KBC Ireland

Dynamic Registration

Contents

Introduction

The PSD2 Accounts Access APIs of KBC Ireland are developed based on the Open Banking UK standards (v3.1) and the oAuth/OIDC framework. These APIs will need a unique client identifier and a client secret and these will be issued by the KBC Ireland Dynamic Registration API.

Before you start

You will need information about our configuration. You can find more information on our configuration endpoint *here*. Note – you will need a valid eIDAS certificate (QWAC) to access this endpoint.

You will need a set of valid eIDAS certificates – QWAC and QSEAL.

You will need to host your key set (JWKS) and should have the URL ready.

Register and Manage an Application

KBC Ireland dynamic registration API enables you to register your application details and contact details of authorized representatives with us and receive a unique client identifier and a client secret. You can also manage your applications once it is successfully registered.

We support the following endpoints;

Endpoint	Description	Security
POST /register	Register a new oAuth client.	Use your QWAC to access this endpoint.
GET /register/{ClientId}	Retrieve details of an existing client.	Client credentials
PUT /register/{ClientId}	Modify an existing client.	Client credentials
DELETE /register/{ClientId}	Delete an existing client.	Client credentials

POST /register

This endpoint allows you to register your application and on successful validations you will receive a unique client identifier and a client secret. You will need to use your eIDAS certificate (QWAC) to access this endpoint.

Note – if the organization identifier in your eIDAS certificate has an NCA ID issued by Central Bank of Ireland then your client credentials are enabled instantly. For other countries the client credentials will be issued instantly but we will take one business day to enable them for further use.

Payload

Request

The request payload for the POST method is a JWS in the body of the request. The claims for the JWS are listed in the following table.

Claims for JWS Body

Headers			
Name	**Occurrence**	**Description**	**Notes**
typ	1..1	Set to "JWT"	
alg	1..1	RS256	
kid	1..1	The kid will be kept the same as the "x5t" parameter. (X.509 Certificate SHA-1 Thumbprint) of the QSealC.	
Payload			
Name	**Occurrence**	**Description**	**Notes**
iss	1..1	Use the Org ID as specified in your eIDAS certificate. **Note** - KBCI will not process the request where the value does not match the org id in the certificate. E.g. PSDIE-CBI-11221122	String (18)
iat	1..1	The time at which the request was issued by the TPP expressed as "seconds since the epoch"	Integer

exp	1..1	The time at which the request expires expressed as seconds since the epoch. KBCI will not process the request where the current time is greater than the time specified in the claim.	Integer
aud	1..1	Use the Org ID of KBC Ireland. In case the ID in the request does not match our ID then we will reject the request. **Note** - Set to value “PSDIE-CBI-C26910”.	String (18)
jti	1..1	A unique identifier for the JWT. The value must be a UUIDv4 GUID.	String (36)
redirect_uris	1..*	Redirect URIs for interaction with KBCI authorisation flows. Should match the URIs in the SSA. If the software statement defines a master set of redirect URIs, this must match the redirect URIs in the SSA. Each of the URIs must adhere to the following rules: The URI MUST use the https scheme The URI MUST NOT contain a host with a value of localhost	String[] (Each string upto 256)
token_endpoint_auth_method	1..1	We support only client_secret_post in this release of our APIs. **Note** - Set to value "client_secret_post"	String (32)
grant_types	1..*	A JSON array specifying what the TPP can request to be supplied to the token endpoint as exchange for an access token. **Note** - Set the array to contain the following values; client_credentials authorization_code refresh_token	String[] (32)
response_types	0..*	We only support the type "code" in this release of our APIs. **Note** – Set to value "code".	String[] (32)
scope	0..*	Scope of APIs. In addition to a "openid" this should match the scopes in the eIDAS certificates. If not specified we will assign the scopes as per the scope in the certificate. **Note** – based on scope set values to ["accounts", "payments", "fundsconfirmations"]	String[] (32)
software_statement	1..1	Refer the next section. **Note** – this field should contain the JWS. KBCI accepts a self signed SSA. It must be signed using the QSEAL.	JWS
application_type	1..1	**Note** – allowed values are one of “web” or “mobile”.	String (32)

id_token_signed_response_alg	1..1	Algorithm which the TPP expects to sign the id_token, if an id_token is returned. **Note** – in the current release we support only RS256. Further releases will have a wider support. Set to value "RS256".	String (5)
request_object_signing_alg	1..1	Algorithm which the TPP expects to sign the request object if a request object will be part of the authorization request sent to the ASPSP. **Note** – in the current release we support only RS256. Further releases will have a wider support. Set to value "RS256".	String (5)
software_id	0..1	If specified, the software_id in the request MUST match the software_id specified in the SSA. ASPSPs can choose to allow multiple registrations for a given software statement. The Software ID must be represented as a Base62 UUID **Note** – should match the software ID in the SSA.	String (18)
tls_client_auth_dn	0..1	This value must be set if token_endpoint_auth_method is set to tls_client_auth	String (128)

Claims for SSA

Header			
Name	**Occurrence**	**Description**	**Notes**
typ	1..1	Set to "JWT"	
alg	1..1	RS256	
kid	1..1	The kid will be kept the same as the "x5t" parameter. (X.509 Certificate SHA-1 Thumbprint) of QSealC.	
Payload			
Name	**Occurrence**	**Description**	**Notes**
redirect_uris	1..*	Each of the URIs must adhere to the following rules: The URI MUST use the https scheme The URI MUST NOT contain a host with a value of localhost **Note** – this should match the URLs in the request JWS.	
client_name	1..1	User friendly name of the client. This will be displayed to the KBC customer in the consent authorisation flow. **Note** – we will display this name on our mobile app and online application for consent authorization. Ideally, it should be your application's brand name.	String (65)
jwks_uri	1..1	Contains all active QWAC and QSEAL certificates the client	
contacts	1..3	Details of the primary contact in your organisation. KBCI will use these details for all communication in relation to your application.	

contacts.name	1..1	Name of the person.	String (65)
contacts.email	1..1	A valid work email address.	String (35)
contacts.phone	1..1	A valid work phone number in the format +<country code><area code><number>	String (35)
org_id	1..1	Org ID as specified in the QSEAL. **Note** - KBCI will not process the request where the value does not match the org id in the certificate.	
client_description	1..1	A short description of your application.	String (65)
software_id	0..1	If specified, the software_id in the request MUST match the software_id specified in the SSA. ASPSPs can choose to allow multiple registrations for a given software statement. The Software ID must be represented as a Base62 UUID **Note** – should match the software ID in the request JWS.	

Response

The response is a flattened json, which includes the values from the request and a unique client identifier and a client secret.

Samples

Note

Creation of JWS should follow the steps as defined in the Open Banking UK standards (https://openbanking.atlassian.net/wiki/spaces/DZ/pages/83919096/Open+Banking+Security+Profile+-+Implementer+s+Draft+v1.1.2#OpenBankingSecurityProfile-Implementer'sDraftv1.1.2-JSONSecuritySuiteInformationv1.0)

Request JWS Body

eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6InRwcGNsaWVudC5xc2VhbC5mZXQuaWUua2JjLWdyb3VwLmNvbSJ9.eyAgImlzcyI6ICJQU0RJRS1DQkktMTIxMjEyIiwgICJleHAiOiAxOTA4NjIxNTQxLCAgImlhdCI6IDE5MDg2MjE1NDEsICAiYXVkIjogIlBTRElFLUNCSS1DMjY5MTAiLCAgImp0aSI6ICIxMjMyMTMyMTMiLCAgInJlZGlyZWN0X3VyaXMiOiBbICAgICJodHRwczovL215YXBwL2NiPyIsICAgICJodHRwczovL215T3RoZXJhcHAvY2I_IiwgICAgImh0dHBzOi8vbXlPdGhlck90aGVyL2NiPyIgIF0sICAidG9rZW5fZW5kcG9pbnRfYXV0aF9tZXRob2QiOiAiY2xpZW50X3NlY3JldF9wb3N0IiwgICJncmFudF90eXBlcyI6IFsgICAgImNsaWVudF9jcmVkZW50aWFscyIsICAgICJhdXRob3JpemF0aW9uX2NvZGUiLCAgICAicmVmcmVzaF90b2tlbiIgIF0sICAic2NvcGUiOiBbICAgICJhY2NvdW50cyIsICAgICJwYXltZW50cyIsICAgICJmdW5kc2NvbmZpcm1hdGlvbnMiICBdLCAgImFwcGxpY2F0aW9uX3R5cGUiOiAid2ViIiwgICJpZF90b2tlbl9zaWduZWRfcmVzcG9uc2VfYWxnIjogIlJTMjU2IiwgICJyZXF1ZXN0X29iamVjdF9zaWduaW5nX2FsZyI6ICJSUzI1NiIsICAidGxzX2NsaWVudF9hdXRoX2RuIjogInRsc19jbGllbnRfYXV0aF9kbiIsICAic29mdHdhcmVfaWQiOiAiMTIzMTIzIiwgICJzb2Z0d2FyZV9zdGF0ZW1lbnQiOiAiZXlK

MGVYQWlPaUpLVjFRaUxDSmhiR2NpT2lKU1V6STFOaUlzSW10cFpDSTZJblJ3Y0dOc2FXVnVkQzV4YzJWaGJ
DNW1aWFF1YVdVdWEySmpMV2R5YjNWd0xtTnZiU0o5LmV5QWdJbkpsWkdseVpXTjBYM1Z5YVhNaU9p
QmJJQ0FnSUNKb2RIUndjem92TDIxNVlYQndMMk5pUHlJc0lDQWdJQ0pvZEhSd2N6b3ZMMjE1VDNSb1pY
SmhjSEF2WTJJX0lpd2dJQ0FnSW1oMGRIQnpPaTh2YlhsUGRHaGxjazkwYUdWeUwyTmlQeUlnSUYwc0lDQ
WljMjltZEhkaGNtVmZhV1FpT2lBaU1USXpNVEl6SWl3Z0lDSmpiR2xsYm5SZmJtRnRaU0k2SUNKQmNIQWd
WVUZVSURFaUxDQWdJbU5zYVdWdWRGOWtaWE5qY21sd2RHbHZiaUk2SUNKVVpYTjBJR05zYVdWdWR
DQjJhV0VnUjFjZ01TSXNJQ0FpYW5kcmMxOTFjbWtpT2lBaWFIUjBjRG92THpFeU55NHdMakF1TVRvNE1EZ
3dMM0J6WkRJdGFXVXZjWE5sWVd3dFpHbHpZMjkyWlhKNUwzWXlMakF2YTJWNWN5SXNJQ0FpYjNKblg
ybGtJam9nSWxCVFJFbEZMVU5DU1MxRE1qWXdNVGtpTENBZ0ltTnZiblJoWTNSeklqb2dXeUFnSUNCN0lD
QWdJQ0FnSW01aGJXVWlPaUFpU205b2JpQkViMlVpTENBZ0lDQWdJQ0psYldGcGJDSTZJQ0pxYjJodUxtUn
ZaVUJsZUdGdGNHeGxMbU52YlNJc0lDQWdJQ0FnSW5Cb2IyNWxJam9nTXpVek1UVTFOVGs1T1RrZ0lDQ
WdmU3dnSUNBZ2V5QWdJQ0FnSUNKdVlXMWxJam9nSWtwaGJtVWdSRzlsSWl3Z0lDQWdJQ0FpWlcxaGF
Xd2lPaUFpYW1GdVpTNWtiMlZBWlhoaGJYQnNaUzVqYjIwaUxDQWdJQ0FnSUNKd2FHOXVaU0k2SURNM
U16RTFOVFU1TURBd0lDQWdJSDBnSUYxOS5SLVdmX0FaVmJSOHFnRlVLcHV4eEswTGRZWEUzNk9ua1A1
T29ZTTNzMVZmdkNaUUo0SFFwek9nekdwNDk4bnN1T0plMVl6SVZydHkwa05CYXp1TjJGUnJSRE14RnFyb
3ZJWFZpVzlhbm50RG01SVdhdm50cWUtTHV1OHBXWXV1NzBPeXE2bnJOUm9xOWZ0cDlCWEZpcmFVWV
FJV3ZuVlNCZHk3LVdkSWg5MVdaRW5uRkY0YUsxNWJDSFViYXdkMnJKNzN2NnR4cEt2MTIzZkNXckNlWmx
SeDJoUmVmVERxZnVodXNTZTk2NWFCMU94bjFDLXFXaGc2X2lOTkctbUV1a0o3T3dsdndkcWFQWEZJYkU
1NUxSSkFteWpqYlBTUVBGS0JJbVpfV29jS0p0cE5qZjlrU0s2eldLZW4yay1UYlR1Ty03YXRjS3h0OFlsSjNIalNh
clEifQ.N2sFnqa712gi1jAUvSq8tbvznvU3yB2nAwsuGXkxQeGh8gMvHOkPQOpdRL9birIImeramJ8vQEELdfe
a3gLhxpRkDGOx4oxo6O52GE84wiVdCHwCMoq_udfkAHrgPXhqGSciJY88jITGQ1Rsh7YgLYGF3VKQSMBBr
eL8OF2C00UbXwRpYd0J7ZemkH9M7qYFkm-
4oxcnD8U1yBZBRvpSdRcYdhm_Hg2TKJpZBZbc1ms68KOfzpRuXdEVTyZXi8PCQ3VhJEcfkRaouCq-
RSatY89malvxOauCvu9lbRpE8qo7GszbExOKYAJ9jmvp5wdhpUo4pJHrDScL2nQexoEXuw

Decoded JWS – Payload

{
 "typ": "JWT",
 "alg": "RS256",
 "kid": "your qseal KID"
}.

{
 "iss": "PSDIE-CBI-121212",
 "exp": 1908621541,
 "iat": 1908621541,
 "aud": "PSDIE-CBI-C26910",
 "jti": "123213213",
 "redirect_uris": [
  "https://myapp/cb?",
  "https://myOtherapp/cb?",
  "https://myOtherOther/cb?"
 ],

```
 "token_endpoint_auth_method": "client_secret_post",
 "grant_types": [
  "client_credentials",
  "authorization_code",
  "refresh_token"
 ],
 "scope": [
  "accounts",
  "payments",
  "fundsconfirmations"
 ],
 "application_type": "web",
 "id_token_signed_response_alg": "RS256",
 "request_object_signing_alg": "RS256",
 "tls_client_auth_dn": "tls_client_auth_dn",
 "software_id": "123123",
 "software_statement":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6InRwcGNsaWVudC5xc2VhbC5mZXQuaWUua2JjLWdyb3
VwLmNvbSJ9.eyAgInJlZGlyZWN0X3VyaXMiOiBbICAgICJodHRwczovL215YXBwL2NiPyIsICAgICJodHRwczov
L215T3RoZXJhcHAvY2I_IiwgICAgImh0dHBzOi8vbXlPdGhlck90aGVyL2NiPyIgIF0sICAic29mdHdhcmVfaWQi
OiAiMTIzMTIzIiwgICJjbGllbnRfbmFtZSI6ICJBcHAgVUFUIDEiLCAgImNsaWVudF9kZXNjcmlwdGlvbiI6ICJUZX
N0IGNsaWVudCB2aWEgR1cgMSIsICAiandrc191cmkiOiAiaHR0cDovLzEyNy4wLjAuMTo4MDgwL3BzZDIta
WUvcXNlYWwtZGlzY292ZXJ5L3YyLjAva2V5cyIsICAib3JnX2lkIjogIlBTRElFLUNCSS1DMjYwMTkiLCAgImNvb
nRhY3RzIjogWyAgICB7ICAgICAgIm5hbWUiOiAiSm9obiBEb2UiLCAgICAgICJlbWFpbCI6ICJqb2huLmRvZUBl
eGFtcGxlLmNvbSIsICAgICAgInBob25lIjogMzUzMTU1NTk5OTkgICAgfSwgICAgeyAgICAgICJuYW1lIjogIkphb
mUgRG9lIiwgICAgICAiZW1haWwiOiAiamFuZS5kb2VAZXhhbXBsZS5jb20iLCAgICAgICJwaG9uZSI6IDM1Mz
E1NTU5MDAwICAgIH0gIF19.R-
Wf_AZVbR8qgFUKpuxxK0LdYXE36OnkP5OoYM3s1VfvCZQJ4HQpzOgzGp498nsuOJe1YzIVrty0kNBazuN2F
RrRDMxFqrovIXViW9anntDm5IWavntqe-Luu8pWYuu70Oyq6nrNRoq9ftp9BXFiraUYQIWvnVSBdy7-
WdIh91WZEnnFF4aK15bCHUbawd2rJ73v6txpKv123fCWrCeZlRx2hRefTDqfuhusSe965aB1Oxn1C-
qWhg6_iNNG-mEukJ7OwlvwdqaPXFIbE55LRJAmyjjbPSQPFKBImZ_WocKJtpNjf9kSK6zWKen2k-TbTuO-
7atcKxt8YlJ3HjSarQ"
}
```


Decoded SSA


```
{
 "typ": "JWT",
 "alg": "RS256",
 "kid": "your qseal KID "
}.
{
 "redirect_uris": [
  "https://myapp/cb?",
  "https://myOtherapp/cb?",
```

```
  "https://myOtherOther/cb?"
 ],
 "software_id": "123123",
 "client_name": "App UAT 1",
 "client_description": "Test client via GW 1",
 "jwks_uri": "http://127.0.0.1:8080/psd2-ie/qseal-discovery/v2.0/keys",
 "org_id": "PSDIE-CBI-C26019",
 "contacts": [
  {
   "name": "John Doe",
   "email": "john.doe@example.com",
   "phone": 35315559999
  },
  {
   "name": "Jane Doe",
   "email": "jane.doe@example.com",
   "phone": 35315559000
  }
 ]
}
```


Response Body


```
{
   "client_id": "your client id",
   "client_secret": "your client secret",
   "client_secret_expires_at": "0",
   "redirect_uris": [
      "https://myapp/cb?",
      "https://myOtherapp/cb?",
      "https://myOtherOther/cb?"
   ],
   "token_endpoint_auth_method": "client_secret_post",
   "grant_types": [
      "client_credentials",
      "authorization_code",
      "refresh_token"
   ],
   "response_types": [
      "code"
   ],
   "scope": [
      "payments",
      "accounts",
      "fundsconfirmations"
```

```
  ],
  "application_type": "web",
  "id_token_signed_response_alg": "RS256",
  "request_object_signing_alg": "RS256",
  "client_name": "App UAT 1",
  "client_description": "Test client via GW 1",
  "jwks_uri": "http://127.0.0.1:8080/psd2-ie/qseal-discovery/v2.0/keys",
  "org_id": "PSDIE-CBI-C26019",
  "contacts": [
    {
      "name": "John Doe",
      "email": "john.doe@example.com",
      "phone": "35315559999"
    },
    {
      "name": "Jane Doe",
      "email": "jane.doe@example.com",
      "phone": "35315559000"
    }
  ],
  "tls_client_auth_dn": "tls_client_auth_dn",
  "software_id": "123123"
}
```


PUT /register/{ClientId}

This endpoint allows you to modify the details of your application and on successful validations you will receive the response as defined for the POST method. You cannot modify the client identifier and client secret.

You will need a client credential type of access token to use this endpoint. You can request a token for one of the registered scopes.

Note – ensure that you include all claims in the JWS even if you don't what to change a value. In such cases keep the original value in the claim. In case a claim is missing the request will not be processed.

Payload

The payload for the PUT method is exactly the same as for POST.

Request

Refer the section for POST method.

Response

Refer the section for POST method.

Samples

Refer the section for POST method.

GET /register/{ClientId}

This endpoint allows you to retrieve the details of your application.

You will need a client credential type of access token to use this endpoint. You can request a token for one of the registered scopes.

Payload

Request

Use the client identifier in the path of the request.

Response

Refer the section for POST method.

Samples

Refer the section for POST method for response.

DELETE /register/{ClientId}

This endpoint allows you to delete the details of your application.

You will need a client credential type of access token to use this endpoint. You can request a token for one of the registered scopes.

Note –

1) Deleting a client will invalidate all tokens.
2) All consent records created for that client will not be accessible any more.
3) Deletion cannot be reversed.

Payload

Request

Use the client identifier in the path of the request.

Response

HTTP 204

Samples

Refer the section for POST method for response.

Error Codes

HTTP Status	Scenario
400 Bad Request	Invalid claim values or missing claims. Details will be provided in the error message.
403 Forbidden	Invalid access token or missing header.
429 Too Many Requests	The operation was refused as too many requests have been made within a certain timeframe.
500 Internal Server Error	Unexpected condition which prevented the request to be fulfilled.