Table Of Content

Package gemom.authentication

Class Summary

AttributeHelper

The **AttributeHelper** class provides useful static methods for working with attribute values of an httpSession context.

AuthInterface

The **AuthInterface** class provides all the services a Service Provider (aka Relaying Party) has to use to perform the authentication process.

SessionStore

The **SessionStore** class provides a way to manage the httpSession context objects for RP internal purposes.

---

gemom.authentication

Class AttributeHelper

```
java.lang.Object
    |
    +--gemom.authentication.AttributeHelper
```

< Constructors > < Methods >

---

public class **AttributeHelper**
extends java.lang.Object

The **AttributeHelper** class provides useful static methods for working with attribute values of an httpSession context. The structure of the httpSession object is the follow:

- *ReturnedTokenType* - Specifies the type of returned security token (e.g. "urn:oasis:names:tc:SAML:1.0:assertion")
- *ReturnedTokenId* - Specifies a unique identifier (44 chars long) for the returned security token (e. g. "5tvOpBHYGpvZ6SC6Iu7mK/MSNq/1HSuvtBaW36RoZSs="). It is sometimes referred as PPID (Private Personal Identifier)
- *{Required Claims}* - Is the list of the required claims indicated in the Relying Party Authentication policy (i.e. "givenname", "surname", "emailaddress", etc.)
- *SessionId* - Specifies a unique identifier (20 chars long) for the httpSession context(e.g. "30kf7HAWq8Pg8jd0yAfa")

It is possible to access the content of the httpSession context with the following methods using the name of the attribute that you want to access. The httpSession context instance on which this class operates has to be obtained successfully completing the authentication process.

Version:

1.0

Author:

Leonardo Straniero TXT e-solution Spa

Constructors

AttributeHelper

```
public  AttributeHelper()
```

Methods

getAttributes

```
public static java.lang.String[] getAttributes(javax.servlet.http.HttpSession
session)
```

This method allows to acquire all the attributes' names present in the provided httpSession context.

Parameters:

session - The instance of httpSession context whose attributes' list is required.

Returns:

An Array of strings containing the attributes' names.

---

getMultiValuedAttribute

```
public static java.lang.String[]
getMultiValuedAttribute(javax.servlet.http.HttpSession session,
                                                    java.lang.String
attrname)
```

This method provides a way to read from the httpSession context a multivalues attribute.

Parameters:

session - The instance of httpSession context containing the user information you are looking for.
attrname - The name of the attribute whose values you are interested to.

Returns:

An Array of strings containing the values associated to the *attrname*

---

getSingleValuedAttribute

```
public static java.lang.Object
getSingleValuedAttribute(javax.servlet.http.HttpSession session,
                                                        java.lang.String
attrname)
```

This method allows to read the value of the httpSession context attribute whose name is specified by the second input parameter.

Parameters:

session - The instance of httpSession context containing the user information you are looking for.
attrname - The name of the attribute whose value you are interested to.

Returns:

An object containing the value of the requested attribute.

---

gemom.authentication

Class AuthInterface

```
java.lang.Object
    |
    +--gemom.authentication.AuthInterface
```

< Constructors > < Methods >

---

public class **AuthInterface**
extends java.lang.Object

The **AuthInterface** class provides all the services a Service Provider (aka Relaying Party) has to use to perform the authentication process. Methods are provided to recover the RP Authentication Policy and its Certificates Chain to be passed to the client side (Service Consumer) in order to request from the IdP an Authentication Token that meets RP specifications as detailed in the Authneitcation Policy. Additionally, this class provides the method to validate and properly process the authentication token creating and filling up an httpSession context instance with the provided user's claims. To validate and decrypt the authentication token this class makes use of the XMLSecurity library and a set of files containing the Authentication Policy and the Service Provider X.509 certificate. Please refer to the installation and configuration manual for more details.

Author:

Leonardo Straniero - TXT e-solution Spa

Version:

1.0

Constructors

AuthInterface

```
public  AuthInterface()
```

Methods

authenticate

```
public javax.servlet.http.HttpSession authenticate(java.lang.String
securityToken)
```

This method is the entry point to validate and process a received authentication token. It makes use of the XMLSecurty library to decrypt and check digital signatures. It uses the information contained in the keystore to perform these actions.

Parameters:

securityToken - A string, provided by the Service Comsumer process, with authentication token issued by the Security Token Service (STS).

Returns:

javax.servlet.http.HttpSession - an httpSession context instance containing the result of the authentication token validation. If successful the httpSession context instance contains all attributes and associated values as provided by the authentication token, as well as an internally generated *SessionID* that univocally identifies this instance.

---

getCertificates

```
public java.lang.String getCertificates()
```

This method acquires the certificates chain from the RP's keystore and serializes it as a suitable string.

Returns:

The string representing the RP's certificates chain.

---

getRP_policy

```
public java.lang.String getRP_policy()
```

This method acquires the RP's authentication policy as saved in the authentication context during the *init()* phase.

Returns:

The string representing the RP Authentication Policy (required claims, optional claims, token type, etc.)

---

init

`public void **init**()`

This method initializes the server side authentication library reading the configuration parameters from the configuration files. It acquires the Relaying Party's keystore location information and the authentication policy of the RP. Please refer to the installation and configuration manual for more details. This method sets up suitable internal instances of objects (e.g. the authenticaiton context) to be used by other methods to carry out the authentication procedure and to process the received authentication token.

---

invalidateSession

`public void **invalidateSession**(javax.servlet.http.HttpSession session)`

This method has to be used to destroy an httpSession context instance freeing all associated memory.

Parameters:

session - The HttpSession context instance to be removed.

---

gemom.authentication

Class SessionStore

```
java.lang.Object
    |
    +--gemom.authentication.SessionStore
```

< Constructors > < Methods >

---

public class **SessionStore**
extends java.lang.Object

The **SessionStore** class provides a way to manage the httpSession context objects for RP internal purposes. The goal of this class is, therefore, to provide to the RP main login with a way to store, search and manage all the active session contexts. This class provides methods to:

- store a new httpSession context instance into the "sessions pool";
- search the "sessions pool" to check if a specific httpSession context instance, as identified by its *SessionId*, exists;
- list all *SessionIds* present in the "sessions pool";
- get a copy of an httpSession context instance, as identified by its *SessionId*, so that you can apply on it the AttributeHelper methods;
- delete an httpSession context instance from the "sessions pool". This operation invalidates the instance and frees its memory.

Version:

1.0

Author:

Leonardo Straniero TXT e-solution Spa

Constructors

SessionStore

```
public  SessionStore()
```

Methods

addSession

```
public static synchronized boolean addSession(javax.servlet.http.HttpSession
session)
```

This method allows you to store an httpSession context in the "sessions pool".

Parameters:

session - The httpSession context instance to be added to the "sessions pool".

Returns:

If the store process has been completed successfully or not.

---

existSession

```
public static synchronized boolean existSession(java.lang.String sessionId)
```

This method checks if an httpSession context instance with the provided *SessionId* exists in the "session pool".

Parameters:

sessionId - - he identifier that univocally identifies the httpSession context to be checked.

Returns:

A *true*/*false* result.

---

getSession

```
public static synchronized javax.servlet.http.HttpSession
getSession(java.lang.String sessionId)
```

This method has to be used to retrieve a specific httpSession context instance, as identified by its *SessionId*, from the "session pool".

Parameters:

sessionId - - The identifier that univocally identifies the httpSession context to be retrieved.

Returns:

The specified httpSession context object, if such context exists in the "session pool", *null* otherwise.

---

getSessionsId

```
public static synchronized java.util.Set getSessionsId()
```

This method must be used to acquire the list of all httpSession context identifiers currently stored in the "session pool".

Returns:

The list of *sessionIds* currently held in the "session pool".

---

removeSession

```
public static synchronized boolean removeSession(java.lang.String sessionId)
```

This method has to be used to delete an httpSession context instance from the "sessions pool" and to invalidate and free its memory. If the provided *SessionId* does not match any sessionId in the "sessions pool" the method returns *false*, otherwise returns *true*.

Parameters:

sessionId - The unique identifier of the httpSession context to be removed.

Returns:

If the remove process has been completed successfully or not.

INDEX