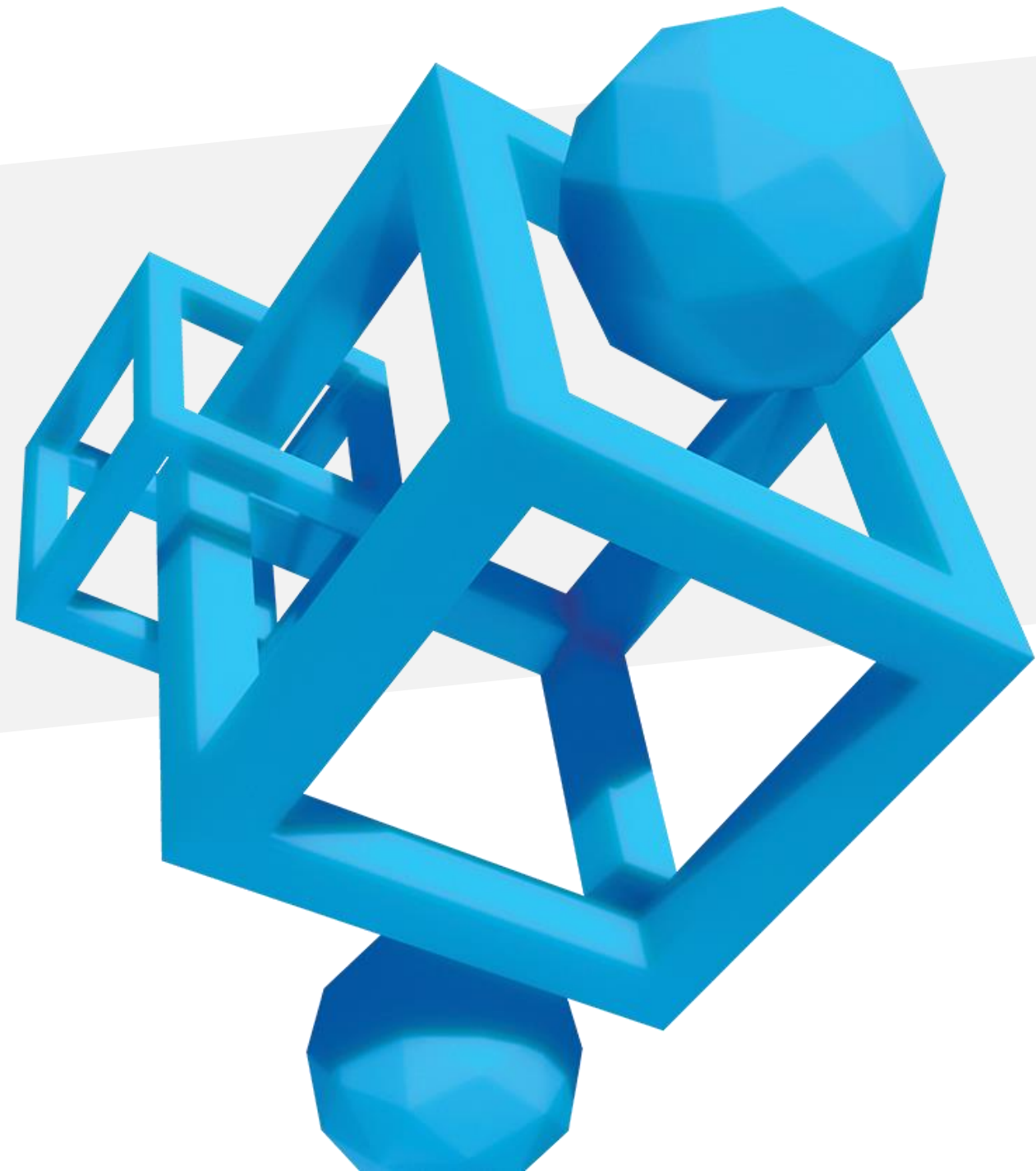


Seamless OAuth2.0 and OpenID Connect in VAST

Johan Brichau

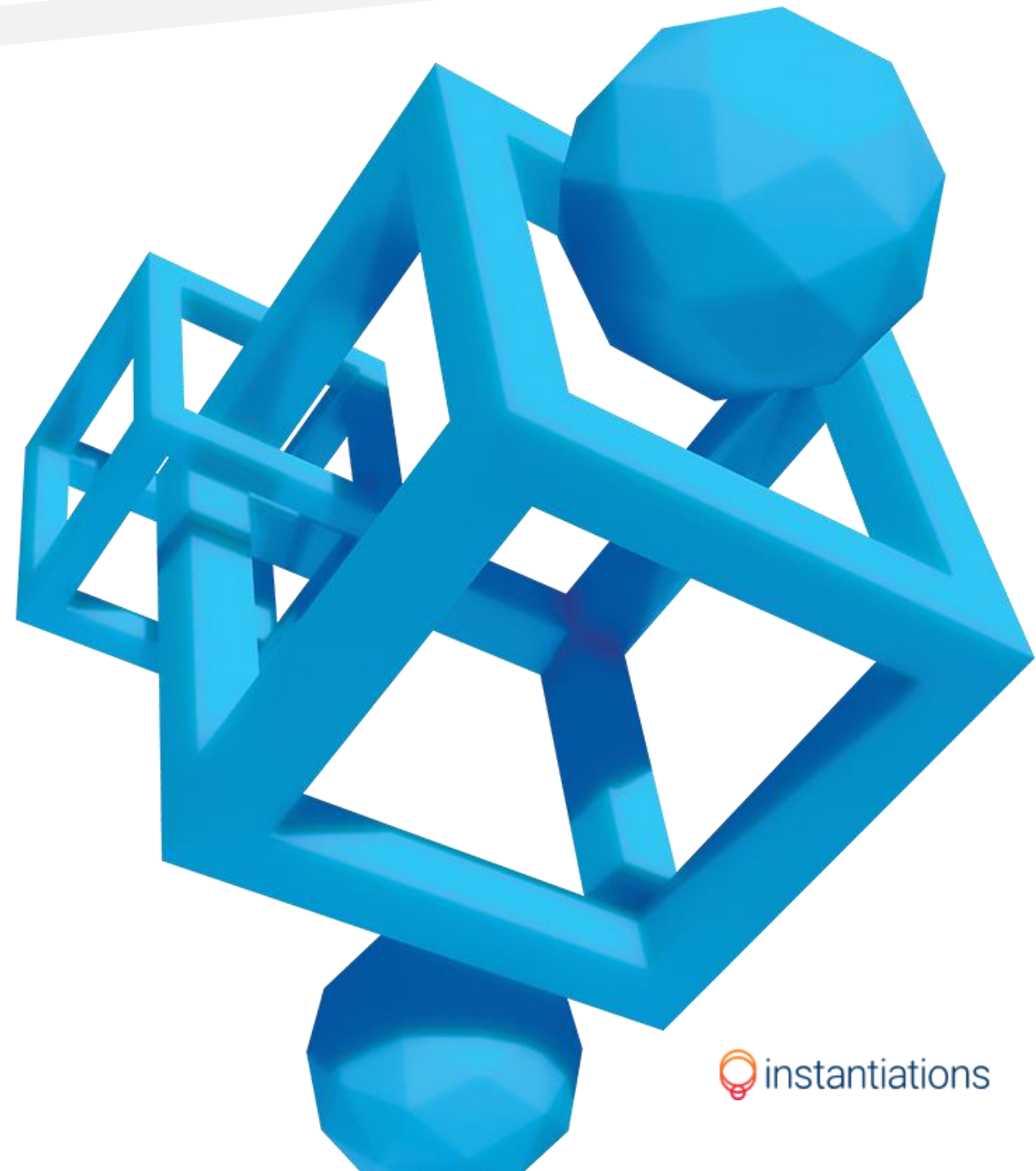
VAST Consultant and Senior Software Engineer

✉ jbrichau@instantiations.com



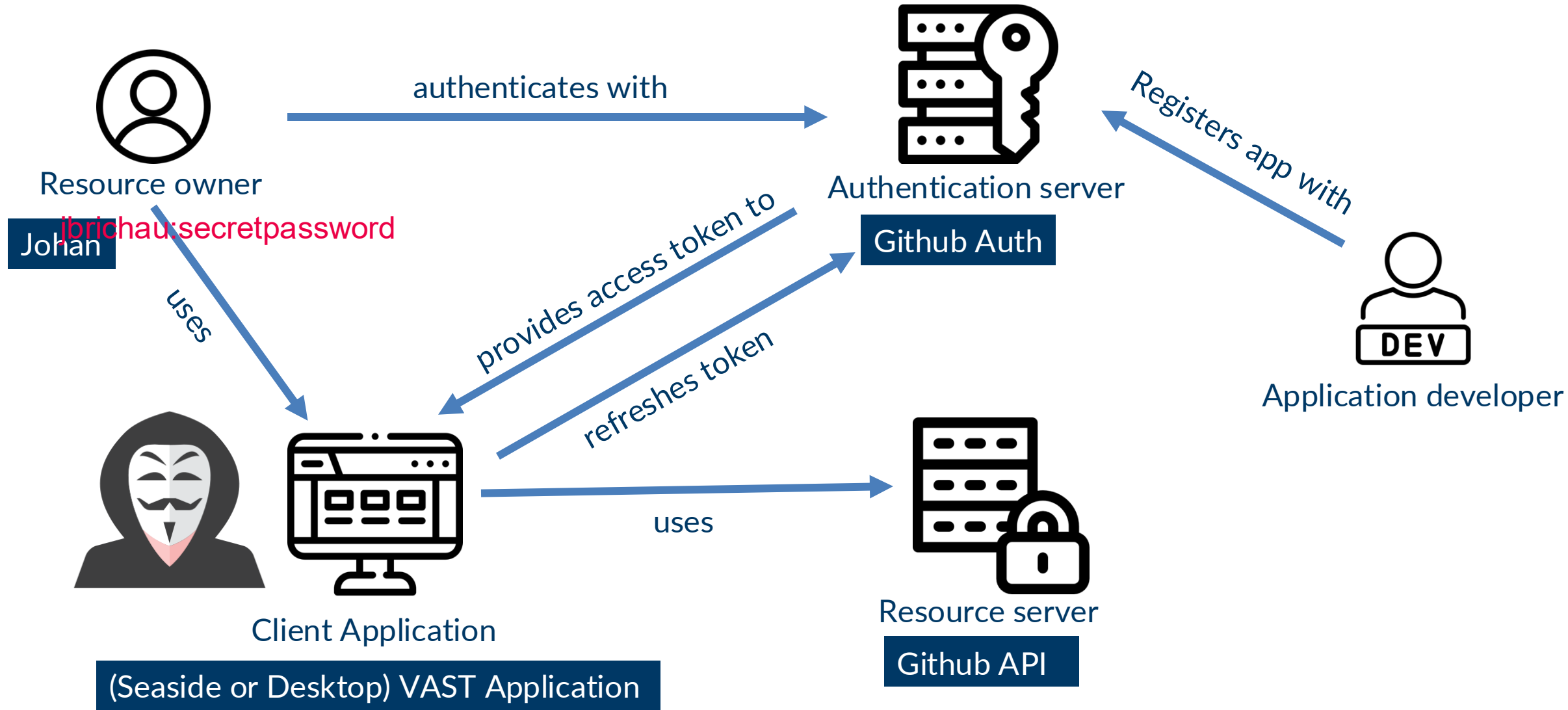
Agenda

- What is OAuth2.0 ?
- Demo - Seaside application flow with Github
- OAuth2.0 client code example
- What is OpenID Connect (OIDC) ?
- Demo – Seaside and Desktop application flow with Google/Microsoft
- Convenience with VAST Async framework
- Json Web Token
- Final Remarks



OAuth2.0

What is OAuth2.0 ?



What is OAuth2.0 ?

- Industry-standard **authorization** protocol
 - Access a protected (web) service
- Defines authorization through **delegation**
- Works over **HTTPS**
- **Framework** + extensions

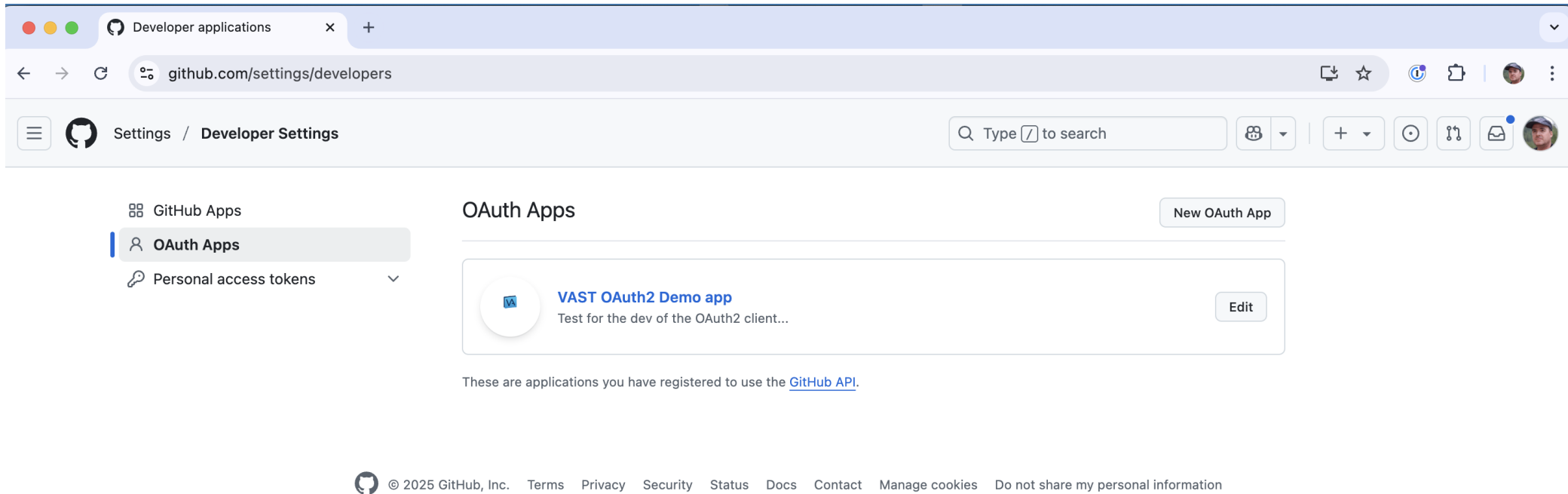


User can authorize an application to use a service without sharing his/her credentials with that application.



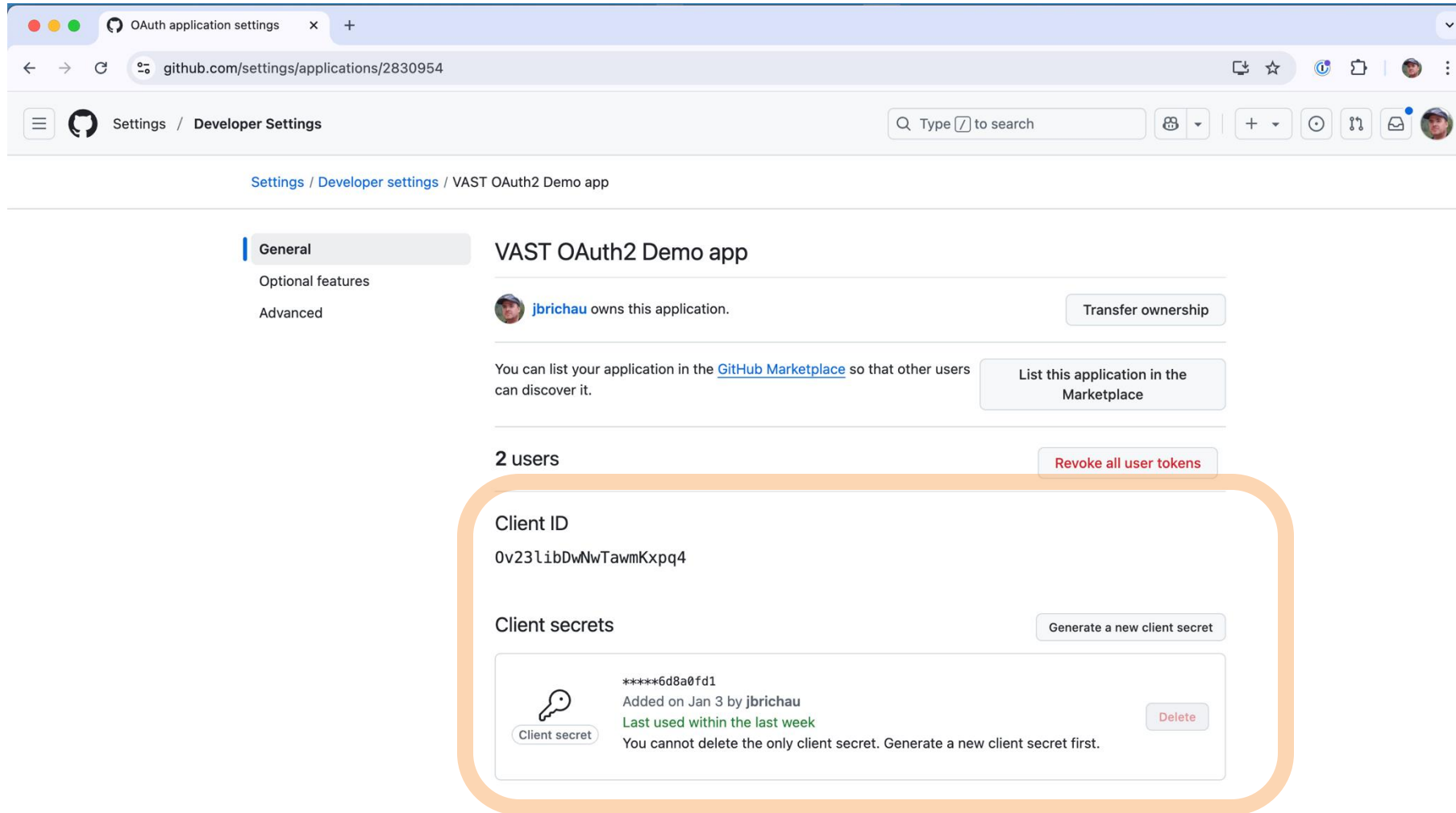
Demo: Github OAuth Application

- Register client application with the OAuth2.0 provider (Github)



Demo: Github OAuth Application

- Retrieve **Client ID** and **Client secret**



The screenshot shows the Github OAuth application settings page for an application named 'VAST OAuth2 Demo app'. The page is viewed in a browser window with the address bar showing 'github.com/settings/applications/2830954'. The left sidebar contains a 'General' tab and links for 'Optional features' and 'Advanced'. The main content area shows the application name, owner 'jbrichau', and a 'Transfer ownership' button. Below this, there is a section for '2 users' with a 'Revoke all user tokens' button. The 'Client ID' is displayed as '0v23libDwNwTawmKxpq4'. The 'Client secrets' section shows a single secret '*****6d8a0fd1' added on Jan 3 by jbrichau, with a 'Delete' button. A message states: 'You cannot delete the only client secret. Generate a new client secret first.' The entire 'Client ID' and 'Client secrets' section is highlighted with an orange rounded rectangle.

OAuth application settings x +

github.com/settings/applications/2830954

Settings / Developer Settings

Settings / Developer settings / VAST OAuth2 Demo app

General

Optional features

Advanced

VAST OAuth2 Demo app

jbrichau owns this application. [Transfer ownership](#)


You can list your application in the [GitHub Marketplace](#) so that other users can discover it. [List this application in the Marketplace](#)

2 users [Revoke all user tokens](#)

Client ID

0v23libDwNwTawmKxpq4

Client secrets [Generate a new client secret](#)

 *****6d8a0fd1
Added on Jan 3 by jbrichau
Last used within the last week
[Delete](#)

Client secret

You cannot delete the only client secret. Generate a new client secret first.

Demo: Github OAuth Application

- Define the application's **callback url**

The full URL to your application homepage.

Application description

Test for the dev of the OAuth2
client in VAST

This is displayed to all users of your application.

Authorization callback URL *

http://localhost:8888

Your application's callback URL. Read our [OAuth documentation](#) for more information.

Demo: Github OAuth2.0 Seaside application

The screenshot shows a web browser window with the title "OAuth2 & OIDC demo app". The address bar shows the URL "localhost:8888/examples/oauth2oidc?_s=zHFpYItVIGUffmRD&_k=UMSImlgTLMtFfney". The page has a blue header with the title "OAuth2 & OIDC demo app" and links to "Google", "Microsoft", and "Github".

The main content area features the "Github" logo and the text "OAuth2". Below this, it says "To learn about your client id, secret and other parameters, see: <https://docs.github.com/en/apps/oauth-apps/building-oauth-apps/creating-an-oauth-app>".

The interface is divided into three main sections:

- Authentication:** Contains fields for "Client ID" (Ov23libDwNwTawmIKxppq4) and "Client Secret" (a masked field).
- Authorization:** Contains a "Scope" field.
- Endpoints:** Contains fields for "Authorize URL" (https://github.com/login/oauth/authorize) and "Access Token URL" (https://github.com/login/oauth/access_token).

On the right side, there is a section titled "My github repositories:" with a list of repositories:

- Adafruit-Motor-HAT-NodeJs-Library
- AdventOfCode
- backbone-couchdb
- BackboneCounter
- Booklet-uFFI
- busytoilet
- CameraBuddy
- docker-pharo-runtime
- docker-pharo-vm
- Door-Actuator
- esug.github.io

A large blue "Live" watermark is overlaid on the center of the image.

At the bottom of the browser window, a status bar shows "New Session", "Configure", "Halos", "Deprecated (1)", and "359/875 ms".

OAuth2.0 client in VAST

```
14.1.0arm64-b577-OAuth2OIDC-ESUG: Workspace
File Edit View Options Encoding Language

2 grant := SstOAuth2AuthorizationCodeGrant
3   newWithClientId: '12345679'
4   authorizationBaseUrl: 'https://oauth2-service.com/authorize' sstAsUrl
5   tokenUrl: 'https://oauth2-service.com/accesstoken' sstAsUrl.
6 grant
7   clientSecret: 'abcdefghk1125-02';
8   scope: 'api-access'.
9 authorizationUrl := grant
10   authorizationUrlWithRedirectTo: 'https://myapplication.com/authorize-callback' sstAsUrl
11   state: 'random state string'.
12 "Fictitious method; doing redirect depends on application implementation. See further."
13 self
14   redirectTo: authorizationUrl
15   handleCallbackAt: 'https://myapplication.com/authorize-callback' sstAsUrl
16   with: [ :authorizeResultRequest |
17     httpClientWithOAuth2 := grant createClientFromAuthorizationResponseData: authorizeResultRequest queryFields ]
18 "Use client to access protected resource."
19 httpClientWithOAuth2 get: 'https://api-service.com/protectedinfo'
```

OAuth2.0 client in VAST

```
14.1.0arm64-b577-OAuth2OIDC-ESUG: Workspace
File Edit View Options Encoding Language

2 grant := SstOAuth2AuthorizationCodeGrant
3   newWithClientId: '12345679'
4   authorizationBaseUrl: 'https://oauth2-service.com/authorize' sstAsUrl
5   tokenUrl: 'https://oauth2-service.com/accesstoken' sstAsUrl.
6 grant
7   clientSecret: 'abcdefghk1125-02';
8   scope: 'api-access'.
9 authorizationUrl := grant
10   authorizationUrlWithRedirectTo: 'https://myapplication.com/authorize-callback' sstAsUrl
11   state: 'random state string'.
12 "Fictitious method; doing redirect depends on application implementation. See further."
13 self
14   redirectTo: authorizationUrl
15   handleCallbackAt: 'https://myapplication.com/authorize-callback' sstAsUrl
16   with: [ :authorizeResultRequest |
17     httpClientWithOAuth2 := grant createClientFromAuthorizationResponseData: authorizeResultRequest queryFields ]
18 "Use client to access protected resource."
19 httpClientWithOAuth2 get: 'https://api-service.com/protectedinfo'
```

OAuth2.0 client in VAST

```
14.1.0arm64-b577-OAuth2OIDC-ESUG: Workspace
File Edit View Options Encoding Language

2 grant := SstOAuth2AuthorizationCodeGrant
3   newWithClientId: '12345679'
4   authorizationBaseUrl: 'https://oauth2-service.com/authorize' sstAsUrl
5   tokenUrl: 'https://oauth2-service.com/accesstoken' sstAsUrl.
6 grant
7   clientSecret: 'abcdefghk1125-02';
8   scope: 'api-access'.
9 authorizationUrl := grant
10   authorizationUrlWithRedirectTo: 'https://myapplication.com/authorize-callback' sstAsUrl
11   state: 'random state string'.
12 "Fictitious method; doing redirect depends on application implementation. See further."
13 self
14   redirectTo: authorizationUrl
15   handleCallbackAt: 'https://myapplication.com/authorize-callback' sstAsUrl
16   with: [ :authorizeResultRequest |
17     httpClientWithOAuth2 := grant createClientFromAuthorizationResponseData: authorizeResultRequest queryFields ]
18 "Use client to access protected resource."
19 httpClientWithOAuth2 get: 'https://api-service.com/protectedinfo'
```

OAuth2.0 client in VAST

```
14.1.0arm64-b577-OAuth2OIDC-ESUG: Workspace
File Edit View Options Encoding Language

2 grant := SstOAuth2AuthorizationCodeGrant
3   newWithClientId: '12345679'
4   authorizationBaseUrl: 'https://oauth2-service.com/authorize' sstAsUrl
5   tokenUrl: 'https://oauth2-service.com/accesstoken' sstAsUrl.
6 grant
7   clientSecret: 'abcdefghk1125-02';
8   scope: 'api-access'.
9 authorizationUrl := grant
10   authorizationUrlWithRedirectTo: 'https://myapplication.com/authorize-callback' sstAsUrl
11   state: 'random state string'.
12 "Fictitious method; doing redirect depends on application implementation. See further."
13 self
14   redirectTo: authorizationUrl
15   handleCallbackAt: 'https://myapplication.com/authorize-callback' sstAsUrl
16   with: [ :authorizeResultRequest |
17     httpClientWithOAuth2 := grant createClientFromAuthorizationResponseData: authorizeResultRequest queryFields ]
18 "Use client to access protected resource."
19 httpClientWithOAuth2 get: 'https://api-service.com/protectedinfo'
```

OAuth2.0 redirection in Seaside (1)

14.1.0arm64-b577-OAuth2OIDC-ESUG: SstOAuth2AndOIDCExamplesApp(17/06/2025 10:43:29) Browser: SstOAuth2AndOIDCSeasideExample> initiateAuthorizationOn:

File Edit Classes Methods Info Categories Options Breakpoints

OAuth2AndOIDCSeasideExample -- all --

- WAPainter
- WAPresenter
- WAComponent
- SstOAuth2AndOIDCSeaside**
- SubApplication

- Actions
- hooks
- Initialization
- Private
- Rendering

Public Private All

- googleConfiguration
- handleAuthorizationRedirect:
- initialize
- initializeOIDCProviders
- initialRequest:
- initiateAuthorizationOn:**
- microsoftConfiguration

Public Private Instance Class

Class Definition Method Source Method Comment Method Notes

initiateAuthorizationOn: [html](#)

| [redirectUrl](#) |

"Create a Seaside callback sitting at <redirectUrl> that will handle the redirect from the authorization service."

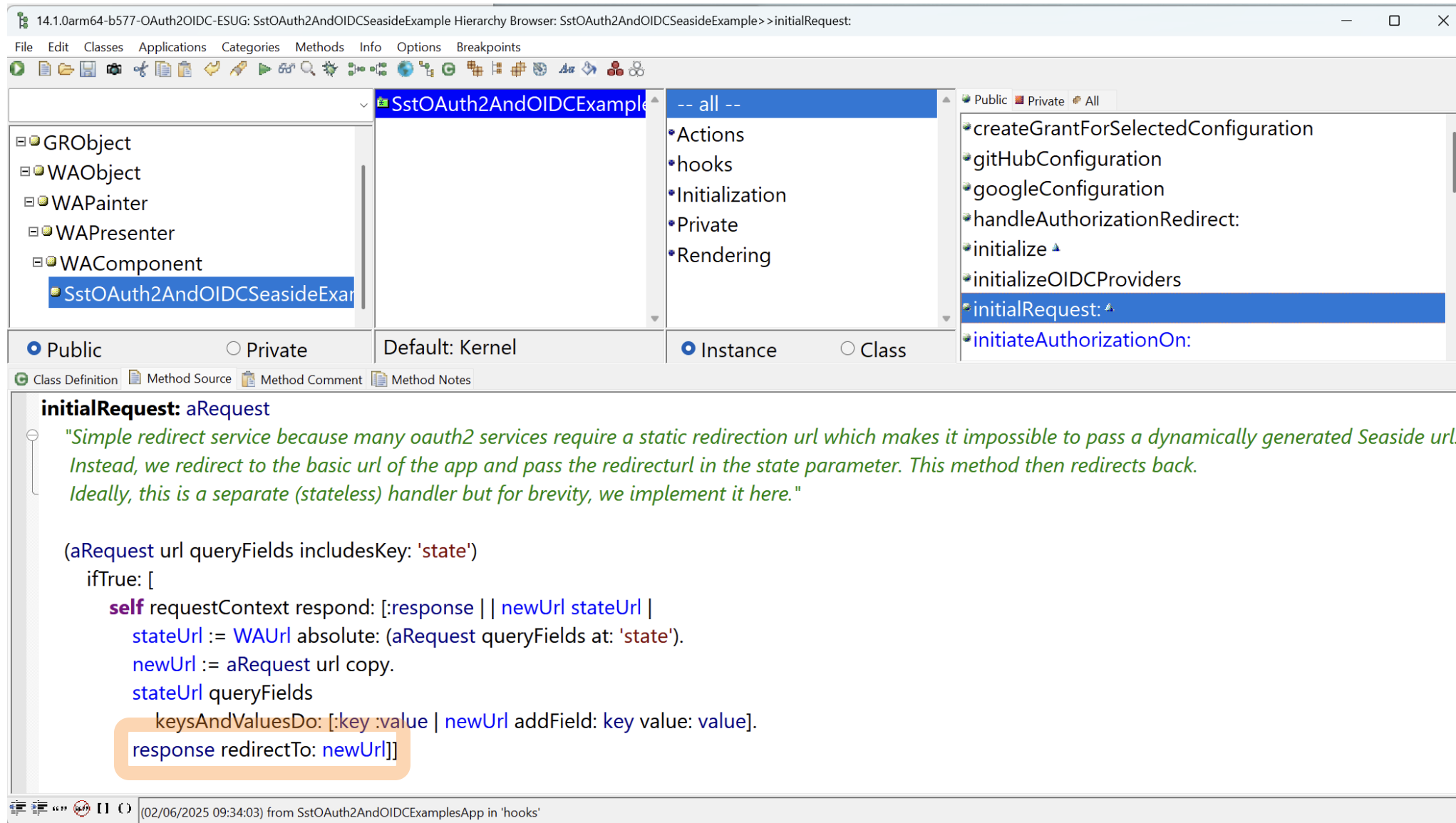
[redirectUrl](#) := html urlForAction: [**self** handleAuthorizationRedirect: **self** requestContext].

grant := **self** createGrantForSelectedConfiguration.

self requestContext redirectTo: (**grant** authorizationUrlWithRedirectTo: **self** application url asString sstAsUrl state: [redirectUrl](#) asString)

(17/06/2025 10:50:12) from SstOAuth2AndOIDCExamplesApp in 'Actions'

OAuth2.0 redirection in Seaside (2)



The screenshot shows the Squeak IDE interface. The top menu bar includes File, Edit, Classes, Applications, Categories, Methods, Info, Options, and Breakpoints. The left pane shows a class hierarchy with SstOAuth2AndOIDCSeasideExample selected. The middle pane shows a list of methods, with 'initialRequest:' highlighted. The right pane shows the source code of the 'initialRequest:' method. The status bar at the bottom indicates the current location is in 'hooks'.

14.1.0arm64-b577-OAuth2OIDC-ESUG: SstOAuth2AndOIDCSeasideExample Hierarchy Browser: SstOAuth2AndOIDCSeasideExample>>initialRequest:

File Edit Classes Applications Categories Methods Info Options Breakpoints

SstOAuth2AndOIDCSeasideExample

- GRObjct
- WAOBJct
- WAPainter
- WAPresenter
- WAComponent
 - SstOAuth2AndOIDCSeasideExample

Public Private All

- createGrantForSelectedConfiguration
- gitHubConfiguration
- googleConfiguration
- handleAuthorizationRedirect:
- initialize
- initializeOIDCProviders
- initialRequest:
- initiateAuthorizationOn:

Public Private

Default: Kernel

Instance Class

Class Definition Method Source Method Comment Method Notes

initialRequest: aRequest

"Simple redirect service because many oauth2 services require a static redirection url which makes it impossible to pass a dynamically generated Seaside url. Instead, we redirect to the basic url of the app and pass the redirecturl in the state parameter. This method then redirects back. Ideally, this is a separate (stateless) handler but for brevity, we implement it here."

(aRequest url queryFields includesKey: 'state')

ifTrue: [
 self requestContext respond: [:response || newUrl stateUrl |
 stateUrl := WAUrl absolute: (aRequest queryFields at: 'state').
 newUrl := aRequest url copy.
 stateUrl queryFields
 keysAndValuesDo: [:key :value | newUrl addField: key value: value].
 response redirectTo: newUrl]]

(02/06/2025 09:34:03) from SstOAuth2AndOIDCExamplesApp in 'hooks'

OAuth2.0 is a framework. VAST client supports:

- OAuth2.0 [\(rfc 6749\)](#)
 - Grant types:
 - Authorization Code Grant - *(Most common and web-based)*
 - Client Credentials Grant - *(Machine-machine communication, no user involved)*
 - ~~Implicit Grant~~ - *(not supported, useful for clients running in browser)*
 - Resource Owner Password Grant - *(deprecated, exists for compatibility)*
 - JWT Authorization/Bearer Grant [\(rfc 7523\)](#) - *(Existing trust relationship)*
 - PKCE [\(rfc 7636\)](#)
- OAuth 2.0 for Native Apps [\(rfc 8252\)](#)
- Threat Model and Security Considerations [\(rfc 6819\)](#)
- Towards OAuth2.1
 - OAuth2.0 with existing extensions and additions
 - OAuth2.0 without insecure options

OpenID Connect (OIDC)

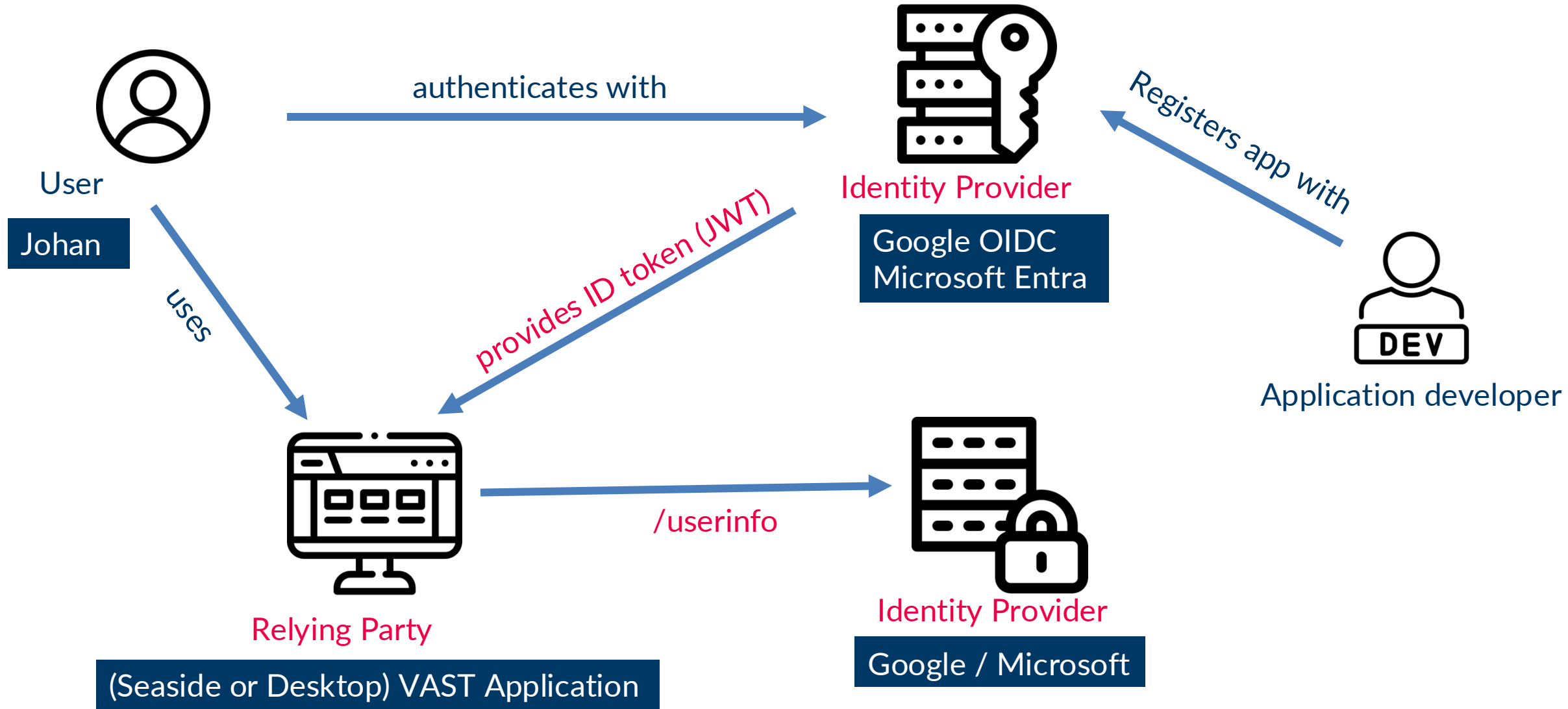
OpenID Connect (OIDC)

- **Authentication** protocol built on top of OAuth2.0
- Verifies the **identity** of a user through an Identity Provider
- Uses **JSON Web Token** with identity claims
- Typically used to implement **Single Sign-On**
- An alternative to the more complex SAML 2.0 protocol



User authentication and identity management delegated to a trusted party.

OpenID Connect (OIDC)



OIDC Discovery

- Each issuer has a **well-known url** with meta-document

<https://login.microsoftonline.com/common/v2.0/.well-known/openid-configuration>

<https://accounts.google.com/.well-known/openid-configuration>

- Meta-document describes OIDC Issuer
- Automated configuration of OAuth2.0 client
 - Endpoint urls
 - Possible values for different configuration parameters
 - Encryption keys for JWT signature verification

OIDC Google / Microsoft

OAuth2 & OIDC demo app

localhost:8888/examples/oauth2oidc?_s=9RqEPB_YQ-gXTIE-&_k=Zwl4tPrisvMmzr-f

Google

OpenID Connect (OIDC)

To learn about your client id, secret and other parameters, see: <https://developers.google.com/identity/openid-connect/openid-connect>

Authentication

Client ID

46095877763-c8mdiprcuckbc0252r76te8p889o9bd

Client Secret

.....

Authorization

Scope

<https://mail.google.com/>

Add scope

Endpoints

Authorize URL

<https://accounts.google.com/o/oauth2/v2/auth>

Access Token URL

<https://oauth2.googleapis.com/token>

Access Token

Expires on 2025-06-17 11:41:24

Token not refreshable

JWT header

```
{  "alg": "RS256",  "kid": "0d8a67399e7882acae7d7f1e2280255",  "typ": "JWT"}
```

JWT claims

```
{  "iss": "https://accounts.google.com",  "azp": "46095877763-c8mdiprcuckbc0252r76te8p889o9bdk.apps.googleusercontent.com",  "aud": "46095877763-c8mdiprcuckbc0252r76te8p889o9bdk.apps.googleusercontent.com"}
```

OAuth2.0 Desktop App Demo

Issuer: Google OIDC

Authorize URL: <https://accounts.google.com/o/oauth2/v2/auth>

Access token URL: <https://oauth2.googleapis.com/token>

Client ID:

Client Secret:

Scope:

Auth in browser Auth in WebView2 Auth with client credentials

Access Token:

Refresh Token:

Refresh:

Inspect Credentials

Help

OIDC Google / Microsoft

```
14.1.0arm64-b577-OAuth2OIDC-ESUG: Workspace
File Edit View Options Encoding Language

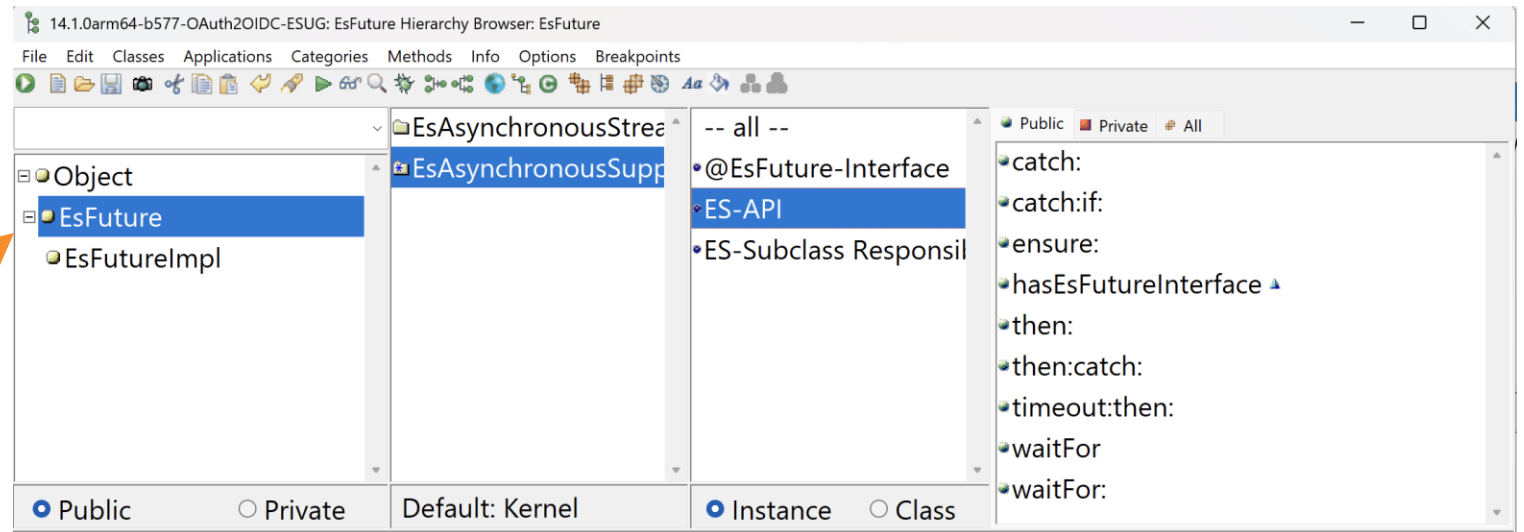
1 "Create a grant object and use it to create the url to which the user needs to be redirected to authenticate."
2 grant := SstOIDCAuthorizationCodeFlow
3     newWithClientId: clientId
4     providerMetadata: SstOIDCProviderMetadata fromIssuerUrl: 'https://accounts.google.com' sstAsUrl.
5 grant
6     clientSecret: 'abcdefghk1125-02';
7     scope: 'offline-access'.
8 authorizationUrl := grant
9     authorizationUrlWithRedirectTo: 'http://localhost:8888/callback' sstAsUrl
10    state: 'random state string'.
11 "Use the async framework as demonstrated in the implementation of the desktop application:"
12 (grant
13     startAuthorization: [:authorizeUrl :callback |
14         self
15             setupHttpListenerAt: 'http://:8888'
16             callback: [:request | callback value: ('http://localhost:8888/callback', request header url) sstAsUrl].
17             OsProcessStarter startShell: { 'start'. "'OAuth2 demo authorize'". ""%1"" bindWith: authorizeUrl asString }}
18     withCallbackAt: 'http://localhost:8888/callback' sstAsUrl
19     state: nil)
20     then: [:client | httpClientWithOAuth2 := client ]
21     catch: [:error | self halt].
22
23 "Use the client to retrieve the userInfo or retrieve the JWT from the credentials so you can authenticate the user in a SSO scenario."
24 httpClientWithOAuth2 userInfo.
25 httpClientWithOAuth2 credentials idTokenAsSstJwt.
```

OIDC Google / Microsoft

```
14.1.0arm64-b577-OAuth2OIDC-ESUG: Workspace
File Edit View Options Encoding Language

1 "Create a grant object and use it to create the url to which the user needs to be redirected to authenticate."
2 grant := SstOIDCAuthorizationCodeFlow
3     newWithClientId: clientId
4     providerMetadata: SstOIDCProviderMetadata fromIssuerUrl: 'https://accounts.google.com' sstAsUrl.
5 grant
6     clientSecret: 'abcdefghk1125-02';
7     scope: 'offline-access'.
8 authorizationUrl := grant
9     authorizationUrlWithRedirectTo: 'http://localhost:8888/callback' sstAsUrl
10    state: 'random state string'.
11 "Use the async framework as demonstrated in the implementation of the desktop application:"
12 (grant
13     startAuthorization: [:authorizeUrl :callback |
14         self
15             setupHttpListenerAt: 'http://:8888'
16             callback: [:request | callback value: ('http://localhost:8888/callback', request header url) sstAsUrl].
17             OsProcessStarter startShell: { 'start'. "'OAuth2 demo authorize'". "%1" bindWith: authorizeUrl asString }}
18     withCallbackAt: 'http://localhost:8888/callback' sstAsUrl
19     state: nil)
20     then: [:client | httpClientWithOAuth2 := client ]
21     catch: [:error | self halt].
22
23 "Use the client to retrieve the userInfo or retrieve the JWT from the credentials so you can authenticate the user in a SSO scenario."
24 httpClientWithOAuth2 userInfo.
25 httpClientWithOAuth2 credentials idTokenAsSstJwt.
```

Async programming convenience



```
grant := self createAuthorizationCodeGrant...  
(grant  
  startAuthorization: [:authorizeUrl callback |  
    self  
      setupHttpListenerAt http://: , self localHttpServerPort asString  
      callback: [:request | callback value: (self redirectUrlString , request header url) sstAsUrl].  
      OsProcessStart startShell: { 'start'. "'OAuth2 demo authorize'". "%1" bindWith: authorizeUrl asString } }  
  withCallbackAt: self redirectUrlString sstAsUrl  
  state: nil)  
  then: [:client | self credentials: client oauth2Credentials]  
  catch: [:error | self halt]
```


OIDC Google / Microsoft

```
14.1.0arm64-b577-OAuth2OIDC-ESUG: Workspace
File Edit View Options Encoding Language

1 "Create a grant object and use it to create the url to which the user needs to be redirected to authenticate."
2 grant := SstOIDCAuthorizationCodeFlow
3     newWithClientId: clientId
4     providerMetadata: SstOIDCProviderMetadata fromIssuerUrl: 'https://accounts.google.com' sstAsUrl.
5 grant
6     clientSecret: 'abcdefghk1125-02';
7     scope: 'offline-access'.
8 authorizationUrl := grant
9     authorizationUrlWithRedirectTo: 'http://localhost:8888/callback' sstAsUrl
10    state: 'random state string'.
11 "Use the async framework as demonstrated in the implementation of the desktop application:"
12 (grant
13     startAuthorization: [:authorizeUrl :callback |
14         self
15             setupHttpListenerAt: 'http://:8888'
16             callback: [:request | callback value: ('http://localhost:8888/callback', request header url) sstAsUrl].
17             OsProcessStarter startShell: { 'start'. "'OAuth2 demo authorize'". "%1" bindWith: authorizeUrl asString }}
18     withCallbackAt: 'http://localhost:8888/callback' sstAsUrl
19     state: nil)
20     then: [:client | httpClientWithOAuth2 := client ]
21     catch: [:error | self halt].
22
23 "Use the client to retrieve the userInfo or retrieve the JWT from the credentials so you can authenticate the user in a SSO scenario."
24 httpClientWithOAuth2 userInfo.
25 httpClientWithOAuth2 credentials idTokenAsSstJwt.
```

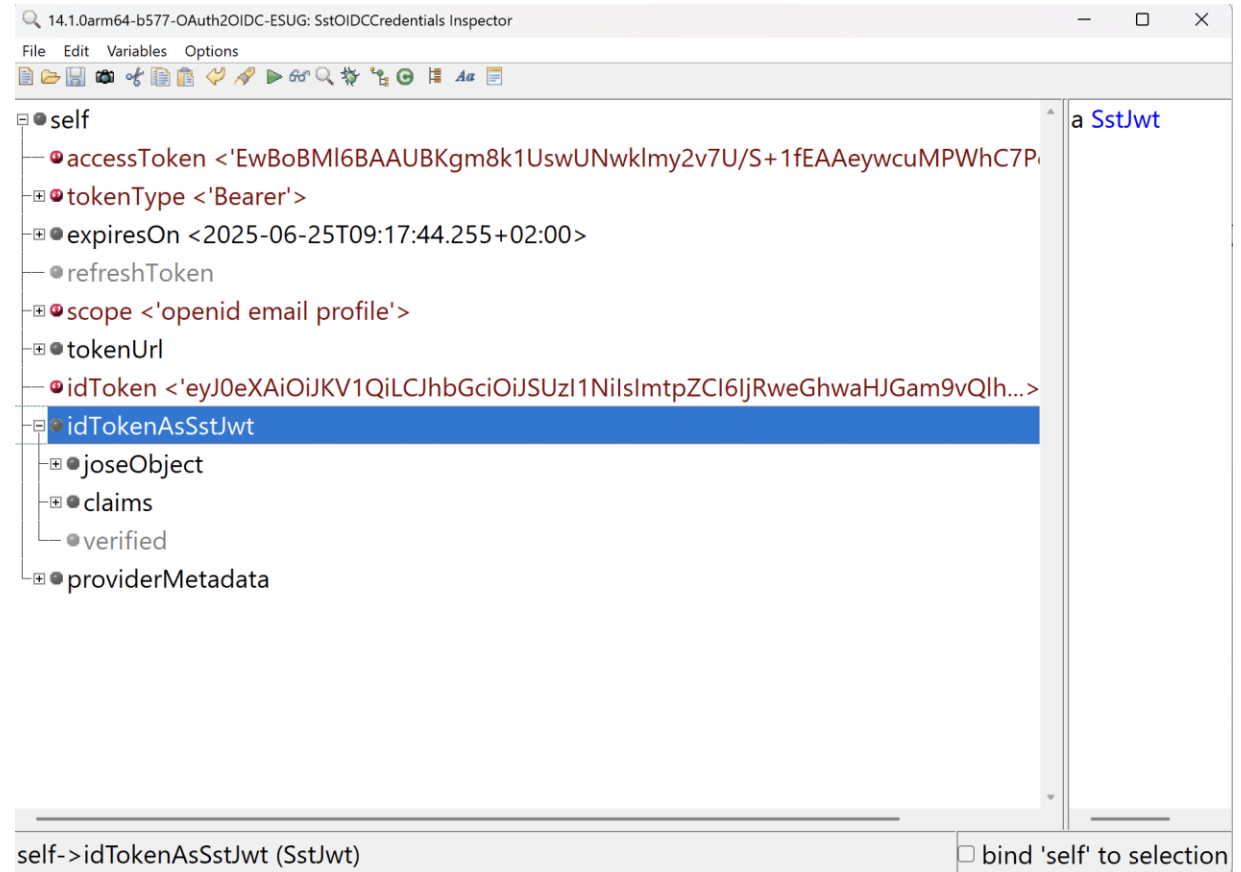
OIDC Google / Microsoft

```
14.1.0arm64-b577-OAuth2OIDC-ESUG: Workspace
File Edit View Options Encoding Language

1 "Create a grant object and use it to create the url to which the user needs to be redirected to authenticate."
2 grant := SstOIDCAuthorizationCodeFlow
3     newWithClientId: clientId
4     providerMetadata: SstOIDCProviderMetadata fromIssuerUrl: 'https://accounts.google.com' sstAsUrl.
5 grant
6     clientSecret: 'abcdefghk1125-02';
7     scope: 'offline-access'.
8 authorizationUrl := grant
9     authorizationUrlWithRedirectTo: 'http://localhost:8888/callback' sstAsUrl
10    state: 'random state string'.
11 "Use the async framework as demonstrated in the implementation of the desktop application:"
12 (grant
13     startAuthorization: [:authorizeUrl :callback |
14         self
15             setupHttpListenerAt: 'http://:8888'
16             callback: [:request | callback value: ('http://localhost:8888/callback', request header url) sstAsUrl].
17             OsProcessStarter startShell: { 'start'. "'OAuth2 demo authorize'". "%1" bindWith: authorizeUrl asString })
18     withCallbackAt: 'http://localhost:8888/callback' sstAsUrl
19     state: nil)
20     then: [:client | httpClientWithOAuth2 := client ]
21     catch: [:error | self halt].
22
23 "Use the client to retrieve the userInfo or retrieve the JWT from the credentials so you can authenticate the user in a SSO scenario."
24 httpClientWithOAuth2 userInfo.
25 httpClientWithOAuth2 credentials idTokenAsSstJwt.
```

Json Web Token

- Standard and web-safe transmission of trusted information (rfc 7519)
- Client library verifies:
 - Signature (RSA 256)
 - Issuer
 - Validity
 - ...
- Separate VAST Library



14.1.0arm64-b577-OAuth2OIDC-ESUG: SstJwt Inspector

File Edit Variables Options

self

- joseObject
- claims
- properties
 - 'aio' <'DmpijAlCvrQUclBMNbWDi...
 - 'aud' <'12b9a8b9-ed41-4feb-b37...
 - 'email' <'johan@inceptive.be'>
 - 'exp' <1750918670>
 - 'iat' <1750831970>
 - 'iss' <'https://login.microsoftonlin...
 - 'name' <'Johan Brichau'>
 - 'nbf' <1750831970>
 - 'nonce' <'FgnTriNvmKNYqYmac6ON6xe5icsiWixM7_er9scrsgs'>
 - 'oid' <'00000000-0000-0000-7e0c-783ae3638e3a'>
 - 'preferred_username' <'johan@in...
 - 'sub' <'AAAAAAAAAAAAAAAAAAAAAAKtdW6SL26wBX0OnPHQdKnM'>
 - 'tid' <'9188040d-6c67-4c5b-b112-36a304b66dad'>
 - 'ver' <'2.0'>

```
a SstJwt
a SstJws
{
  "ver": "2.0",
  "iss": "https://login.microsoftonline.com/9188040d-6c67-4c5b-b112-36a304b66dad",
  "sub": "AAAAAAAAAAAAAAAAAAAAAAKtdW6SL26wBX0OnPHQdKnM",
  "aud": "12b9a8b9-ed41-4feb-b377-acb418c52065",
  "exp": 1750918670,
  "iat": 1750831970,
  "nbf": 1750831970,
  "name": "Johan Brichau",
  "preferred_username": "johan@inceptive.be",
  "oid": "00000000-0000-0000-7e0c-783ae3638e3a",
  "email": "johan@inceptive.be",
  "tid": "9188040d-6c67-4c5b-b112-36a304b66dad",
  "nonce": "FgnTriNvmKNYqYmac6ON6xe5icsiWixM7_er9scrsgs",
  "aio": "DmpijAlCvrQUclBMNbWDiAtK4ybcHJbAdCI8RRnBm!YSHRJz048iyp5wsL!ag..."
}
```

EsOrderedDictionary('2.0' 'https://login.microsoftonline.com/9188040d-6c67-4c5b-...

Multiple variables selected. ☐ bind 'self' to selection

Final Remarks

OAuth2.0 / OIDC Client in VAST

- Additional API parameters

```
1 grant := SstOAuth2AuthorizationCodeGrant
2     newWithClientId: '12345679'
3     authorizationBaseUrl: 'https://oauth2-service.com/authorize' sstAsUrl
4     tokenUrl: 'https://oauth2-service.com/accesstoken' sstAsUrl.
5 grant
6     clientSecret: 'abcdefghk1125-02';
7     addAuthorizationParameter: 'access_type' value: 'offline'
```

- Convenience API for Google and Microsoft

- ▣ SstOAuth2AbstractGrant
- ▣ SstOAuth2AuthorizationCodeGrant
- ▣ SstOIDCAuthorizationCodeFlow
 - ▣ SstGoogleOIDCAuthorizationCodeFlow
 - ▣ SstMicrosoftOIDCAuthorizationCodeFlow

Recap and Final Remarks

- VAST Library passes OIDC conformance tests for **Basic Relying Party Profile**
- Client support for OAuth2.0 APIs and OpenID Connect
 - Compatible with SstHttpClient
 - Convenience methods for Microsoft Entra and Google OIDC
 - Desktop and Seaside example
 - External webbrowser and Webview2
 - Convenience integration with VAST's Async framework
- Included with **VAST 15 (2026)**
- Preview available on request for VAST 14

Seamless OAuth2.0 & OpenID Connect in VAST

Thank you for listening!

Questions?

Johan Brichau

VAST Consultant and Senior Software Engineer

✉ jbrichau@instantiations.com

Contact

General Inquiry
info@instantiations.com

Sales
sales@instantiations.com

VAST Support Portal
vast-support.instantiations.com

North America, Toll Free
855 476 2558

International
+1 503 263 0058