Network Programming
[CACS355]
BCA 6th Sem

Er. Sital Prasad Mandal

(Email : info.sitalmandal@gmail.com)
Bhadrapur, Jhapa, Nepal

https://networkprogam-mmc.blogspot.com/

Unit-2
Internet Addresses

Unit-2

Internet Addresses

- IP (Internet Protocol) Addresses
  - IPv4 (4 Bytes): dotted quad format
    - www.tu.edu.np 140.127.208.17
  - IPv6 (16 Bytes): 8 blocks of 4 hexadecimal digits separated by colons
    - www.tu.edu.np ::ffff:8c7f:d011
    - 2400:cb00:2048:0001:0000:0000:6ca2:c665 → 2400:cb00:2048:1::6ca2:c665
  - Mixed: last 4 bytes of the IPv6 written as an IPv4 dotted quad address
    - www.tu.edu.np ::ffff:140.127.208.17
    - FEDC:BA98:7654:3210:FEDC:BA98:7654:3210
      FEDC:BA98:7654:3210:FEDC:BA98:118.84.50.16
- Domain Names – Resolved by DNS Servers
  - FQDN: Fully Qualified Domain Name
    - www.bca.tu.edu.np.
  - One name can map to multiple IP addresses
  - One IP addresses can also have multiple names

Unit-2

The InetAddress Class

- Creating new InetAddress objects
  - No public constructors; use static factory methods directly
    - Automatically connect to a DNS server to resolve a hostname
    - Throws an UnknownHostException, a subclass of IOException, if not found

- getByName(): lookup the name and the numeric address
- getAllByName(): lookup all the addresses of a host
- getLocalHost(): return an InetAddress object for the local host
  - Return 'localhost/127.0.0.1' if lookup failed
- getByAddress(): create an InetAddress object from given address
  - Without talking to DNS

Unit-2

The InetAddress Class

Creating New InetAddress Objects

```
InetAddress address = InetAddress.getByName ("www.google.com.np" );
InetAddress address = InetAddress.getByName("208.201.239.100");
```

Syntax

```
InetAddress address = InetAddress.getByName("www. tu.edu.np");
System.out.println(address);
```

Result

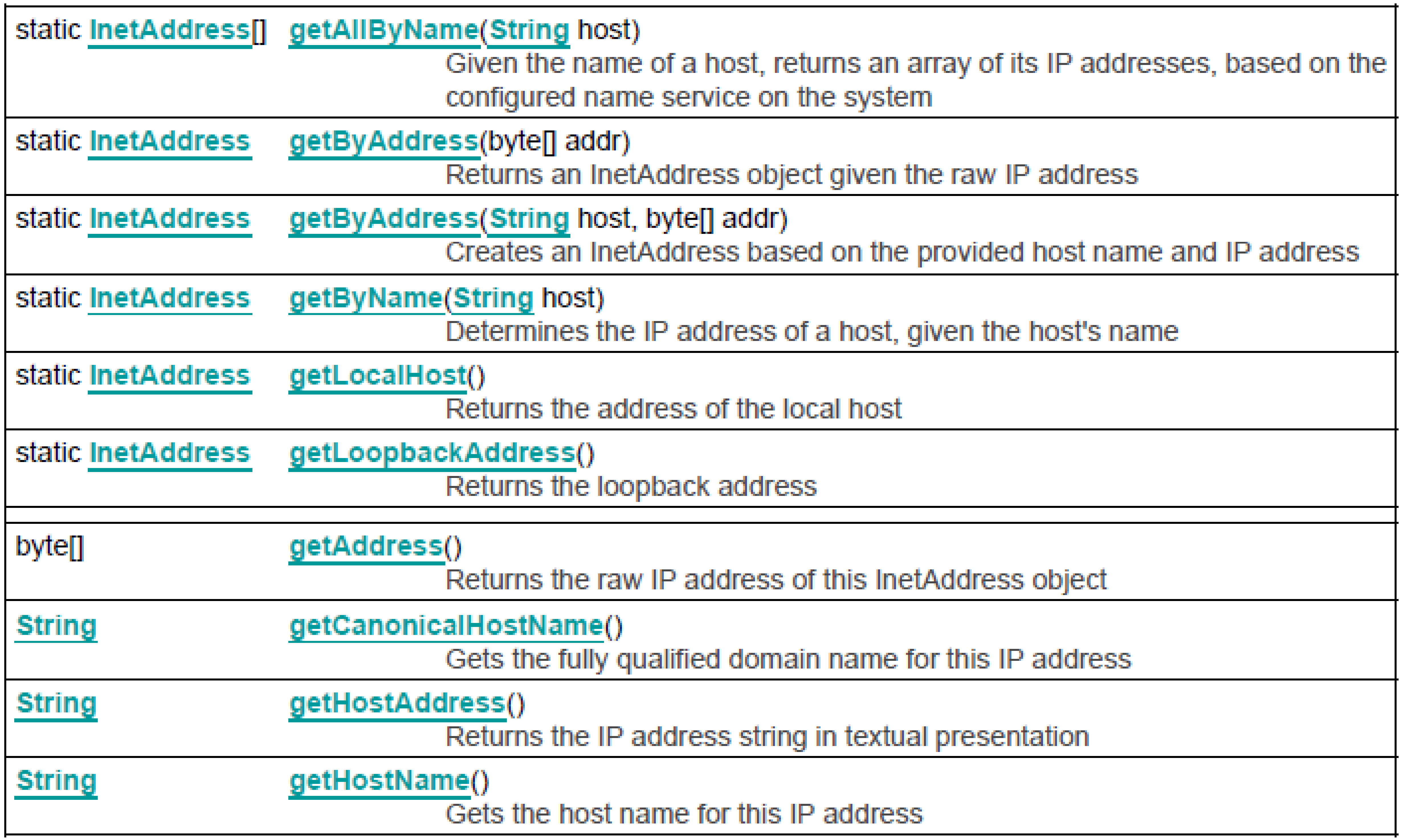
```
% java JavaFile
www.tu.edu.np/208.201.239.36
```

https://networkprogam-mmc.blogspot.com/

Unit-2

Getter Methods

Create Objects and Getter Methods



static InetAddress[]	getAllByName(String host) Given the name of a host, returns an array of its IP addresses, based on the configured name service on the system
static InetAddress	getByAddress(byte[] addr) Returns an InetAddress object given the raw IP address
static InetAddress	getByAddress(String host, byte[] addr) Creates an InetAddress based on the provided host name and IP address
static InetAddress	getByName(String host) Determines the IP address of a host, given the host's name
static InetAddress	getLocalHost() Returns the address of the local host
static InetAddress	getLoopbackAddress() Returns the loopback address
byte[]	getAddress() Returns the raw IP address of this InetAddress object
String	getCanonicalHostName() Gets the fully qualified domain name for this IP address
String	getHostAddress() Returns the IP address string in textual presentation
String	getHostName() Gets the host name for this IP address

Unit-2

Getter Methods

Example 4-3. Given the address, find the hostname

```java
import java.net.*;

public class ReverseTest {

  public static void main (String[] args) throws UnknownHostException {
    InetAddress ia = InetAddress.getByName("208.201.239.100");
    System.out.println(ia.getCanonicalHostName());
  }
}
```

```
% java ReverseTest
oreilly.com
```

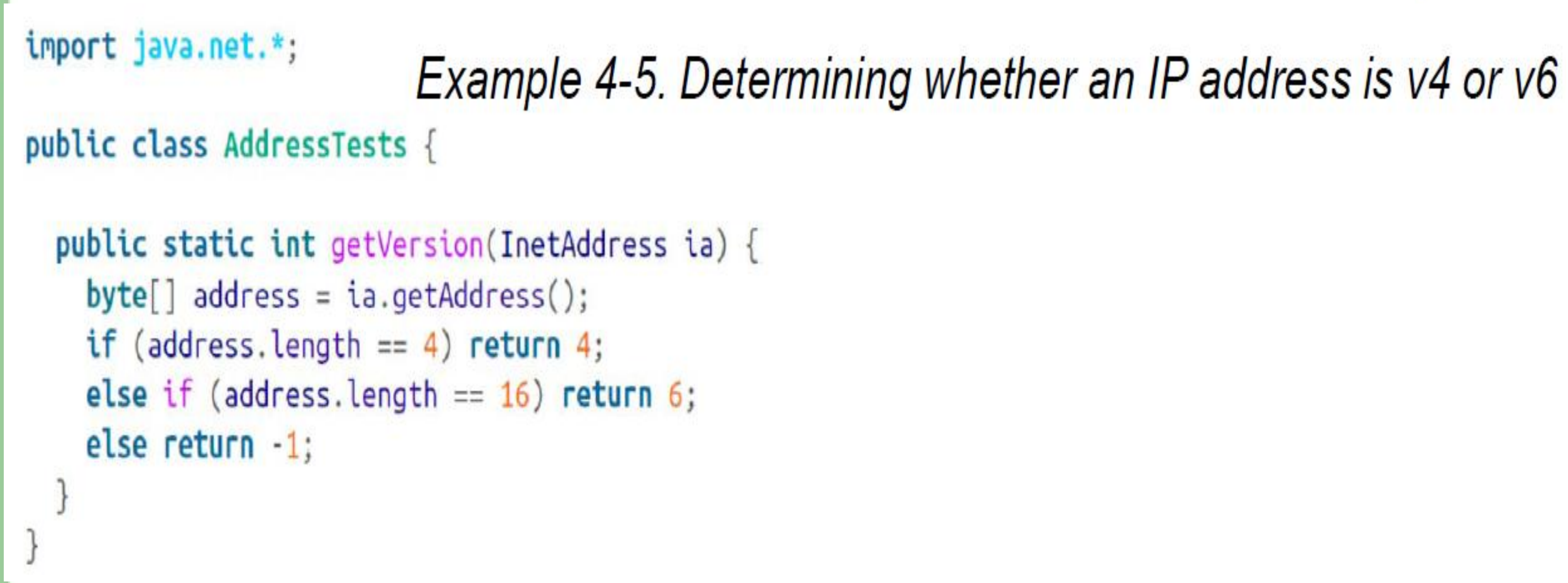
Example 4-4. Find the IP address of the local machine

```java
import java.net.*;

public class MyAddress {

  public static void main(String[] args) {
    try {
      InetAddress me = InetAddress.getLocalHost();
      String dottedQuad = me.getHostAddress();
      System.out.println("My address is " + dottedQuad);
    } catch (UnknownHostException ex) {
      System.out.println("I'm sorry. I don't know my own address.");
    }
  }
}
```

```
% java MyAddress
My address is 152.2.22.14.
```

Unit-2

Getter Methods

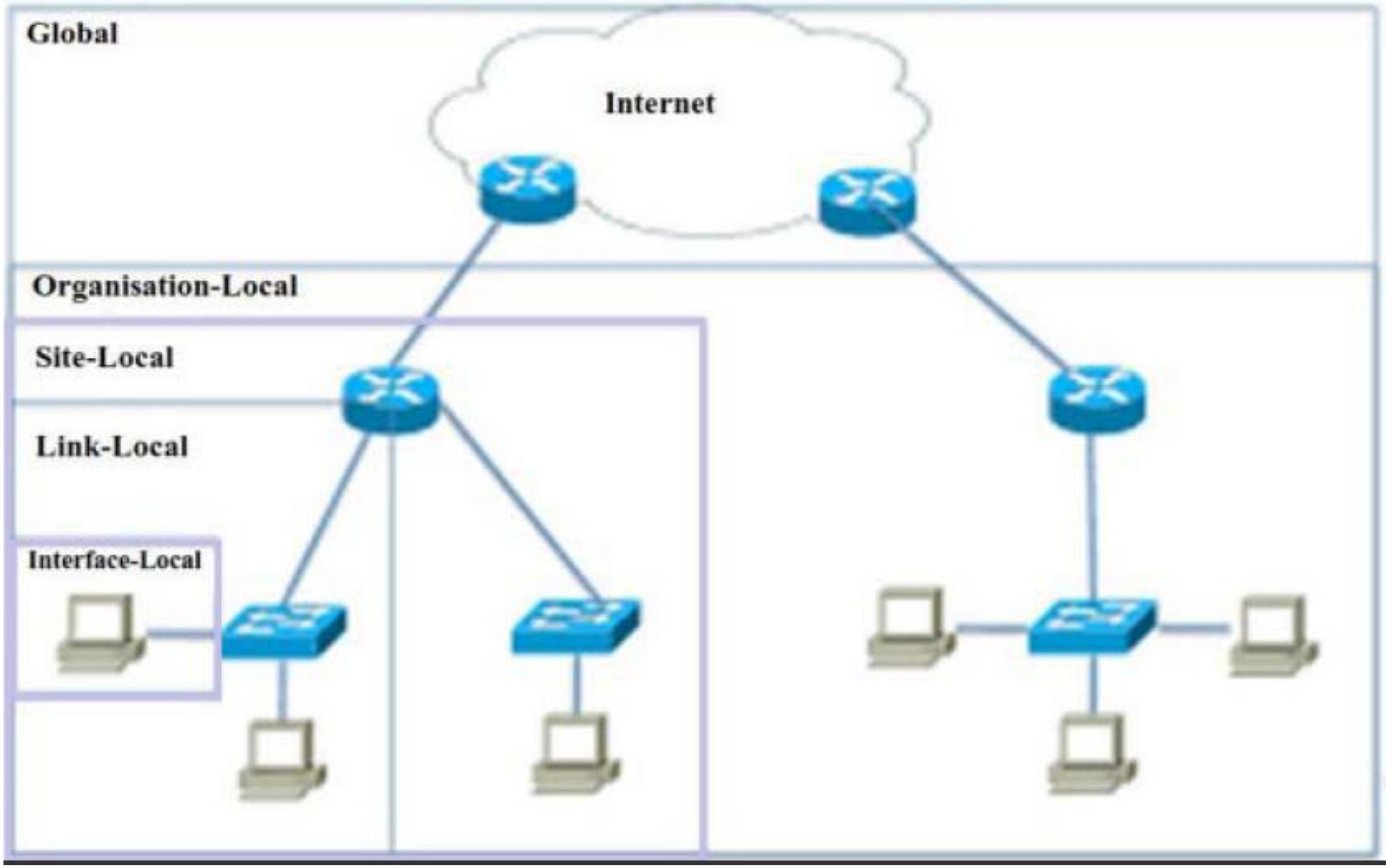


Example 4-5. Determining whether an IP address is v4 or v6

```
import java.net.*;

public class AddressTests {

  public static int getVersion(InetAddress ia) {
    byte[] address = ia.getAddress();
    if (address.length == 4) return 4;
    else if (address.length == 16) return 6;
    else return -1;
  }
}
```

Unit-2

Address Types



https://networkprogam-mmc.blogspot.com/

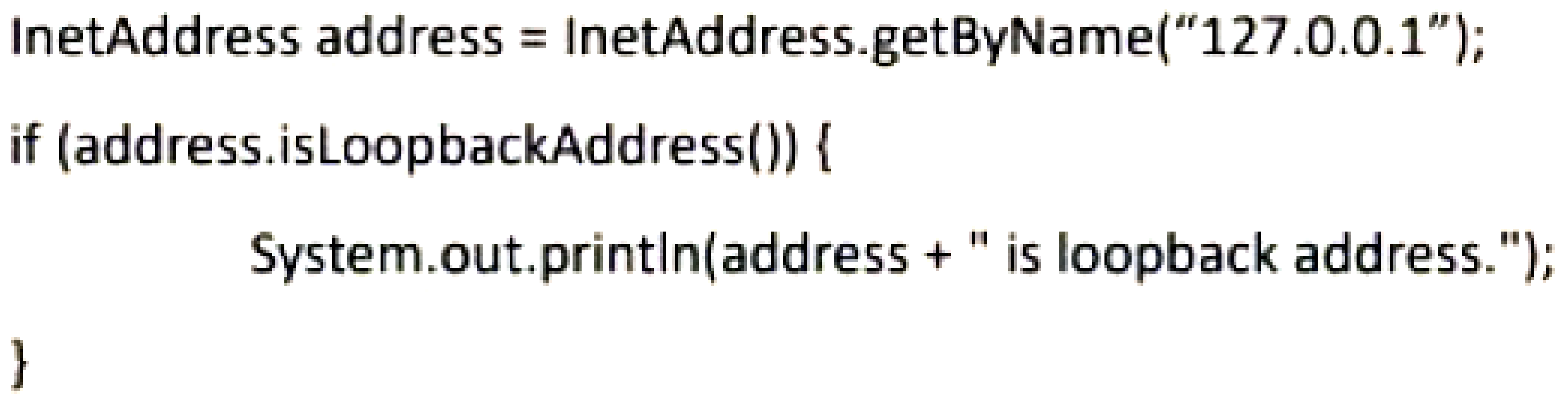
Unit-2

Address Types

boolean	isAnyLocalAddress()	Utility routine to check if the InetAddress in a wildcard address (0.0.0.0 / ::)
boolean	isLinkLocalAddress()	Utility routine to check if the InetAddress is an IPv6 link local address (Begin with FE80:0000:0000:0000 (8 Bytes) + Local address (often MAC))
boolean	isLoopbackAddress()	Utility routine to check if the InetAddress is a loopback address (127.0.0.1 / ::1)
boolean	isMCGlobal()	Utility routine to check if the multicast address has global scope (IPv4-all Multicast/IPv6-begin with FF0E or FF1E)
boolean	isMCLinkLocal()	Utility routine to check if the multicast address has subnet/link scope (IPv4-all Multicast/IPv6-begin with FF02 or FF12)
boolean	isMCNodeLocal()	Utility routine to check if the multicast address has node scope (for test) (IPv4-all Multicast/IPv6-begin with FF01 or FF11)
boolean	isMCOrgLocal()	Utility routine to check if the multicast address has organization scope (IPv6-begin with FF08 or FF18)
boolean	isMCSiteLocal()	Utility routine to check if the multicast address has site scope (IPv6-begin with FF05 or FF15)
boolean	isMulticastAddress()	Utility routine to check if the InetAddress is an IP multicast address (224.0.0.0~239.255.255.255 / FF00::)
boolean	isReachable(int timeout)	Test whether that address is reachable (Use traceroute/ICMP echo requests)
boolean	isReachable(NetworkInterface netif, int ttl, int timeout)	Test whether that address is reachable
boolean	isSiteLocalAddress()	Utility routine to check if the InetAddress is a IPv6 site local address Like LinkLocalAddress, but May be forwarded by routers (Begin with EEC0:0000:0000:0000 (8 Bytes) + Local address (often MAC))

Unit-2

Testing Reachability



```
InetAddress address = InetAddress.getByName("127.0.0.1");
if (address.isLoopbackAddress()) {
            System.out.println(address + " is loopback address.");
}
```

Unit-2

Object Methods

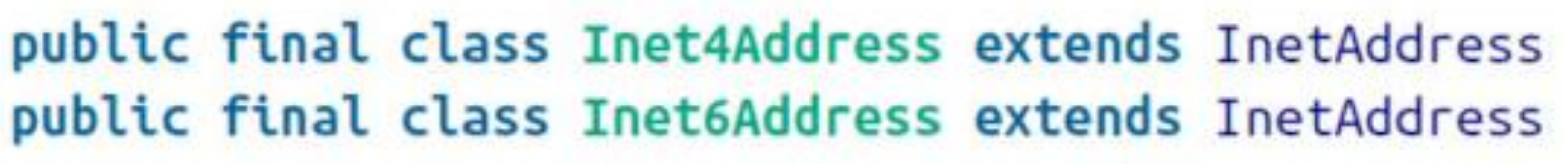
Example 4-7. Are www.ibiblio.org and helios.ibiblio.org the same?

```java
public class IBiblioAliases {

  public static void main (String args[]) {
    try {
      InetAddress ibiblio = InetAddress.getByName("www.ibiblio.org");
      InetAddress helios = InetAddress.getByName("helios.ibiblio.org");
      if (ibiblio.equals(helios)) {
        System.out.println
            ("www.ibiblio.org is the same as helios.ibiblio.org");
      } else {
        System.out.println
            ("www.ibiblio.org is not the same as helios.ibiblio.org");
      }
    } catch (UnknownHostException ex) {
      System.out.println("Host lookup failed.");
    }
  }
}
```

```
% java IBiblioAliases
www.ibiblio.org is the same as helios.ibiblio.org
```

Unit-2

Inet4Address and Inet6Address



```
public final class Inet4Address extends InetAddress
public final class Inet6Address extends InetAddress
```

- Both overrides several of the methods in InetAddress but does not change their behavior in
  - Most of the time, simply not needed to know this
- Inet6Address.isIPv4CompatibleAddress(): one new method
  - Only the last four bytes are nonzero – IPv4 address stuffed into an IPv6
  - 0:0:0:0:0:0:d.d.d.d

Unit-2

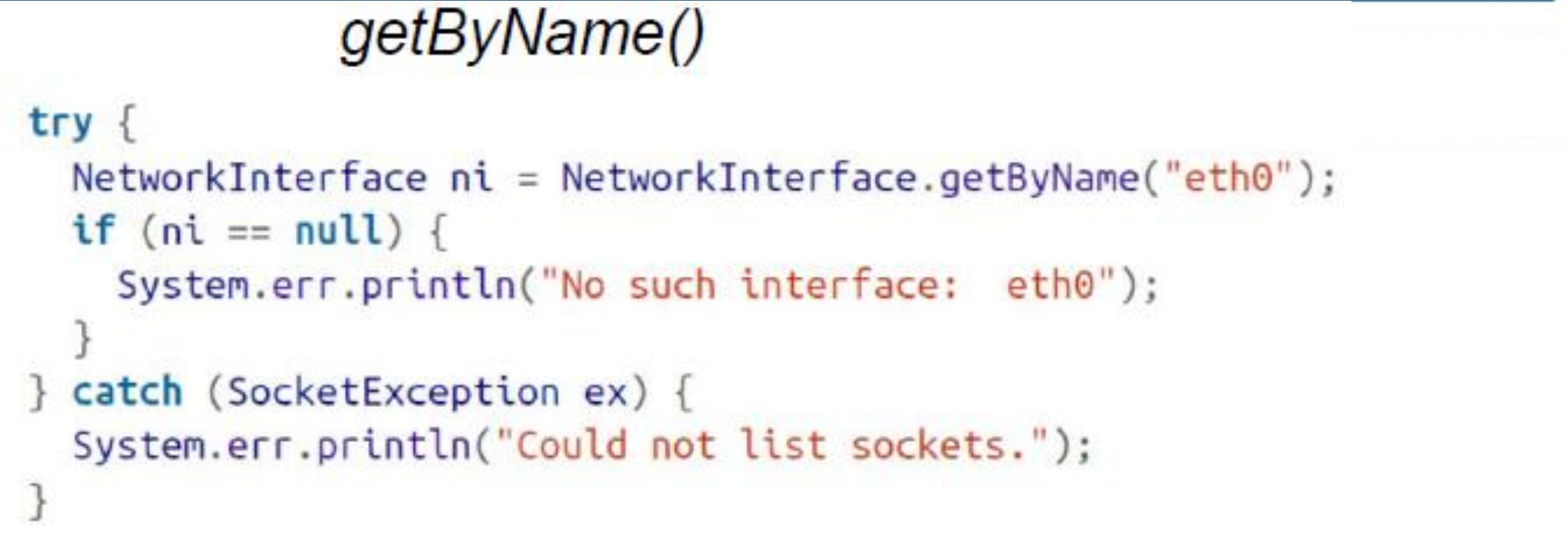
NetworkInterface Factory Methods

- java.net.NetworkInterface objects represent physical hardware and virtual addresses

static NetworkInterface	getByIndex(int index) Get a network interface given its index
static NetworkInterface	getByInetAddress(InetAddress addr) Convenience method to search for a network interface that has the specified Internet Protocol (IP) address bound to it
static NetworkInterface	getByName(String name) Searches for the network interface with the specified name
Enumeration <InetAddress>	getInetAddresses() Convenience method to return an Enumeration with all or a subset of the InetAddresses bound to this network interface
List<InterfaceAddress>	getInterfaceAddresses() Get a List of all or a subset of the InterfaceAddresses of this network interface
static Enumeration <NetworkInterface>	getNetworkInterfaces() Returns all the interfaces on this machine
NetworkInterface	getParent() Returns the parent NetworkInterface of this interface if this is a subinterface, or null if it is a physical (non virtual) interface or has no parent
Enumeration <NetworkInterface>	getSubInterfaces() Get an Enumeration with all the subinterfaces (also known as virtual interfaces) attached to this network interface

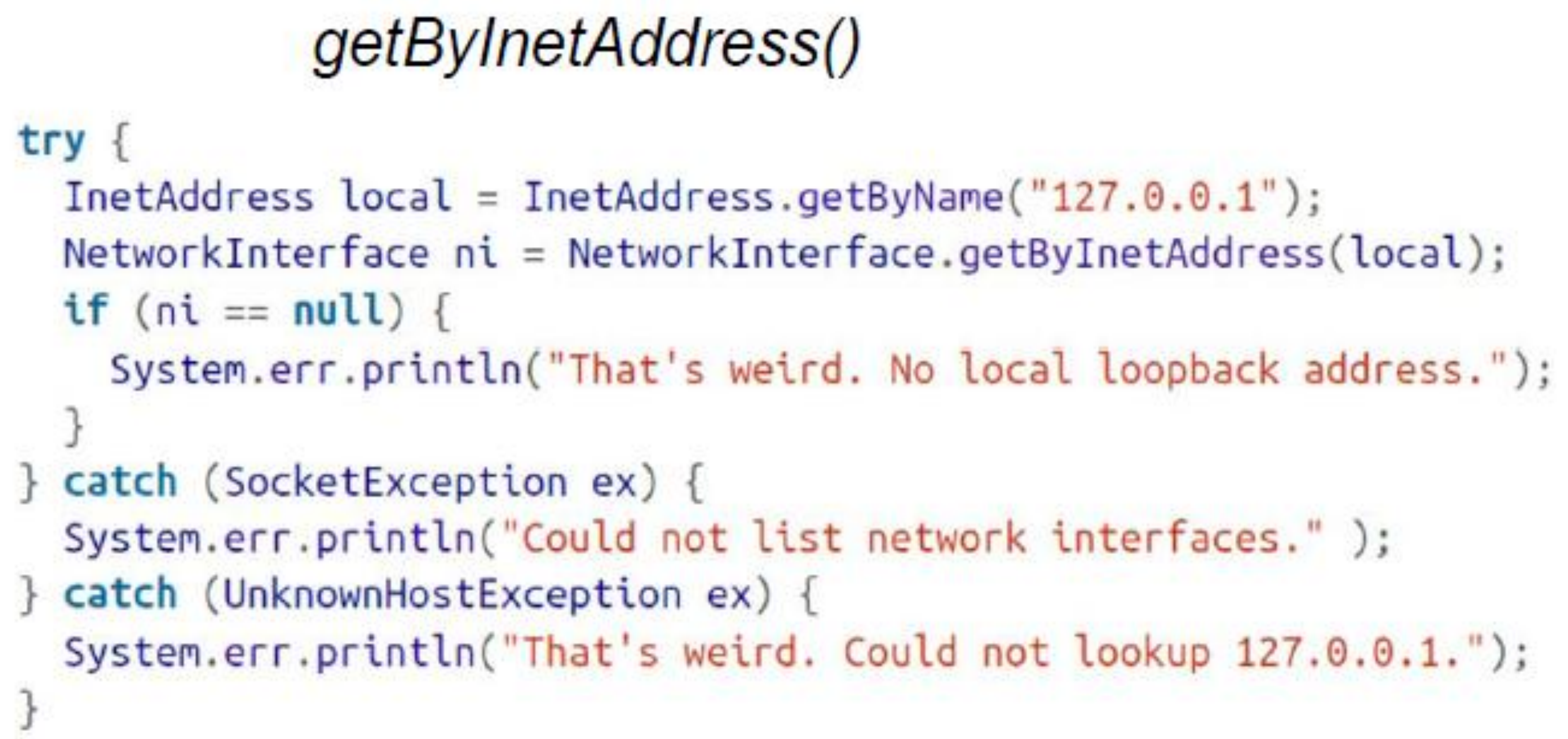
Unit-2

The NetworkInterface Class



getByName()

```
try {
  NetworkInterface ni = NetworkInterface.getByName("eth0");
  if (ni == null) {
    System.err.println("No such interface:  eth0");
  }
} catch (SocketException ex) {
  System.err.println("Could not list sockets.");
}
```

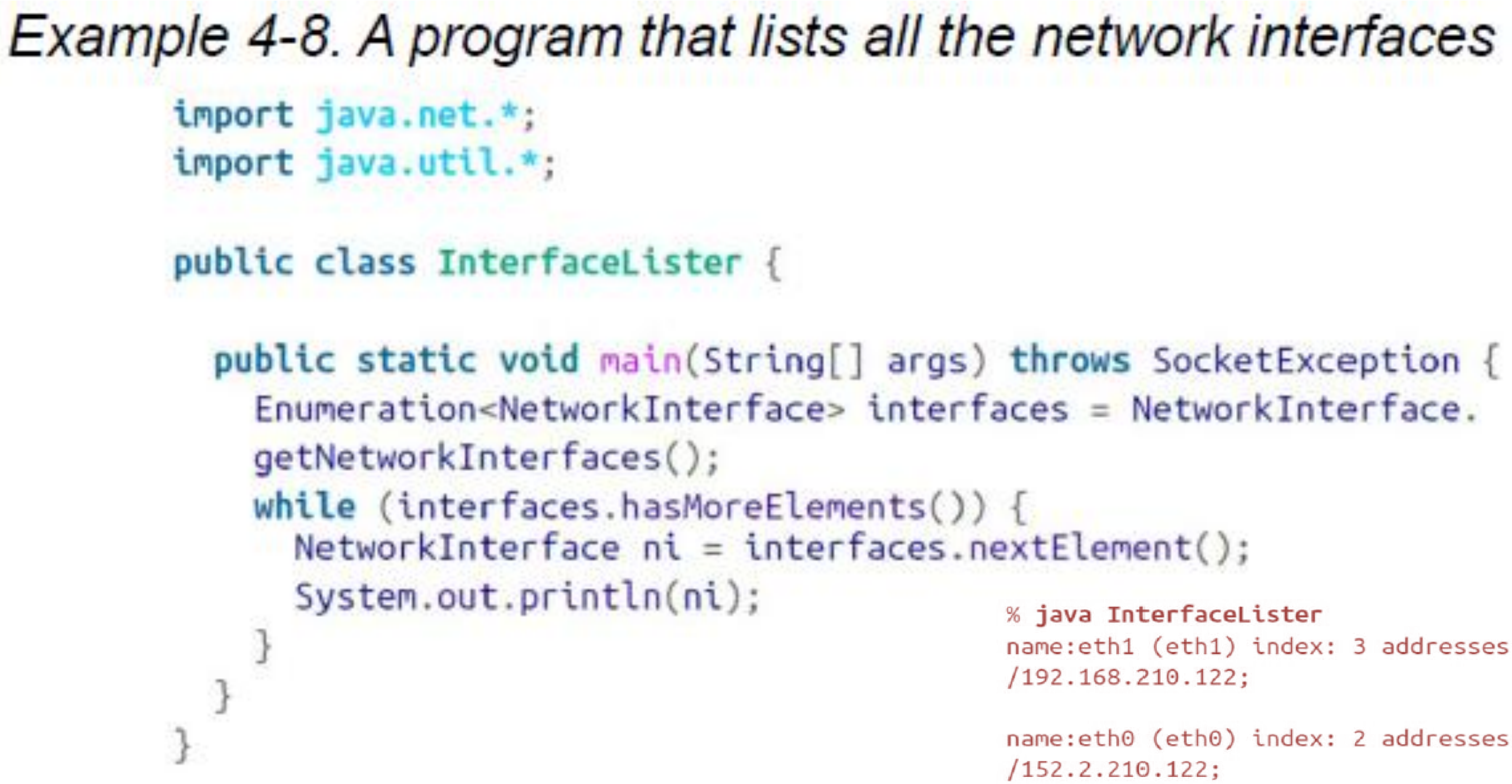


getByInetAddress()

```
try {
  InetAddress local = InetAddress.getByName("127.0.0.1");
  NetworkInterface ni = NetworkInterface.getByInetAddress(local);
  if (ni == null) {
    System.err.println("That's weird. No local loopback address.");
  }
} catch (SocketException ex) {
  System.err.println("Could not list network interfaces." );
} catch (UnknownHostException ex) {
  System.err.println("That's weird. Could not lookup 127.0.0.1.");
}
```

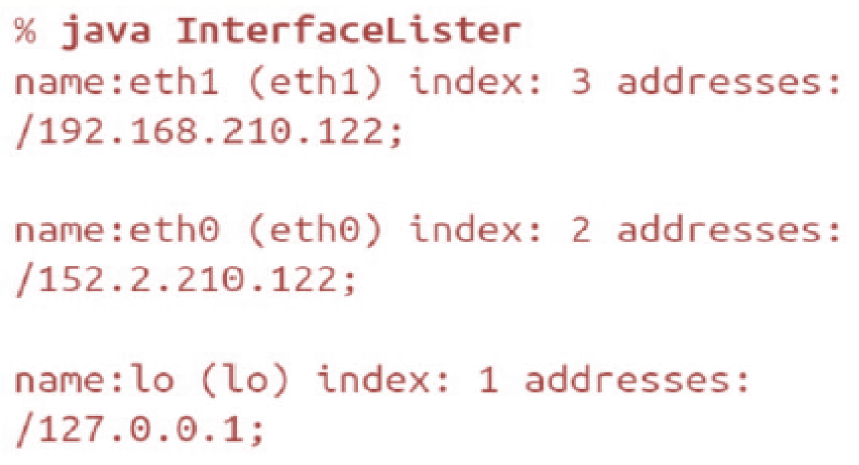
Unit-2

The NetworkInterface Class



Example 4-8. A program that lists all the network interfaces

```
import java.net.*;
import java.util.*;

public class InterfaceLister {

  public static void main(String[] args) throws SocketException {
    Enumeration<NetworkInterface> interfaces = NetworkInterface.
    getNetworkInterfaces();
    while (interfaces.hasMoreElements()) {
      NetworkInterface ni = interfaces.nextElement();
      System.out.println(ni);
    }
  }
}
```

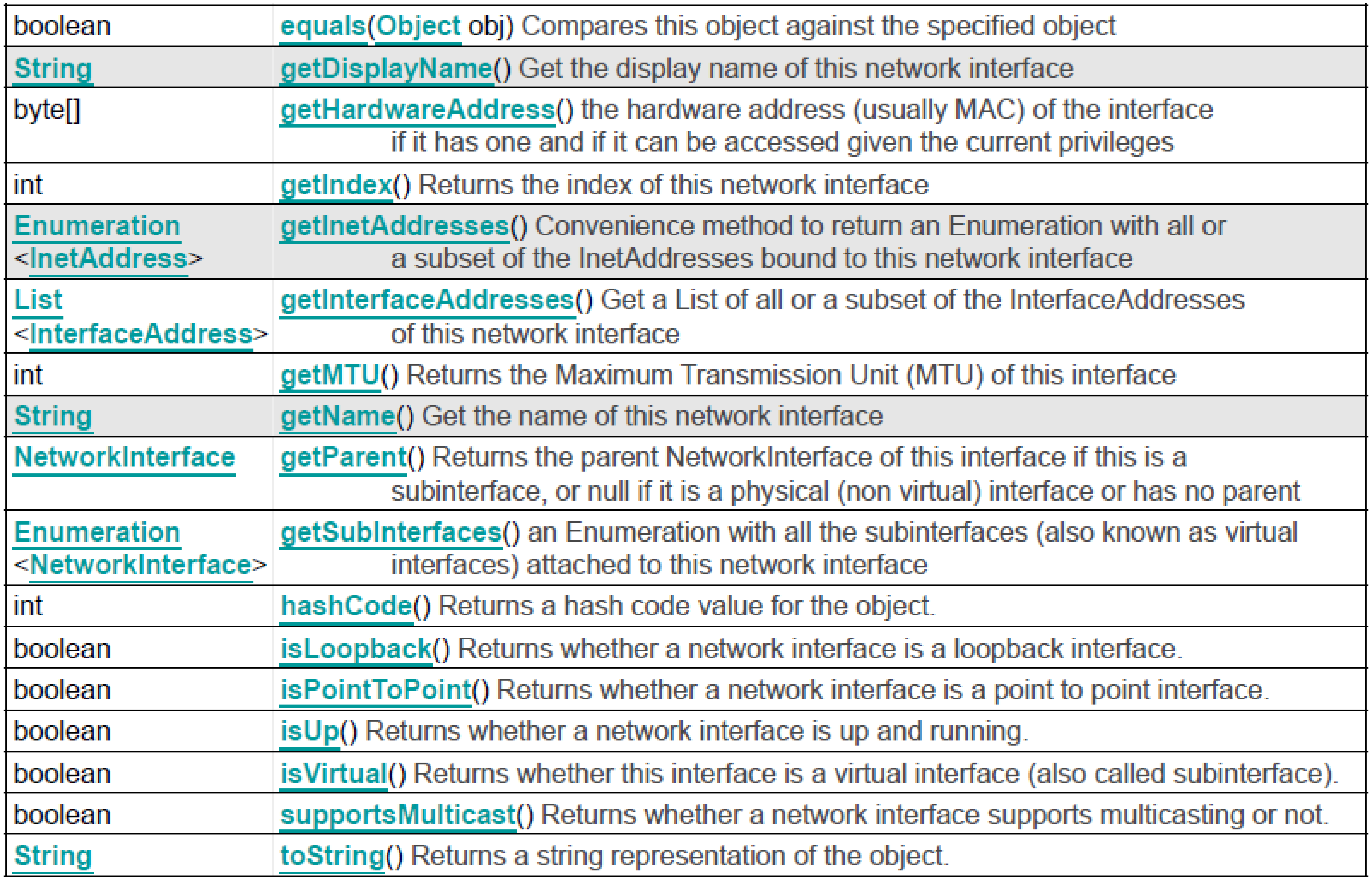


```
% java InterfaceLister
name:eth1 (eth1) index: 3 addresses:
/192.168.210.122;

name:eth0 (eth0) index: 2 addresses:
/152.2.210.122;

name:lo (lo) index: 1 addresses:
/127.0.0.1;
```

https://networkprogam-mmc.blogspot.com/

Unit-2

The NetworkInterface Getter Methods



boolean	equals(Object obj) Compares this object against the specified object
String	getDisplayName() Get the display name of this network interface
byte[]	getHardwareAddress() the hardware address (usually MAC) of the interface if it has one and if it can be accessed given the current privileges
int	getIndex() Returns the index of this network interface
Enumeration <InetAddress>	getInetAddresses() Convenience method to return an Enumeration with all or a subset of the InetAddresses bound to this network interface
List <InterfaceAddress>	getInterfaceAddresses() Get a List of all or a subset of the InterfaceAddresses of this network interface
int	getMTU() Returns the Maximum Transmission Unit (MTU) of this interface
String	getName() Get the name of this network interface
NetworkInterface	getParent() Returns the parent NetworkInterface of this interface if this is a subinterface, or null if it is a physical (non virtual) interface or has no parent
Enumeration <NetworkInterface>	getSubInterfaces() an Enumeration with all the subinterfaces (also known as virtual interfaces) attached to this network interface
int	hashCode() Returns a hash code value for the object.
boolean	isLoopback() Returns whether a network interface is a loopback interface.
boolean	isPointToPoint() Returns whether a network interface is a point to point interface.
boolean	isUp() Returns whether a network interface is up and running.
boolean	isVirtual() Returns whether this interface is a virtual interface (also called subinterface).
boolean	supportsMulticast() Returns whether a network interface supports multicasting or not.
String	toString() Returns a string representation of the object.