SECURITYMANAGER CHECKSECURITYACCESS METHOD

Description

The **java.lang.SecurityManager.checkSecurityAccess*Stringtarget*** method determines whether the permission with the specified permission target name should be granted or denied. If the requested permission is allowed, this method returns quietly. If denied, a SecurityException is raised. This method creates a SecurityPermission object for the given permission target name and calls checkPermission with it.

Declaration

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

```
public void checkSecurityAccess(String target)
```

Parameters

- **target** -- the target name of the SecurityPermission.

Return Value

This method does not return a value.

Exception

- **SecurityException** -- if the calling thread does not have permission for the requested access.
- **NullPointerException** -- if target is null.
- **IllegalArgumentException** -- if target is empty

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.checkSecurityAccess method.

```
package com.tutorialspoint;

public class SecurityManagerDemo extends SecurityManager {

   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
   SecurityManagerDemo sm = new SecurityManagerDemo();

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

```
   // perform the check
   sm.checkSecurityAccess("read");

   // 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.security.SecurityPermission read)
```