| | [**Overview**](http://docs.google.com/overview-summary.html) | **Package** | Class | [**Use**](http://docs.google.com/package-use.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV PACKAGE**](http://docs.google.com/java/math/package-summary.html)   [**NEXT PACKAGE**](http://docs.google.com/java/nio/package-summary.html) | [**FRAMES**](http://docs.google.com/index.html?java/net/package-summary.html)    [**NO FRAMES**](http://docs.google.com/package-summary.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |

## Package java.net

Provides the classes for implementing networking applications.

**See:**

[**Description**](#3znysh7)

| **Interface Summary** | |
| --- | --- |
| [**ContentHandlerFactory**](http://docs.google.com/java/net/ContentHandlerFactory.html) | This interface defines a factory for content handlers. |
| [**CookiePolicy**](http://docs.google.com/java/net/CookiePolicy.html) | CookiePolicy implementations decide which cookies should be accepted and which should be rejected. |
| [**CookieStore**](http://docs.google.com/java/net/CookieStore.html) | A CookieStore object represents a storage for cookie. |
| [**DatagramSocketImplFactory**](http://docs.google.com/java/net/DatagramSocketImplFactory.html) | This interface defines a factory for datagram socket implementations. |
| [**FileNameMap**](http://docs.google.com/java/net/FileNameMap.html) | A simple interface which provides a mechanism to map between a file name and a MIME type string. |
| [**SocketImplFactory**](http://docs.google.com/java/net/SocketImplFactory.html) | This interface defines a factory for socket implementations. |
| [**SocketOptions**](http://docs.google.com/java/net/SocketOptions.html) | Interface of methods to get/set socket options. |
| [**URLStreamHandlerFactory**](http://docs.google.com/java/net/URLStreamHandlerFactory.html) | This interface defines a factory for URL stream protocol handlers. |

| **Class Summary** | |
| --- | --- |
| [**Authenticator**](http://docs.google.com/java/net/Authenticator.html) | The class Authenticator represents an object that knows how to obtain authentication for a network connection. |
| [**CacheRequest**](http://docs.google.com/java/net/CacheRequest.html) | Represents channels for storing resources in the ResponseCache. |
| [**CacheResponse**](http://docs.google.com/java/net/CacheResponse.html) | Represent channels for retrieving resources from the ResponseCache. |
| [**ContentHandler**](http://docs.google.com/java/net/ContentHandler.html) | The abstract class ContentHandler is the superclass of all classes that read an Object from a URLConnection. |
| [**CookieHandler**](http://docs.google.com/java/net/CookieHandler.html) | A CookieHandler object provides a callback mechanism to hook up a HTTP state management policy implementation into the HTTP protocol handler. |
| [**CookieManager**](http://docs.google.com/java/net/CookieManager.html) | CookieManager provides a concrete implementation of [CookieHandler](http://docs.google.com/java/net/CookieHandler.html), which separates the storage of cookies from the policy surrounding accepting and rejecting cookies. |
| [**DatagramPacket**](http://docs.google.com/java/net/DatagramPacket.html) | This class represents a datagram packet. |
| [**DatagramSocket**](http://docs.google.com/java/net/DatagramSocket.html) | This class represents a socket for sending and receiving datagram packets. |
| [**DatagramSocketImpl**](http://docs.google.com/java/net/DatagramSocketImpl.html) | Abstract datagram and multicast socket implementation base class. |
| [**HttpCookie**](http://docs.google.com/java/net/HttpCookie.html) | An HttpCookie object represents an http cookie, which carries state information between server and user agent. |
| [**HttpURLConnection**](http://docs.google.com/java/net/HttpURLConnection.html) | A URLConnection with support for HTTP-specific features. |
| [**IDN**](http://docs.google.com/java/net/IDN.html) | Provides methods to convert internationalized domain names (IDNs) between a normal Unicode representation and an ASCII Compatible Encoding (ACE) representation. |
| [**Inet4Address**](http://docs.google.com/java/net/Inet4Address.html) | This class represents an Internet Protocol version 4 (IPv4) address. |
| [**Inet6Address**](http://docs.google.com/java/net/Inet6Address.html) | This class represents an Internet Protocol version 6 (IPv6) address. |
| [**InetAddress**](http://docs.google.com/java/net/InetAddress.html) | This class represents an Internet Protocol (IP) address. |
| [**InetSocketAddress**](http://docs.google.com/java/net/InetSocketAddress.html) | This class implements an IP Socket Address (IP address + port number) It can also be a pair (hostname + port number), in which case an attempt will be made to resolve the hostname. |
| [**InterfaceAddress**](http://docs.google.com/java/net/InterfaceAddress.html) | This class represents a Network Interface address. |
| [**JarURLConnection**](http://docs.google.com/java/net/JarURLConnection.html) | A URL Connection to a Java ARchive (JAR) file or an entry in a JAR file. |
| [**MulticastSocket**](http://docs.google.com/java/net/MulticastSocket.html) | The multicast datagram socket class is useful for sending and receiving IP multicast packets. |
| [**NetPermission**](http://docs.google.com/java/net/NetPermission.html) | This class is for various network permissions. |
| [**NetworkInterface**](http://docs.google.com/java/net/NetworkInterface.html) | This class represents a Network Interface made up of a name, and a list of IP addresses assigned to this interface. |
| [**PasswordAuthentication**](http://docs.google.com/java/net/PasswordAuthentication.html) | The class PasswordAuthentication is a data holder that is used by Authenticator. |
| [**Proxy**](http://docs.google.com/java/net/Proxy.html) | This class represents a proxy setting, typically a type (http, socks) and a socket address. |
| [**ProxySelector**](http://docs.google.com/java/net/ProxySelector.html) | Selects the proxy server to use, if any, when connecting to the network resource referenced by a URL. |
| [**ResponseCache**](http://docs.google.com/java/net/ResponseCache.html) | Represents implementations of URLConnection caches. |
| [**SecureCacheResponse**](http://docs.google.com/java/net/SecureCacheResponse.html) | Represents a cache response originally retrieved through secure means, such as TLS. |
| [**ServerSocket**](http://docs.google.com/java/net/ServerSocket.html) | This class implements server sockets. |
| [**Socket**](http://docs.google.com/java/net/Socket.html) | This class implements client sockets (also called just "sockets"). |
| [**SocketAddress**](http://docs.google.com/java/net/SocketAddress.html) | This class represents a Socket Address with no protocol attachment. |
| [**SocketImpl**](http://docs.google.com/java/net/SocketImpl.html) | The abstract class SocketImpl is a common superclass of all classes that actually implement sockets. |
| [**SocketPermission**](http://docs.google.com/java/net/SocketPermission.html) | This class represents access to a network via sockets. |
| [**URI**](http://docs.google.com/java/net/URI.html) | Represents a Uniform Resource Identifier (URI) reference. |
| [**URL**](http://docs.google.com/java/net/URL.html) | Class URL represents a Uniform Resource Locator, a pointer to a "resource" on the World Wide Web. |
| [**URLClassLoader**](http://docs.google.com/java/net/URLClassLoader.html) | This class loader is used to load classes and resources from a search path of URLs referring to both JAR files and directories. |
| [**URLConnection**](http://docs.google.com/java/net/URLConnection.html) | The abstract class URLConnection is the superclass of all classes that represent a communications link between the application and a URL. |
| [**URLDecoder**](http://docs.google.com/java/net/URLDecoder.html) | Utility class for HTML form decoding. |
| [**URLEncoder**](http://docs.google.com/java/net/URLEncoder.html) | Utility class for HTML form encoding. |
| [**URLStreamHandler**](http://docs.google.com/java/net/URLStreamHandler.html) | The abstract class URLStreamHandler is the common superclass for all stream protocol handlers. |

| **Enum Summary** | |
| --- | --- |
| [**Authenticator.RequestorType**](http://docs.google.com/java/net/Authenticator.RequestorType.html) | The type of the entity requesting authentication. |
| [**Proxy.Type**](http://docs.google.com/java/net/Proxy.Type.html) | Represents the proxy type. |

| **Exception Summary** | |
| --- | --- |
| [**BindException**](http://docs.google.com/java/net/BindException.html) | Signals that an error occurred while attempting to bind a socket to a local address and port. |
| [**ConnectException**](http://docs.google.com/java/net/ConnectException.html) | Signals that an error occurred while attempting to connect a socket to a remote address and port. |
| [**HttpRetryException**](http://docs.google.com/java/net/HttpRetryException.html) | Thrown to indicate that a HTTP request needs to be retried but cannot be retried automatically, due to streaming mode being enabled. |
| [**MalformedURLException**](http://docs.google.com/java/net/MalformedURLException.html) | Thrown to indicate that a malformed URL has occurred. |
| [**NoRouteToHostException**](http://docs.google.com/java/net/NoRouteToHostException.html) | Signals that an error occurred while attempting to connect a socket to a remote address and port. |
| [**PortUnreachableException**](http://docs.google.com/java/net/PortUnreachableException.html) | Signals that an ICMP Port Unreachable message has been received on a connected datagram. |
| [**ProtocolException**](http://docs.google.com/java/net/ProtocolException.html) | Thrown to indicate that there is an error in the underlying protocol, such as a TCP error. |
| [**SocketException**](http://docs.google.com/java/net/SocketException.html) | Thrown to indicate that there is an error in the underlying protocol, such as a TCP error. |
| [**SocketTimeoutException**](http://docs.google.com/java/net/SocketTimeoutException.html) | Signals that a timeout has occurred on a socket read or accept. |
| [**UnknownHostException**](http://docs.google.com/java/net/UnknownHostException.html) | Thrown to indicate that the IP address of a host could not be determined. |
| [**UnknownServiceException**](http://docs.google.com/java/net/UnknownServiceException.html) | Thrown to indicate that an unknown service exception has occurred. |
| [**URISyntaxException**](http://docs.google.com/java/net/URISyntaxException.html) | Checked exception thrown to indicate that a string could not be parsed as a URI reference. |

## Package java.net Description

Provides the classes for implementing networking applications.

The java.net package can be roughly divided in two sections:

* *A Low Level API*, which deals with the following abstractions:
  + *Addresses*, which are networking identifiers, like IP addresses.
  + *Sockets*, which are basic bidirectional data communication mechanisms.
  + *Interfaces*, which describe network interfaces.
* *A High Level API*, which deals with the following abstractions:
  + *URIs*, which represent Universal Resource Identifiers.
  + *URLs*, which represent Universal Resource Locators.
  + *Connections*, which represents connections to the resource pointed to by *URLs*.

## Addresses

Addresses are used throughout the java.net APIs as either host identifiers, or socket endpoint identifiers.

The [InetAddress](http://docs.google.com/java/net/InetAddress.html) class is the abstraction representing an IP (Internet Protocol) address. It has two subclasses:

* [Inet4Address](http://docs.google.com/java/net/Inet4Address.html) for IPv4 addresses.
* [Inet6Address](http://docs.google.com/java/net/Inet6Address.html) for IPv6 addresses.

But, in most cases, there is no need to deal directly with the subclasses, as the InetAddress abstraction should cover most of the needed functionality.

### **About IPv6**

Not all systems have support for the IPv6 protocol, and while the Java networking stack will attempt to detect it and use it transparently when available, it is also possible to disable its use with a system property. In the case where IPv6 is not available, or explicitly disabled, Inet6Address are not valid arguments for most networking operations any more. While methods like [InetAddress.getByName(java.lang.String)](http://docs.google.com/java/net/InetAddress.html#getByName(java.lang.String)) are guaranteed not to return an Inet6Address when looking up host names, it is possible, by passing literals, to create such an object. In which case, most methods, when called with an Inet6Address will throw an Exception.

## Sockets

Sockets are means to establish a communication link between machines over the network. The java.net package provides 4 kinds of Sockets:

* [Socket](http://docs.google.com/java/net/Socket.html) is a TCP client API, and will typically be used to [connect](http://docs.google.com/java/net/Socket.html#connect(java.net.SocketAddress)) to a remote host.
* [ServerSocket](http://docs.google.com/java/net/ServerSocket.html) is a TCP server API, and will typically [accept](http://docs.google.com/java/net/ServerSocket.html#accept()) connections from client sockets.
* [DatagramSocket](http://docs.google.com/java/net/DatagramSocket.html) is a UDP endpoint API and is used to [send](http://docs.google.com/java/net/DatagramSocket.html#send(java.net.DatagramPacket)) and [receive](http://docs.google.com/java/net/DatagramSocket.html#receive(java.net.DatagramPacket)) [datagram packets](http://docs.google.com/java/net/DatagramPacket.html).
* [MulticastSocket](http://docs.google.com/java/net/MulticastSocket.html) is a subclass of DatagramSocket used when dealing with multicast groups.

Sending and receiving with TCP sockets is done through InputStreams and OutputStreams which can be obtained via the [Socket.getInputStream()](http://docs.google.com/java/net/Socket.html#getInputStream()) and [Socket.getOutputStream()](http://docs.google.com/java/net/Socket.html#getOutputStream()) methods.

## Interfaces

The [NetworkInterface](http://docs.google.com/java/net/NetworkInterface.html) class provides APIs to browse and query all the networking interfaces (e.g. ethernet connection or PPP endpoint) of the local machine. It is through that class that you can check if any of the local interfaces is configured to support IPv6.

## High level API

A number of classes in the java.net package do provide for a much higher level of abstraction and allow for easy access to resources on the network. The classes are:

* [URI](http://docs.google.com/java/net/URI.html) is the class representing a Universal Resource Identifier, as specified in RFC 2396. As the name indicates, this is just an Identifier and doesn't provide directly the means to access the resource.
* [URL](http://docs.google.com/java/net/URL.html) is the class representing a Universal Resource Locator, which is both an older concept for URIs and a means to access the resources.
* [URLConnection](http://docs.google.com/java/net/URLConnection.html) is created from a URL and is the communication link used to access the resource pointed by the URL. This abstract class will delegate most of the work to the underlying protocol handlers like http or ftp.
* [HttpURLConnection](http://docs.google.com/java/net/HttpURLConnection.html) is a subclass of URLConnection and provides some additional functionalities specific to the HTTP protocol.

The recommended usage is to use [URI](http://docs.google.com/java/net/URI.html) to identify resources, then convert it into a [URL](http://docs.google.com/java/net/URL.html) when it is time to access the resource. From that URL, you can either get the [URLConnection](http://docs.google.com/java/net/URLConnection.html) for fine control, or get directly the InputStream.

Here is an example:

URI uri = new URI("http://java.sun.com/");

URL url = uri.toURL();

InputStream in = url.openStream();

## Protocol Handlers

As mentioned, URL and URLConnection rely on protocol handlers which must be present, otherwise an Exception is thrown. This is the major difference with URIs which only identify resources, and therefore don't need to have access to the protocol handler. So, while it is possible to create an URI with any kind of protocol scheme (e.g. myproto://myhost.mydomain/resource/), a similar URL will try to instantiate the handler for the specified protocol; if it doesn't exist an exception will be thrown.

By default the protocol handlers are loaded dynamically from the default location. It is, however, possible to add to the search path by setting the java.protocol.handler.pkgs system property. For instance if it is set to myapp.protocols, then the URL code will try, in the case of http, first to load myapp.protocols.http.Handler, then, if this fails, http.Handler from the default location.

Note that the Handler class **has to** be a subclass of the abstract class [URLStreamHandler](http://docs.google.com/java/net/URLStreamHandler.html).

**Since:** JDK1.0

| | [**Overview**](http://docs.google.com/overview-summary.html) | **Package** | Class | [**Use**](http://docs.google.com/package-use.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV PACKAGE**](http://docs.google.com/java/math/package-summary.html)   [**NEXT PACKAGE**](http://docs.google.com/java/nio/package-summary.html) | [**FRAMES**](http://docs.google.com/index.html?java/net/package-summary.html)    [**NO FRAMES**](http://docs.google.com/package-summary.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).