Socket编程

Java为我们提供了一组功能强大的类，方便我们使用Socket进行网络程序的开发。

1. InetAddress类

InetAddress类是Java包装用来表示IP地址的高级表示。几乎所有的Java网络相关的类都和它有关系，例如：serversocket,socket,URL,DataGramSocket,DataGRamPacket等。

下面就InetAddress所提供的一些类做一个简单的介绍。

1.InetAddress.getByName()

通过主机名获取对应的ip地址：（当然也支持通过IP地址来查找主机名）

该方法会建立与本地DNS的一个连接，来获取名称和地址之间的映射关系。（如果本地DNS缓存了地址，就不再获取连接。）

2\*\*.InetAddress.getLocalHost()\*\*

得到真正的主机名和IP地址。如果获取失败，则会返回一个会送地址，即主机名为“localhost”,以及一个点四分段的ip：‘127.0.0.1’ 。

3\*\*缓存\*\*

由于DNS查找的开销很大，并且如果一个地址不可达，可能会需要几秒钟的时间来确认。InetAddress提供了缓存机制，一旦得到一个给定主机的地址，就会缓存起来，即使重新定义一个InetAddress类，也不会重新再获取一次，而是直接从缓存中获取。但是需要注意一种情况。就是DNS在传输过程中可能的超时。当第一次请求是发生超时失败，而第二次的时候请求已经到达，并缓存了起来。所有Java在缓存不可达的地址时，只缓存了10s。

4.安全性问题

从主机名创建一个InetAddress对象，被认为是一种不安全的操作。因为这需要一个DNS查找，在默认的安全管理控制器下的不可行applet只允许获取它本机的主机地址以及ip。不允许不可行代码由任何其他主机名创建InetAddress对象。不可信的代码不允许对第三方主机完成任意查找。

5.获取方法

public String getHostName()

publlic String getCanonicalHostName()

public byte[] getAddress()

public String getHostAddress()

没有对应的setHostName()以及getHostName()。代表着无法通过反射来改变其内部的。因此是线程安全的。

6.地址类型

ip地址分为不同的类型，例如：‘’127.0.0.1‘’属于本地回送地址。Java提供了10中方法来判断一个ip地址属于那种不同的分段。

public boolean isAnyLocalAddress()

# \* getCanonicalHostName方法和getHostName方法的区别

getCanonicalHostName方法和getHostName方法一样，也是得到远程主机的域名。但它们有一个区别。getCanonicalHostName得到的是主机名，而getHostName得到的主机别名。getCanonicalHostName的定义如下：  
public String getCanonicalHostName()  
在访问某些域名时，getCanonicalHostName方法和getHostName方法的返回值是一样的，这和[DNS服务器](https://www.baidu.com/s?wd=DNS%E6%9C%8D%E5%8A%A1%E5%99%A8&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)如何解释主机名和主机别名以及它们的设置有关。如通过[www.ibm.com](http://www.ibm.com/)创建InetAddress对象后，使用getCanonicalHostName方法和getHostName方法返回的结果都是[www.ibm.com](http://www.ibm.com/)（有时直接返回IP地址，这可能和IBM的[DNS服务器](https://www.baidu.com/s?wd=DNS%E6%9C%8D%E5%8A%A1%E5%99%A8&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "_blank)的处理机制有关）。如果DNS不允许通过IP地址得到域名，那么这两个方法就会返回IP地址来代替域名。getCanonicalHostName方法可以分三种情况来讨论：  
  
1. 使用getLocalHost创建InetAddress对象  
在这种情况下getCanonicalHostName方法和getHostName方法得到的都是本机名。  
2. 使用域名创建InetAddress对象  
在这种情况下，getCanonicalHostName方法是否要访问[DNS服务器](https://www.baidu.com/s?wd=DNS%E6%9C%8D%E5%8A%A1%E5%99%A8&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)，取决于DNS服务器如何解释主机名和主机别名。也就是说，是否在创建InetAddress对象时就将主机名和主机别名都确定了。在前面已经讲过，使用域名创建InetAddress对象后，调用getHostName方法不会访问DNS服务器。但getCanonicalHostName方法就不一定了。这和DNS服务器的设置有关。如[www.126.com](http://www.126.com/)就需要访问DNS服务器，而[www.ibm.com](http://www.ibm.com/)就不需要访问DNS服务器。  
3．使用IP地址创建InetAddress对象  
在这种情况下，getCanonicalHostName方法和getHostName方法是完全一样的，也就是说，它们得到的都是主机名，而不是主机别名。  
之所以要使用主机别名，是因为有时主机名可能比较复杂，如Oracle官方网站的主机名bigip-otn-portal.oracle.com，因此，为了使用户访问网站更方便，就增加了更简单的主机别名，如[www.oracle.com](http://www.oracle.com/)。一个主机名可能对应多个主机别名，如oracle.com也是Oracle的主机别名。在IE的地址栏中输入http:// bigip-otn-portal.oracle.com和[http://oracle.com](http://oracle.com/" \t "_blank)都可以访问Oracle官方网站。但我们发现，有很多网站通过主机名无法访问，只有通过一些别名才能访问，如126只能通过[www.126.com](http://www.126.com/)和126.com两个主机别名访问，而不能通过它的主机名zz-9-77-a8.bta.net.cn来访问。这是因为在服务端通过[HTTP协议](https://www.baidu.com/s?wd=HTTP%E5%8D%8F%E8%AE%AE&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)做了限制，这个在前面已经讨论过了。例程3-8对比了getCanonicalHostName和getHostName方法在不同情况下的输出结果。

1. URL

统一资源定位符，标识Internet上某一资源的地址。

URL由两部分组成：协议名称和资源名称，中间用冒号隔开。

在java.net包中，提供了URL类来表示URL。