ConnectedAppPlugin Class

Contains methods for extending the behavior of a connected app, for example, customizing how a connected app is invoked depending on the protocol used. This class gives you more control over the interaction between Salesforce and your connected app.

Namespace

Auth

Usage

When you create a connected app, you specify general information about the app and settings for OAuth, web apps, mobile apps, and canvas apps. To customize how the app is invoked, create a connected app handler with this `ConnectedAppPlugin` Apex class. For example, use this class to support new authentication protocols or respond to user attributes in a way that benefits a business process.

When you create a connected app handler, you also configure the `ConnectedAppPlugin` class to run as an execution user. The execution user authorizes access for the connected app. For example, when you use the `authorize` method, the execution user authorizes the connected app to access data.

If you don't specify an execution user, the plug-in runs as an Automated Process User, which is a system user that executes tasks behind the scenes. Most `ConnectedAppPlugin` methods require that you specify an execution user, with the exception of the `customAttributes` method. For more information, see Create a Custom Connected App Handler.

Example

This example authorizes the connected app user to use the connected app if the context is SAML and the user has reached the quota tracked in a custom field. It returns the user's permission set assignments. The example uses `Auth.InvocationContext` to modify a SAML assertion before it's sent to the service provider.

```
global class ConnectedAppPluginExample extends Auth.ConnectedAppPlugin
{
    // Authorize the app if the user has achieved quota tracked in a custom field
    global override Boolean authorize(Id userId, Id connectedAppId, Boolean isAdminApprove
    {
        // Create a custom boolean field HasAchievedQuota__c on the user record
        // and then uncomment the block below
        // User u = [select id, HasAchievedQuota__c from User where id =: userId].get(0);
        // return u.HasAchievedQuota__c;

        return isAdminApproved;
    }

    // Call a flow during refresh
    global override void refresh(Id userId, Id connectedAppId, Auth.InvocationContext cont
    {
       try
       {
          Map<String, Object> inputVariables = new Map<String, Object>();
```

```
            } catch ( Exception e ) {
               System.debug('FLOW Exception:' + e);
            }
        }

        // Return a user's permission set assignments
        global override Map<String,String> customAttributes(Id userId, Id connectedAppId, Map<
            formulaDefinedAttributes, Auth.InvocationContext context)
        {
            List<PermissionSetAssignment> psas = [SELECT id, PermissionSet.Name FROM Permissio
            WHERE PermissionSet.IsOwnedByProfile = false AND (AssigneeId = :userId)];
            String permsets = '[';
            for (PermissionSetAssignment psa :psas)
            {
                permsets += psa.PermissionSet.Name + ';';
            }
            permsets += ']';
            formulaDefinedAttributes.put('PermissionSets', permsets);
            return formulaDefinedAttributes;
        }
    }
```

- ConnectedAppPlugin Methods

ConnectedAppPlugin Methods

The following are methods for `ConnectedAppPlugin`.

- **authorize(userId, connectedAppId, isAdminApproved)**
  Deprecated and available only in API versions 35.0 and 36.0. As of version 37.0, use `authorize(userId, connectedAppId, isAdminApproved, context)` instead.
- **authorize(userId, connectedAppId, isAdminApproved, context)**
  Authorizes the specified user to access the connected app. If the connected app is set for users to self-authorize, this method isn't invoked.
- **customAttributes(userId, connectedAppId, formulaDefinedAttributes)**
  Deprecated and available only in API versions 35.0 and 36.0. As of version 37.0, use `customAttributes(userId, connectedAppId, formulaDefinedAttributes, context)` instead.
- **customAttributes(userId, connectedAppId, formulaDefinedAttributes, context)**
  Sets new attributes for the specified user. When the connected app gets the user's attributes from the UserInfo endpoint or through a SAML assertion, use this method to update the attribute values.
- **modifySAMLResponse(authSession, connectedAppId, samlResponse)**
  Modifies the XML generated by the Salesforce SAML Identity Provider (IDP) before it's sent to the service provider.
- **refresh(userId, connectedAppId)**
  Deprecated and available only in API versions 35.0 and 36.0. As of version 37.0, use `refresh(userId, connectedAppId, context)` instead.
- **refresh(userId, connectedAppId, context)**
  Salesforce calls this method during a refresh token exchange.

authorize(userId, connectedAppId, isAdminApproved)

Deprecated and available only in API versions 35.0 and 36.0. As of version 37.0, use `authorize(userId, connectedAppId, isAdminApproved, context)` instead.

Signature

```
public Boolean authorize(Id userId, Id connectedAppId, Boolean isAdminApproved)
```

Parameters

connectedAppId

Type: String

The 15-character ID of the connected app.

isAdminApproved

Type: Boolean

The approval state of the specified user when the connected app requires approval.

Return Value

Type: Boolean

If the connected app requires admin approval, a returned value of `true` indicates that the current user is approved.

authorize(userId, connectedAppId, isAdminApproved, context)

Authorizes the specified user to access the connected app. If the connected app is set for users to self-authorize, this method isn't invoked.

Signature

```
public Boolean authorize(Id userId, Id connectedAppId, Boolean isAdminApproved,
Auth.InvocationContext context)
```

Parameters

userId

Type: Id

The 15-character ID of the user attempting to use the connected app.

connectedAppId

Type: Id

The 15-character ID of the connected app.

isAdminApproved

Type: Boolean

The approval state of the specified user when the connected app requires approval.

context

Type: InvocationContext

The context in which the connected app is invoked.

Return Value

Type: Boolean

If the connected app requires admin approval, a returned value of `true` indicates that the user is approved.

Usage

`ConnectedAppPlugin` runs on behalf of the current user. But the user must have permission to use the connected app for the plug-in to work. Use this method to authorize the user.

customAttributes(userId, connectedAppId, formulaDefinedAttributes, context) instead.

Signature

```
public Map<String,String> customAttributes(Id userId, Id connectedAppId, Map<String,String>
formulaDefinedAttributes,)
```

Parameters

userId

Type: Id

The 15-character ID of the user attempting to use the connected app.

connectedAppId

Type: Id

The 15-character ID of the connected app.

formulaDefinedAttributes

Type: Map<String,String>

A map of the new set of attributes from the UserInfo endpoint (OAuth) or from a SAML assertion. For more information, see The UserInfo Endpoint in the online help.

Return Value

Type: Map<String,String>

A map of the updated set of attributes.

customAttributes(userId, connectedAppId, formulaDefinedAttributes, context)

Sets new attributes for the specified user. When the connected app gets the user's attributes from the UserInfo endpoint or through a SAML assertion, use this method to update the attribute values.

Signature

```
public Map<String,String> customAttributes(Id userId, Id connectedAppId, Map<String,String>
formulaDefinedAttributes, Auth.InvocationContext context)
```

Parameters

userId

Type: Id

The 15-character ID of the user attempting to use the connected app.

connectedAppId

Type: Id

The 15-character ID for the connected app.

formulaDefinedAttributes

Type: Map<String,String>

A map of the current set of attributes from the UserInfo endpoint (OAuth) or from a SAML assertion. For more information, see The UserInfo Endpoint in the online help.

context

Type: InvocationContext

A map of the updated set of attributes.

modifySAMLResponse(authSession, connectedAppId, samlResponse)

Modifies the XML generated by the Salesforce SAML Identity Provider (IDP) before it's sent to the service provider.

Signature

```
public dom.XmlNode modifySAMLResponse(Map<String,String> authSession, Id connectedAppId,
dom.XmlNode samlResponse)
```

Parameters

authSession

Type: Map<String,String>

The attributes for the authorized user's session. The map includes the 15-character ID of the authorized user who's accessing the connected app.

connectedAppId

Type: Id

The 15-character ID of the connected app.

samlResponse

Type: Dom.XmlNode

Contains the SAML XML response generated by the IDP.

Return Value

Type: Dom.XmlNode

Returns an instance of Dom.XmlNode containing the modified SAML XML response.

Usage

Use this method to modify the XML SAML response to perform an action based on the context of the SAML request before it's verified, signed, and sent to the target service provider. This method enables developers to extend the connected app plug-in to meet their specific needs.

The developer assumes full responsibility for changes made within the connected app plug-in. The plug-in must include validation and error handling. If the plug-in throws an exception, catch it, log it, and stop the process. Don't send anything to the target service provider.

refresh(userId, connectedAppId)

Deprecated and available only in API versions 35.0 and 36.0. As of version 37.0, use `refresh(userId, connectedAppId, context)` instead.

Signature

```
public void refresh(Id userId, Id connectedAppId)
```

Parameters

userId

Type: Id

The 15-character ID of the user requesting the refresh token.

Return Value

Type: void

refresh(userId, connectedAppId, context)

Salesforce calls this method during a refresh token exchange.

Signature

```
public void refresh(Id userId, Id connectedAppId, Auth.InvocationContext context)
```

Parameters

userId

Type: Id

The 15-character ID of the user requesting the refresh token.

connectedAppId

Type: Id

The 15-character ID of the connected app.

context

Type: InvocationContext

The context in which the connected app is invoked.

Return Value

Type: void

DID THIS ARTICLE SOLVE YOUR ISSUE?
Let us know so we can improve!

Share your feedback

DEVELOPER CENTERS
Heroku
MuleSoft
Tableau
Commerce Cloud
Lightning Design System
Einstein
Quip

POPULAR RESOURCES
Documentation
Component Library
APIs
Trailhead
Sample Apps
Podcasts
AppExchange

COMMUNITY
Trailblazer Community
Events and Calendar
Partner Community
Blog
Salesforce Admins
Salesforce Architects