Auth Namespace

The `Auth` namespace provides an interface and classes for single sign-on into Salesforce and session security management.

The following is the interface in the `Auth` namespace.

- AuthConfiguration Class
  Contains methods for configuring settings for users to log in to a Salesforce org using their authentication provider credentials instead of their Salesforce credentials. The authentication provider can be any authentication provider that supports the OpenID Connect protocol, such as Google, Facebook, or Twitter. Users log in to either an Experience Cloud site (`https://MyDomainName.my.site.com`) or your My Domain login URL (`https://MyDomainName.my.salesforce.com`).
- AuthProviderCallbackState Class
  Provides request HTTP headers, body, and query parameters to the `AuthProviderPlugin.handleCallback` method for user authentication. This class allows you to group the information passed in rather than passing headers, body, and query parameters individually.
- AuthProviderPlugin Interface
  This interface is deprecated. For new development, use the abstract class `Auth.AuthProviderPluginClass` to create a custom OAuth-based authentication provider plug-in for single sign-on in to Salesforce.
- AuthProviderPluginClass Class
  Contains methods to create a custom OAuth-based authentication provider plug-in for single sign-on in to Salesforce. Use this class to create a custom authentication provider plug-in if you can’t use one of the authentication providers that Salesforce provides.
- AuthProviderTokenResponse Class
  Stores the response from the `AuthProviderPlugin.handleCallback` method.
- AuthToken Class
  Contains methods for getting and revoking access and refresh tokens that are issued when a user logs in via a single sign-on (SSO) flow that uses an authentication provider, such as Facebook.
- CommunitiesUtil Class
  Contains methods for getting information about an Experience Cloud user.
- ConfigurableSelfRegHandler Interface
  Gives you more control over how customers or partners self-register for your Experience Cloud site by creating a class that implements `Auth.ConfigurableSelfRegHandler`. You choose the user information to collect, and how users identify themselves–with their email address, phone number, or another identifier. When verified, you create a customer or partner user and log in the user to your Experience Cloud site.
- ConfirmUserRegistrationHandler Interface
  Manages single sign-on (SSO) user mappings between Salesforce and a third-party identity provider. Use this interface to confirm user mappings before updating them.
- 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.
- CustomOneTimePasswordDeliveryHandler Interface
  To use a custom SMS provider to send one-time passwords (OTPs) for Experience Cloud identity verification, create a class that implements the `Auth.CustomOneTimePasswordDeliveryHandler` interface.

Contains methods for extending the behavior of an external client app. For example, customize how an external client app is invoked depending on the protocol used. This class gives you more control over the interaction between Salesforce and your external client app.

- **GeneratedUserData Class**
  Stores the output of the Generate User Data invocable action, which you can access in Flow Builder.
- **HeadlessSelfRegistrationHandler Interface**
  Creates customer and partner users during the Headless Registration Flow.
- **HeadlessUserDiscoveryHandler Interface**
  Use this interface to create a headless user discovery handler that you implement during headless login, passwordless login, and forgot password flows.
- **HeadlessUserDiscoveryResponse Class**
  Contains methods to describe the result of headless user discovery using a handler that implements the `Auth.HeadlessUserDiscoveryHandler` interface during headless login, passwordless login, and forgot password flows.
- **HttpCalloutMockUtil Class**
  Contains a method to send fake HTTP callouts for classes in the `Auth` namespace.
- **IntegratingAppType Enum**
  Specifies whether you're integrating your app as a connected app or as an external client app in methods used in your customized Apex token exchange handler, which extends the `Auth.Oauth2TokenExchangeHandler` class.
- **InvocationContext Enum**
  The context in which the connected app is invoked, such as the protocol flow used and the token type issued, if any. Developers can use the context information to write code that is unique to the type of invocation.
- **JsonValueOutput Class**
  Stores the output of the Get User Data from JSON String invocable action, which you can access in Flow Builder..
- **JWS Class**
  Contains methods that apply a digital signature to a JSON Web Token (JWT), using a JSON Web Signature (JWS) data structure. This class creates the signed JWT bearer token, which can be used to request an OAuth access token in the OAuth 2.0 JWT bearer token flow.
- **JWT Class**
  Generates the JSON Claims Set in a JSON Web Token (JWT). The resulting Base64-encoded payload can be passed as an argument to create an instance of the `Auth.JWS` class.
- **JWTBearerTokenExchange Class**
  Contains methods that POST the signed JWT bearer token to a token endpoint to request an access token, in the OAuth 2.0 JWT bearer token flow.
- **JWTUtil Class**
  Contains methods for validating a JSON Web Token (JWT) from an external identity provider as part of the OAuth 2.0 token exchange flow. Use these methods as part of the `validateIncomingToken` method in the `Auth.Oauth2TokenExchangeHandler` class.
- **LightningLoginEligibility Enum**
  Contains a Lightning Login eligibility value used by the `Auth.SessionManagement.getLightningLoginEligibility` method.
- **LoginDiscoveryHandler Interface**
  Salesforce gives you the ability to log in users based on other verification methods than username and password. For example, it can prompt users to log in with their email, phone number, or another identifier like a Federation ID or device identifier. Login Discovery is available to these licenses: Customer Community, Customer Community Plus, External Identity, Partner Community, and Partner Community Plus.
- **LoginDiscoveryMethod Enum**
  Contains methods used to verify the user's identity when the My Domain login process

such as an email address or phone number. Then the handler determines (discovers) how to authenticate the user. Either the user enters a password or is directed to an identity provider's login page.

- **Oauth2TokenExchangeHandler Class**
  Use this class to create a token exchange handler that validates tokens from an external identity provider and maps the token's subject to a Salesforce user during the OAuth 2.0 token exchange flow. The handler can also be used to create users by setting up a new User object and returning it to Salesforce for automatic insertion.
- **OAuth2TokenExchangeType Enum**
  Used during the OAuth 2.0 token exchange flow to specify the type of token that's being exchanged for a Salesforce token.
- **OAuthRefreshResult Class**
  Stores the result of an `AuthProviderPluginClass` refresh method. OAuth authentication flow provides a refresh token that can be used to get a new access token. Access tokens have a limited lifetime as specified by the session timeout value. When an access token expires, use a refresh token to get a new access token.
- **OauthToken Class**
  Contains a method to revoke OAuth access tokens and refresh tokens. This method supports opaque tokens and JSON Web Token (JWT)-based access tokens, including guest and named user JWT-based access tokens.
- **OauthTokenType Enum**
  Specifies the type of Salesforce-issued OAuth 2.0 token being revoked in the `OauthToken.revokeToken` method.
- **RegistrationHandler Interface**
  Salesforce provides the ability to use an authentication provider, such as Facebook© or Janrain©, for single sign-on into Salesforce.
- **SamlJitHandler Interface**
  Use this interface to control and customize Just-in-Time user provisioning logic during SAML single sign-on.
- **SessionManagement Class**
  Contains methods for verifying users' identity, creating custom login flows, customizing security levels, and defining trusted IP ranges for a current session.
- **SessionLevel Enum**
  An `Auth.SessionLevel` enum value is used by the `SessionManagement.setSessionLevel` method.
- **TokenValidationResult Class**
  Contains methods that describe the result of the token validation performed by a token exchange handler using the `validateIncomingToken` method in the `Auth.Oauth2TokenExchangeHandler` class during the OAuth 2.0 token exchange flow.
- **UserData Class**
  Stores user information for authentication provider registration handlers, including handlers that implement the `Auth.RegistrationHandler` interface and handlers built using Flow Builder.
- **VerificationAction Enum**
  Indicates the method that you use to send a one-time password (OTP) to a user during the headless passwordless login flow.
- **VerificationMethod Enum**
  Contains the different ways users can identify themselves when logging in. You can use it to implement mobile-centric passwordless login pages and to self-register (and deregister) verification methods.
- **VerificationPolicy Enum**
  The `Auth.VerificationPolicy` enum contains an identity verification policy value used by the `SessionManagement.generateVerificationUrl` method.
- **VerificationResult Class**
  Contains the result of a verification challenge that you invoke when you create your own Verify page. The challenge can be initiated by either the

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

Privacy Information Terms of Service Legal Use of Cookies Trust Cookie Preferences

Your Privacy Choices Responsible Disclosure Contact