Class java.net.ServerSocket

```
java.lang.Object
  |
  +--java.net.ServerSocket
```

public class **ServerSocket**
extends Object

This class implements server sockets. A server socket waits for requests to come in over the network. It performs some operation based on that request, and then possibly returns a result to the requester.

The actual work of the server socket is performed by an instance of the `SocketImpl` class. An application can change the socket factory that creates the socket implementation to configure itself to create sockets appropriate to the local firewall.

Since:
JDK1.0

See Also:
`SocketImpl`, `setSocketFactory(java.net.SocketImplFactory)`

Constructor Summary

ServerSocket(int port, int backlog, InetAddress bindAddr) Create a server with the specified port, listen backlog, and local IP address to bind to.
ServerSocket(int port, int backlog) Creates a server socket and binds it to the specified local port number.
ServerSocket(int port) Creates a server socket on a specified port.

Method Summary

Socket	**accept**() Listens for a connection to be made to this socket and accepts it.
void	**close**() Closes this socket.
InetAddress	**getInetAddress**() Returns the local address of this server socket.
int	**getLocalPort**() Returns the port on which this socket is listening.
int	**getSoTimeout**() Retrive setting for SO_TIMEOUT. 0 returns implies that the option is disabled (i.e.
protected void	**implAccept**(Socket s) Subclasses of ServerSocket use this method to override accept() to return their own subclass of socket.

static void	**setSocketFactory**(SocketImplFactory fac) Sets the server socket implementation factory for the application.
void	**setSoTimeout**(int timeout) Enable/disable SO_TIMEOUT with the specified timeout, in milliseconds.
String	**toString**() Returns the implementation address and implementation port of this socket as a `String`.

Methods inherited from class java.lang.Object
clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait , wait , wait

Constructor Detail

ServerSocket

```
public ServerSocket(int port)
             throws IOException
```

Creates a server socket on a specified port. A port of `0` creates a socket on any free port.

The maximum queue length for incoming connection indications (a request to connect) is set to `50`. If a connection indication arrives when the queue is full, the connection is refused.

If the application has specified a server socket factory, that factory's `createSocketImpl` method is called to create the actual socket implementation. Otherwise a "plain" socket is created.

Parameters:

`port` - the port number, or `0` to use any free port.

Throws:

IOException - if an I/O error occurs when opening the socket.

See Also:

`SocketImpl`, `SocketImplFactory.createSocketImpl()`, `setSocketFactory(java.net.SocketImplFactory)`

ServerSocket

```
public ServerSocket(int port,
                    int backlog)
             throws IOException
```

Creates a server socket and binds it to the specified local port number. A port number of `0` creates a socket on any free port.

The maximum queue length for incoming connection indications (a request to connect) is set to the `backlog` parameter. If a connection indication arrives when the queue is full, the connection is refused.

If the application has specified a server socket factory, that factory's `createSocketImpl` method is called to create the actual socket implementation. Otherwise a "plain" socket is created.

Parameters:

`port` - the specified port, or `0` to use any free port.

`backlog` - the maximum length of the queue.

Throws:

IOException - if an I/O error occurs when opening the socket.

See Also:

`SocketImpl`, `SocketImplFactory.createSocketImpl()`, `setSocketFactory(java.net.SocketImplFactory)`

ServerSocket

```
public ServerSocket(int port,
                    int backlog,
                    InetAddress bindAddr)
             throws IOException
```

Create a server with the specified port, listen backlog, and local IP address to bind to. The *bindAddr* argument can be used on a multi-homed host for a ServerSocket that will only accept connect requests to one of its addresses. If *bindAddr* is null, it will default accepting connections on any/all local addresses. The port must be between 0 and 65535, inclusive.

Parameters:

`port` - the local TCP port

`backlog` - the listen backlog

`bindAddr` - the local InetAddress the server will bind to

See Also:

`SocketConstants`, `SocketOption`, `SocketImpl`, `JDK1.1`

Method Detail

getInetAddress

```
public InetAddress getInetAddress()
```

Returns the local address of this server socket.

Returns:

the address to which this socket is connected, or `null` if the socket is not yet connected.

---

getLocalPort

```
public int getLocalPort()
```

Returns the port on which this socket is listening.

Returns:

the port number to which this socket is listening.

---

accept

```
public Socket accept()
              throws IOException
```

Listens for a connection to be made to this socket and accepts it. The method blocks until a connection is made.

Throws:

IOException - if an I/O error occurs when waiting for a connection.

---

implAccept

```
protected final void implAccept(Socket s)
                         throws IOException
```

Subclasses of ServerSocket use this method to override accept() to return their own subclass of socket. So a FooServerSocket will typically hand this method an *empty* FooSocket(). On return from implAccept the FooSocket will be connected to a client.

Since:
JDk1.1

---

close

```
public void close()
           throws IOException
```

Closes this socket.

Throws:
IOException - if an I/O error occurs when closing the socket.

---

setSoTimeout

```
public void setSoTimeout(int timeout)
                  throws SocketException
```

Enable/disable SO_TIMEOUT with the specified timeout, in milliseconds. With this option set to a non-zero timeout, a call to accept() for this ServerSocket will block for only this amount of time. If the timeout expires, a **java.io.InterruptedIOException** is raised, though the ServerSocket is still valid. The option **must** be enabled prior to entering the blocking operation to have effect. The timeout must be > 0. A timeout of zero is interpreted as an infinite timeout.

Since:
JDK1.1

---

getSoTimeout

```
public int getSoTimeout()
                 throws IOException
```

Retrive setting for SO_TIMEOUT. 0 returns implies that the option is disabled (i.e., timeout of infinity).

Since:
JDK1.1

---

toString

```
public String toString()
```

Returns the implementation address and implementation port of this socket as a `String`.

Returns:
a string representation of this socket.

Overrides:

toString in class Object

setSocketFactory

```
public static void setSocketFactory(SocketImplFactory fac)
                             throws IOException
```

Sets the server socket implementation factory for the application. The factory can be specified only once.

When an application creates a new server socket, the socket implementation factory's `createSocketImpl` method is called to create the actual socket implementation.

Parameters:

`fac` - the desired factory.

Throws:

IOException - if an I/O error occurs when setting the socket factory.

SocketException - if the factory has already been defined.

See Also:

`SocketImplFactory.createSocketImpl()`

Submit a bug or feature

Submit comments/suggestions about new javadoc look

This documentation was generated with a post-Beta4 version of Javadoc.