JAVA.LANG.SECURITYMANAGER.CHECKCONNECT METHOD

Description

The **java.lang.SecurityManager.checkConnect*Stringhost, intport*** method throws a SecurityException if the calling thread is not allowed to open a socket connection to the specified host and port number. A port number of -1 indicates that the calling method is attempting to determine the IP address of the specified host name.

This method calls checkPermission with the SocketPermission*host* + ":" + *port*, " *connect* " permission if the port is not equal to -1. If the port is equal to -1, then it calls checkPermission with the SocketPermission*host*, " *resolve* " permission. If you override this method, then you should make a call to super.checkConnect at the point the overridden method would normally throw an exception.

Declaration

Following is the declaration for **java.lang.SecurityManager.checkConnect** method

```
public void checkConnect(String host, int port)
```

Parameters

- **host** -- the host name port to connect to.
- **port** -- the protocol port to connect to.

Return Value

This method does not return a value.

Exception

- **SecurityException** -- if the calling thread does not have permission to open a socket connection to the specified host and port.
- **NullPointerException** -- if the host argument is null.

Example

Our examples require that the permissions for each command is blocked. A new policy file was set that allows only the creating and setting of our Security Manager. The file is in C:/java.policy and contains the following text:

```
grant {
  permission java.lang.RuntimePermission "setSecurityManager";
  permission java.lang.RuntimePermission "createSecurityManager";
  permission java.lang.RuntimePermission "usePolicy";
};
```

The following example shows the usage of lang.SecurityManager.checkConnect method.

```
package com.tutorialspoint;

public class SecurityManagerDemo {

   public static void main(String[] args) {

   // set the policy file as the system securuty policy
   System.setProperty("java.security.policy", "file:/C:/java.policy");

   // create a security manager
   SecurityManager sm = new SecurityManager();

   // set the system security manager
   System.setSecurityManager(sm);

   // perform the check
   sm.checkConnect("www.tutorialspoint.com", 8080);

   // print a message if we passed the check
   System.out.println("Allowed!");
   }
}
```

Let us compile and run the above program, this will produce the following result:

```
Exception in thread "main" java.security.AccessControlException: access denied
(java.net.SocketPermission www.tutorialspoint.com:8080 connect,resolve)
```

Loading [MathJax]/jax/output/HTML-CSS/jax.js