见java.lang.Package类的示例。

IMPLEMENTATION_VENDOR

目的 包含了一个包的实现提供商的属性的名字。

语法 public static final Name IMPLEMENTATION_VENDOR

描述 此常量包含了一个包的实现提供商的属性的名字。

关于IMPLEMENTATION_VENDOR属性的值请参阅

Package.getImplementationVendor()。

为了把一个IMPLEMENTATION_VENDOR用于一个包p.q，例如，把它添加到一个名为“p/q/”的装载表表项上，具体做法如下：

Name：p/q/

Implementation-Vendor：Chan，Lee，Kramer

在这个示例中，IMPLEMENTATION_VENDOR属性的值为“Chan，Lee，Kramer”。

IMPLEMENTATION_VENDOR属性可以被指定为一个主属性。这意味着它的值适用于JAR文件中的所有包。因此它可以被各个单独的包重载。

此常量的值为“Implementation-Vendor”。

参见 java.lang.Package.getImplementationVendor()。

示例 见java.lang.Package类的示例。

MAIN_CLASS

目的 用来指定一个应用程序入口点的属性的名字。

语法 public static final Name MAIN_CLASS

描述 此常量包含了用来指定一个应用程序入口点的属性的名字。MAIN_CLASS属性的值必须是一个完整的类名（例如“java.lang.String”）。在一个装载表文件的主属性集合中必须包含有MAIN_CLASS属性。被命名的类必须是public类型，并且包含在JAR文件中，而且还应当有一个带有签名的方法：

public static void main(String[] args)

此常量的值为“Main-Class”。

示例 此例演示了这个属性的使用方法。它首先定义一个装载表文件并指定其主类为p.Main。然后，它利用装载表文件创建了一个JAR文件。最后，用java -jar命令激活JAR文件。

p/Main.java

```
package p;
class Main {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

manifest.mf

```
Manifest-Version: 1.0
Main-Class: p.Main
```

输出

```
> jar cfm main.jar manifest.mf p/Main.class
> java -jar main.jar
Hello World
```

MANIFEST_VERSION

目的	用来指定名单文件版本号的属性的名字。
语法	public static final Name MANIFEST_VERSION
描述	此常量包含了用来指定装载表文件版本号的属性的名字。 MANIFEST_VERSION属性是所有装载表文件都必需的主属性（见 Manifest）。目前，最新的版本是1.0。 此常量的值为“Manifest-Version”。
示例	见MAIN_CLASS。

SEALED

目的	用来封缄一个包的属性的名字。
语法	public static final Name SEALED
描述	此常量包含了用来封缄一个包的属性的名字。（关于封缄包，请参阅Package。） SEALED属性的值可以是“true”或“false”。例如，为了封缄一个包p.q，需要把SEALED属性添加到一个名为“p/q/”的装载表表项上： Name: p/q/ Sealed: true SEALED属性可以被指定为一个主属性。这意味着它的值适用于JAR文件中的所有包。因此它可以被各个单独的包重载。 此常量的值为“Sealed”。
参见	java.lang.Package.isSealed()。
示例	见Package.isSealed()。

SIGNATURE_VERSION

目的	包含JAR文件签名版本号的属性的名字。
语法	public static final Name SIGNATURE_VERSION
描述	此常量包含了包含JAR文件签名版本号的属性的名字。SIGNATURE_VERSION属性在一个签名装载表文件中是一个必需的主属性。签名装载表文件通常由签署JAR文件的工具自动产生。关于此属性更详细的资料，请查阅以下网址中的内容： http://java.sun.com/products/jdk/1.2/docs/guide/jar/manifest.html 此常量的值为“Signature_Version”。

SPECIFICATION_TITLE

目的	包含一个包的指定标题的属性的名字。
语法	public static final Name SPECIFICATION_TITLE

描述	<p>此常量包含了包含一个包的指定标题的属性的名字。</p> <p>关于SPECIFICATION_TITLE 属性的值请参阅 Package.getSpecificationTitle()。</p> <p>为了把一个SPECIFICATION_TITLE用于一个包p.q，例如，把它添加到一个名为“ p/q/ ”的装载表表项上，具体做法如下：</p> <p>Name：p/q/</p> <p>Specification-Title：Demo API Specification</p> <p>在这个示例中，SPECIFICATION_TITLE属性的值为“ Demo API Specification ”。</p> <p>SPECIFICATION_TITLE属性可以被指定为一个主属性。这意味着它的值适用于JAR文件中的所有包。因此它可以被各个单独的包重载。</p> <p>此常量的值为“ Specification-Title ”。</p>
参见	java.lang.Package.getSpecificationTitle()。
示例	见java.lang.Package类的示例。

SPECIFICATION_VERSION

目的	包含一个包的指定版本号的属性的名字。
语法	public static final Name SPECIFICATION_VERSION
描述	<p>此常量包含了包含一个包的指定版本号的属性的名字。</p> <p>关于SPECIFICATION_VERSION属性的值请参阅 Package.getSpecificationVersion()。</p> <p>为了把一个SPECIFICATION_VERSION用于一个包p.q，例如，把它添加到一个名为“ p/q/ ”的装载表表项上，具体做法如下：</p> <p>Name：p/q/</p> <p>Specification-Version：1.2.3</p> <p>在这个示例中，SPECIFICATION_VERSION属性的值为“ 1.2.3 ”。</p> <p>SPECIFICATION_VERSION属性可以被指定为一个主属性。这意味着它的值适用于JAR文件中的所有包。因此它可以被各个单独的包重载。</p> <p>此常量的值为“ Specification-Version ”。</p>
参见	java.lang.Package.getSpecificationVersion()。
示例	见java.lang.Package类的示例。

SPECIFICATION_VENDOR

目的	包含一个包的指定提供商的属性的名字。
语法	public static final Name SPECIFICATION_VENDOR
描述	<p>此常量包含了包含一个包的指定提供商的属性的名字。</p> <p>关于SPECIFICATION_VENDOR属性的值请参阅 Package.getSpecificationVendor()。</p> <p>为了把一个SPECIFICATION_VENDOR用于一个包p.q，例如，把它添加到一个名为“ p/q/ ”的装载表表项上，具体做法如下：</p> <p>Name：p/q/</p> <p>Specification-Vendor：Chan，Lee，Kramer</p>

在这个示例中， `SPECIFICATION_VENDOR` 属性的值为 “ Chan , Lee , Kramer ”。

`SPECIFICATION_VENDOR` 属性可以被指定为一个主属性。这意味着它的值适用于JAR文件中的所有包。因此它可以被各个单独的包重载。

此常量的值为 “ Specification-Vendor ”。

参见 `java.lang.Package.getSpecificationVendor ()`。

示例 见 `java.lang.Package` 类的示例。

toString()

目的 返回属性名。

语法 `public String toString()`

描述 此方法返回 `Attributes.Name` 对象的字符串代表。返回的字符串与提供给构造函数的字符串是一样的。

返回 非空属性名。

重载 `java.lang.Object.toString()`。

示例 见 `equals()`。