SECURITYMANAGER CHECKMULTICAST METHOD

Description

The **java.lang.SecurityManager.checkMulticast***InetAddressmaddr* method throws a SecurityException if the calling thread is not allowed to use *join*/*leave*/*send*/*receive* IP multicast. This method calls checkPermission with the java.net.SocketPermission*maddr*. *getHostAddress*(, "accept,connect") permission. If you override this method, then you should make a call to super.checkMulticast at the point the overridden method would normally throw an exception.

Declaration

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

```
public void checkMulticast(InetAddress maddr)
```

Parameters

- **maddr** -- Internet group address to be used.

Return Value

This method does not return a value.

Exception

- **SecurityException** -- if the caller does not have permission to access members.
- **NullPointerException** -- if the address 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.checkMulticast method.

```
package com.tutorialspoint;

import java.net.InetAddress;

public class SecurityManagerDemo extends SecurityManager {

   public static void main(String[] args) {

   InetAddress add = null;
   try {
   add = InetAddress.getLocalHost();
   } catch (Exception ex) {
   ex.printStackTrace();
   }

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

```
   // create a security manager
   SecurityManagerDemo sm = new SecurityManagerDemo();

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

   // perform the check
   sm.checkMulticast(add);

   // 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 192.168.1.6 connect,accept,resolve)
```