Account and Transaction API Specification - v3.1

Version Control

Version	Date	Author	Comments
3.0	07 Sep 2018	OB R/W API Team	This is the baseline version. No change from RC3. Swagger URLs have been updated to point to the latest stable version.
3.1-draft1	11 Sep 2018	OB R/W API Team	This is the initial draft version for 3.1. Errata • Grammatical Fixes
3.1-draft2	09 Oct 2018	OB R/W API Team	No Change
3.1-draft3	16 Oct 2018	OB R/W API Team	Draft 3 Changes: • Namespaced Enumerations are moved to a separate page • Removed obsolete Static Enumeration - OBExternalFinancialInstitutionIdentification2Code, OBExternalAccountIdentification2Code, OBExternalAccountIdentification3Code from the list • Fixed the wrongly listed OBExternalLimitType2Code, it must be OBExternalLimitType1Code - no change in values • Swagger Specification links updated

3.1-draft4	31 Oct 2018	OB R/W API Team	Draft 4 Changes: • Added CreditorAgent and DebtorAgent to ReadTransactionsDetail permission. • Updated class names in Permissions table • Added a section "ISO enumerations" for fields which are using ISO defined enumeration • Swagger Specification links updated
3.1-RC1	19 Nov 2018	OB R/W API Team	RC1 Changes: • Swagger Specification links updated
3.1	30 Nov 2018	OB R/W API Team	Version 3.1 final release. No changes from Version 3.1 RC1. 12 Dec 2018 Swagger updated to latest release candidate

Overview

This specification describes the Account Information and Transaction API flows and payloads.

The API endpoints described here allow an Account Information Service Provider ('AISP') to:

- Register an intent to retrieve account information by creating an "account access consent". This registers the data "permissions", expiration and historical period allowed for transactions / statements - that the customer (PSU) has consented to provide to the AISP; and
- Subsequently, retrieve account and transaction data.

This specification should be read in conjunction with Read/Write Data API Specification which provides a description of the elements that are common across all the Read/Write Data APIs.

Document Structure

This document consists of the following parts:

Overview: Provides an overview of the API and the key decisions and principles that contributed to the specification.

Basics: Identifies the resources, operations that are permitted on those resources, and various special cases.

Endpoints: Provides the list of endpoints for the API specification. The individual end-points are documented in separate pages along with the data model that they employ and usage examples.

Security & Access Control: Specifies the means for AISPs and PSUs to authenticate themselves and provide consent.

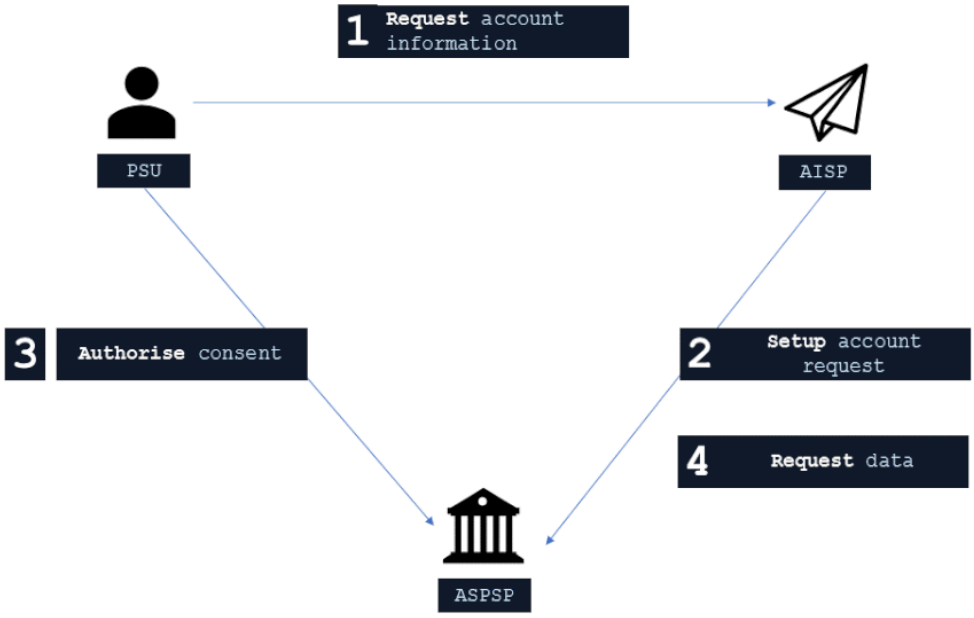
Data & Payloads: Documents data structures and data architecture that applies to all the end-points. End-point specific data structures are documented in separate pages along with the end-points that employ the data structure.

Swagger Specifications: Provides links to the swagger specifications for the APIs.

Basics

Overview

The figure below provides a general outline of an account information request and flow using the Account Info APIs.



Steps

Step 1: Request Account Information

- This flow begins with a PSU consenting to allow an AISP to access account information data.

Step 2: Setup Account Access Consent

- The AISP connects to the ASPSP that services the PSU's account(s) and creates an **account-access-consent** resource. This informs the ASPSP that one of its PSUs is granting access to account and transaction information to an AISP. The ASPSP responds with an identifier for the resource (the ConsentId - which is the intent identifier). This step is carried out by making a **POST** request to **/account-access-consents** endpoint.
- The account-access-consent resource will include these fields below - which describe the data that the PSU has consented with the AISP:
  - Permissions - a list of data clusters that have been consented for access.
  - Expiration Date - an optional expiration for when the AISP will no longer have access to the PSU's data.
  - Transaction Validity Period - the From/To date range which specifies a historical period for transactions and statements which may be accessed by the AISP.
- An AISP may be a broker for data to other parties, and so it is valid for a PSU to have multiple account-access-consents for the same accounts, **with different consent/authorisation parameters agreed**.

Step 3: Authorise Consent

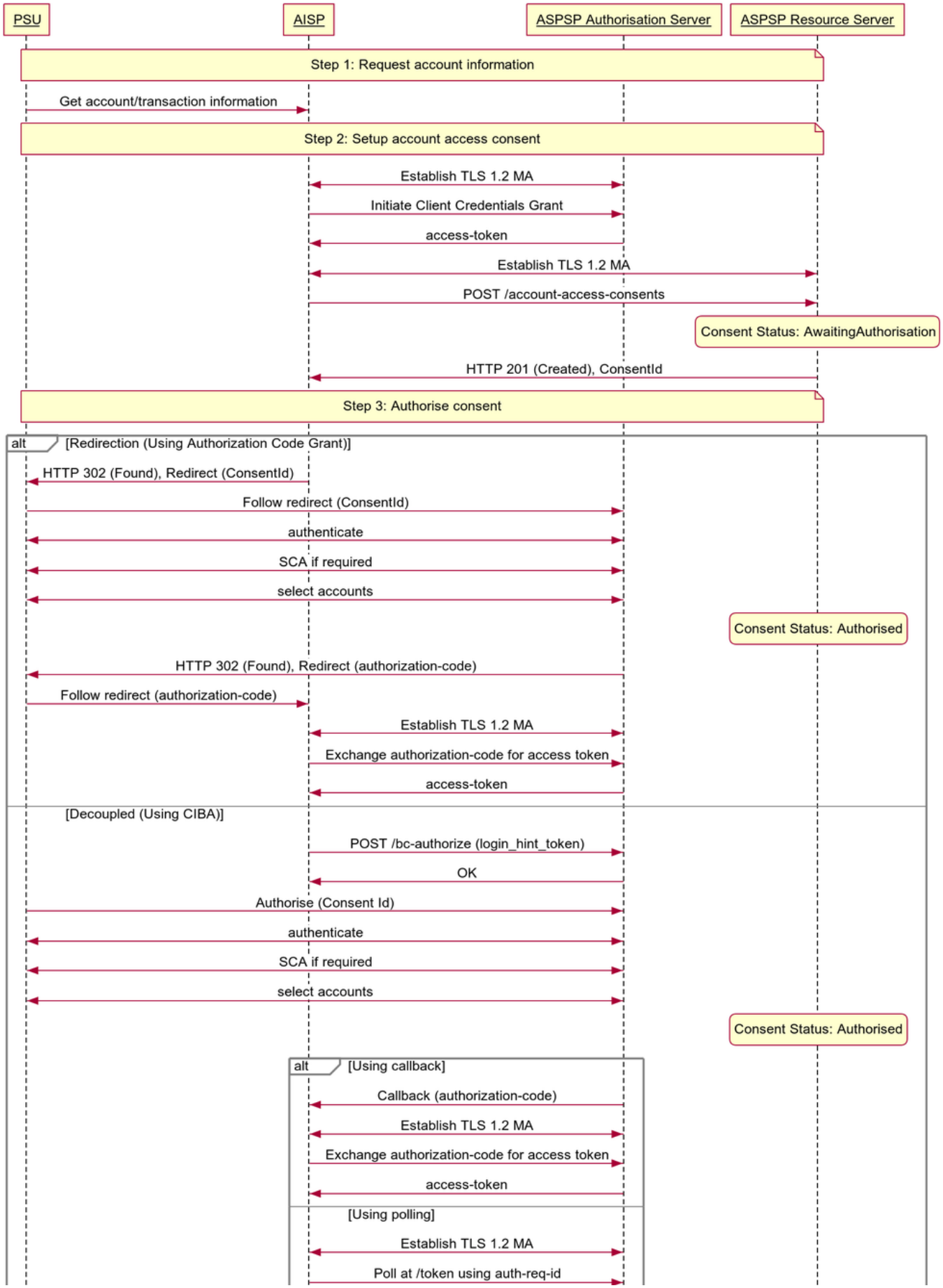
- The AISP requests the PSU to authorise the consent. The ASPSP may carry this out by using a *redirection flow* or a *decoupled flow*.
  - In a redirection flow, the AISP redirects the PSU to the ASPSP.
    - The redirect includes the ConsentId generated in the previous step.
    - This allows the ASPSP to correlate the account-access-consent that was setup.
    - The ASPSP authenticates the PSU.
    - The ASPSP updates the state of the account-access-consent resource internally to indicate that the account access consent has been authorised.
    - Once the consent has been authorised, the PSU is redirected back to the AISP.
  - In a decoupled flow, the ASPSP requests the PSU to authorise consent on an *authentication device* that is separate from the *consumption device* on which the PSU is interacting with the AISP.
    - The decoupled flow is initiated by the AISP calling a back-channel authorisation request.
    - The request contains a 'hint' that identifies the PSU, paired with the consent to be authorised.
    - The ASPSP authenticates the PSU and updates the state of the account-access-consent resource internally to indicate that the account access consent has been authorised.
    - Once the consent has been authorised, the ASPSP can make a callback to the AISP to provide an access token.
- The principle we have agreed is that consent is managed between the PSU and the AISP - so the account-access-consent details must not be changed (with the ASPSP) in this step. The PSU will only be able to authorise or reject the account-access-consent details in its entirety.

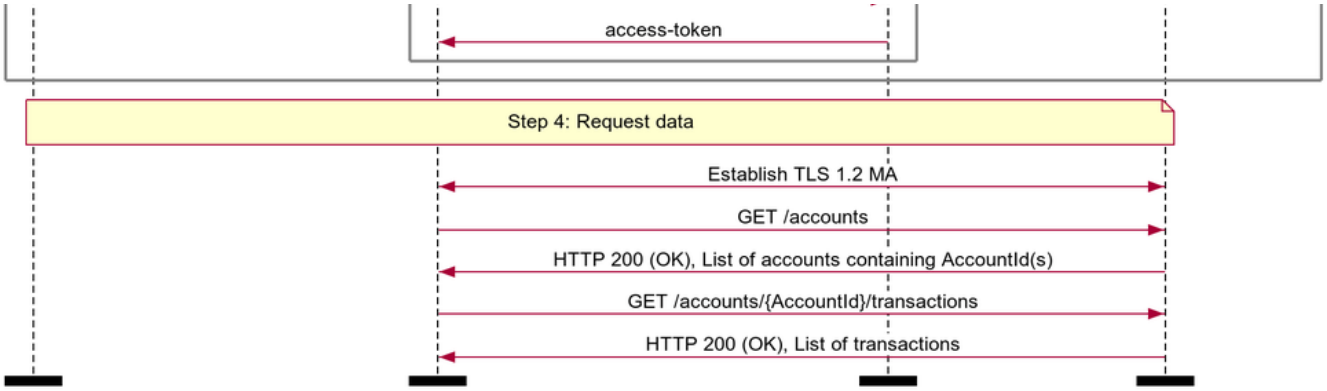
- During authorisation, the PSU selects accounts that are authorised for the AISP request (in the ASPSP's banking interface).

Step 4: Request Data

- This is carried out by making a **GET** request the relevant **resource**.
- The unique AccountId(s) that are valid for the account-access-consent will be returned with a call to GET /accounts. **This will always be the first call once an AISP has a valid access token.**

Sequence Diagram


PSU
AISP
ASPSP Authorisation Server
ASPSP Resource Server
Step 1: Request account information
Get account/transaction information
Step 2: Setup account access consent
Establish TLS 1.2 MA
Initiate Client Credentials Grant
access-token
Establish TLS 1.2 MA
POST /account-access-consents
Consent Status: AwaitingAuthorisation
HTTP 201 (Created), ConsentId
Step 3: Authorise consent
alt
[Redirection (Using Authorization Code Grant)]
HTTP 302 (Found), Redirect (ConsentId)
Follow redirect (ConsentId)
authenticate
SCA if required
select accounts
Consent Status: Authorised
HTTP 302 (Found), Redirect (authorization-code)
Follow redirect (authorization-code)
Establish TLS 1.2 MA
Exchange authorization-code for access token
access-token
[Decoupled (Using CIBA)]
POST /bc-authorize (login_hint_token)
OK
Authorise (Consent Id)
authenticate
SCA if required
select accounts
Consent Status: Authorised
alt
[Using callback]
Callback (authorization-code)
Establish TLS 1.2 MA
Exchange authorization-code for access token
access-token
[Using polling]
Establish TLS 1.2 MA
Poll at /token using auth-req-id



Account Info - High Level Flow

Idempotency

The API endpoints for creating account-access-consent resources **are not** idempotent.

If a time-out error occurs - then we would expect an AISP to create a new account-access-consent resource - rather than try with the same resource.

Release Management

This section overviews the release management and versioning strategy for the Account and Transaction API.

Account Access Consent

The account-access-consent resource is referred to as an account-request resource in v1 and v2 of this specification. For clarity, it has been generalised to 'Consent' in the detail below.

POST

- An AISP **must not** create a Consent on a newer version, and use it on a previous version
  - E.g., A ConsentId for an account-access-consent created in v3, must not be used to access v2 endpoints.

GET

- An AISP **must not** access a Consent on an older version, via the Id for a Consent created in a newer version:
  - E.g., An account-access-consent created in v3 accessed via v2 account-request.
- An ASPSP **must** allow a Consent to be accessed in a newer version.
- An ASPSP **must** ensure Permissions set associated with a Consent are unchanged when accessed in a different version:
  - E.g., An account-request created in v2 will have the same details when accessed via v2 and v3 (as an account-access-consent).
- An ASPSP **must** ensure a Consent's fields are unchanged when accessed in a different version.
- An ASPSP **may** allow expired Consents to be accessed in a newer version.
- An ASPSP **may** choose to populate new fields introduced in a resource from previous version sensible defaults (if mandatory) or not populate at all (if not mandatory):
  - E.g., OBReadResponse1/Data/StatusUpdateDateTime introduced in v2 accessed with v1 AccountRequestId can be populated with Last accessed date time, if not already available in the system of records.

DELETE

- An AISP **must not** delete a Consent on an older version, via an Id for a Consent created in a newer version:
  - E.g., An account-access-consent is created in v3, and request DELETE on v2.
- An ASPSP **must** support deleting a Consent from a previous version via a newer version:
  - E.g., An account-request is created in v2, and request DELETE on v3.

Account Information Resources

GET

- An AISP **may** use a token that is bound to a Consent in a previous version, to access an endpoint of a newer version.
- An AISP **may** use an Id for a Consent created in a previous version to retrieve Account Information resources in a newer version:
    - E.g., AccountRequestId from v2 can be used as ConsentId in v3, to GET /accounts.
- An AISP **must not** use an Id for a Consent from a newer version to access Account Information resources in a previous version:
    - E.g., ConsentId for an account-access-consent created in v3, must not be used to access v2 Account Information endpoints.
- An AISP **must not** use an Id for a Consent from a previous version to access a resource introduced in a newer version (as the Consent will not have Permissions required to access the new resource).
- An ASPSP **must** allow an AISP to use an Id for a Consent from a previous version to access Account Information resource endpoints in a newer version:
    - E.g., AccountRequestId created in v2 must be allowed to access Account Information resource endpoints in v3.
- An ASPSP **must** reject the request to access a resource, for which a Consent's Permissions set does not permit.
- An ASPSP **may** choose to populate new fields introduced in a resource from previous version sensible defaults (if mandatory) or not populate at all:
    - E.g., OBReadResponse1/Data/StatusUpdateDateTime introduced in Version2 accessed with V1 AccountRequestId can be populated with Last accessed date time, if not already available in the system of records.

Endpoints

This section looks at the list of available API endpoints to access Account Information and Transaction data and optionality (definitions of mandatory, conditional or optional are defined in the Principles section).

Endpoint design considerations:

- Having resources that are finer grained (e.g., beneficiaries, direct-debits, standing-orders) means that we can, in the future, manage these resources (with unique identifiers).
- While balances is not a typical resource - we believe having an /accounts/{AccountId}/balances endpoint is simpler to understand than a URI to expand the /accounts resource .
- Some ASPSPs were uncomfortable implementing the bulk APIs (e.g., /accounts, /transactions, /beneficiaries etc.) so the bulk APIs have been specified as optional. However, the bulk endpoint for /accounts is mandatory to discover what accounts have been authorised for the account-access-consent.

We have specified the "mandatory" endpoints for the functioning of the Account Info APIs.

However, endpoints will not be "mandatory" if ASPSPs do not provide these resources via existing online channels e.g., direct debits, standing orders, statements.

Link	Resource	Endpoints	Mandatory?
Account Access Consents v3.1	account-access-consents	POST /account-access-consents	Mandatory
		GET /account-access-consents/{ConsentId}	Mandatory
		DELETE /account-access-consents/{ConsentId}	Mandatory
Accounts v3.1	accounts	GET /accounts	Mandatory
		GET /accounts/{AccountId}	Mandatory
Balances v3.1	balances	GET /accounts/{AccountId}/balances	Mandatory
		GET /balances	Optional
Transactions v3.1	transactions	GET /accounts/{AccountId}/transactions	Mandatory
		GET /transactions	Optional
Beneficiaries v3.1	beneficiaries	GET /accounts/{AccountId}/beneficiaries	Conditional
		GET /beneficiaries	Optional
Direct Debits v3.1	direct-debits	GET /accounts/{AccountId}/direct-debits	Conditional
		GET /direct-debits	Optional
Standing Orders v3.1	standing-orders	GET /accounts/{AccountId}/standing-orders	Conditional
		GET /standing-orders	Optional

Products v3.1	products	GET /accounts/{AccountId}/product	Conditional
		GET /products	Optional
Offers v3.1	offers	GET /accounts/{AccountId}/offers	Conditional
		GET /offers	Optional
Party v3.1	party	GET /accounts/{AccountId}/party	Conditional
		GET /party	Conditional
Scheduled Payments v3.1	scheduled-payments	GET /accounts/{AccountId}/scheduled-payments	Conditional
		GET /scheduled-payments	Optional
Statements v3.1	statements	GET /accounts/{AccountId}/statements	Conditional
		GET /accounts/{AccountId}/statements/{StatementId}	Conditional
		GET /accounts/{AccountId}/statements/{StatementId}/file	Optional
		GET /accounts/{AccountId}/statements/{StatementId}/transactions	Conditional
		GET /statements	Optional

Security & Access Control

Scopes

The access tokens required for accessing the Account Info APIs must have at least the following scope:

Scopes
`accounts`

Grants Types

AISPs **must** use a client credentials grant to obtain a token to access the account-access-consents resource. In the specification, this grant type is referred to as "Client Credentials".

AISPs **must** use an authorization code grant using a redirect or decoupled flow to obtain a token to access all other resources. In the specification, this grant type is referred to as "Authorization Code".

Consent Authorisation

The AISP **must** create an **account-access-consent** resource through a **POST** operation. This resource indicates the *consent* that the AISP claims it has been given by the PSU to retrieve account and transaction information. At this stage, the consent is not yet authorised as the ASPSP has not yet verified this claim with the PSU.

The ASPSP responds with a ConsentId. This is the intent-id that is used when initiating the authorization code grant (as described in the Trust Framework).

As part of the consent authorization flow:

- The ASPSP authenticates the PSU.
- The ASPSP plays back the consent (registered by the AISP) back to the PSU - to get consent authorisation. The PSU may accept or reject the consent in its entirety (but not selectively).
- The ASPSP presents the PSU with a list of accounts to which the consent will apply.

Once these steps are complete, the consent is considered to have been authorised by the PSU.

Consent Elements

The Account Access Consent resource consists of the following fields, which together form the elements of the consent provided by the PSU to the AISP:

- **Permissions:** The set of data clusters that the PSU has consented to allow the AISP to access.
- **ExpirationDateTime:** The date-time up to which the consent is valid.
- **TransactionFromDateTime:** The earliest point of the transaction / statement historical period that the PSU has consented to provide access to the AISP.
- **TransactionToDateTime:** The last point of the transaction / statement historical period that the PSU has consented to provide access to the AISP.

Permissions

Permissions codes will be used to limit the data that is returned in response to a resource request.

When a permission is granted for a "Detail" permission code (e.g., ReadAccountsDetail) it implies that access is also granted to the corresponding "Basic" permission code (e.g., ReadAccountsBasic).

While it is duplication for a TPP to request a "Basic" permission code and the corresponding "Detail" permission code, it is not a malformed request, and the ASPSP must not reject solely on the basis of duplication.

The permissions array **must** contain at least **ReadAccountsBasic** or **ReadAccountsDetail**.

The following combinations of permissions are not allowed, and the ASPSP **must** reject these account-access-consents with a 400 response code:

- Account Access Consents with an empty Permissions array.
- Account Access Consents with a permission code that is not supported by the ASPSP (ASPSPs are expected to publish which API endpoints are supported).
- Account Access Consents with a Permissions array that contains **ReadTransactionsBasic** but does not contain at least one of **ReadTransactionsCredits** and **ReadTransactionsDebits**.
- Account Access Consents with a Permissions array that contains **ReadTransactionsDetail** but does not contain at least one of **ReadTransactionsCredits** and **ReadTransactionsDebits**.
- Account Access Consents with a Permissions array that contains **ReadTransactionsCredits** but does not contain at least one of **ReadTransactionsBasic** and **ReadTransactionsDetail**.
- Account Access Consents with a Permissions array that contains **ReadTransactionsDebits** but does not contain at least one of **ReadTransactionsBasic** and **ReadTransactionsDetail**.

Permissions	Endpoints	Business Logic	Data Cluster Description
ReadAccounts**Basic**	/accounts /accounts/{AccountId}		Ability to read basic account information
ReadAccounts**Detail**	/accounts /accounts/{AccountId}	Access to additional elements in the payload	Ability to read account identification details
ReadBalances	/balances /accounts/{AccountId}/balances		Ability to read **all** balance information
ReadBeneficiaries**Basic**	/beneficiaries /accounts/{AccountId}/beneficiaries		Ability to read basic beneficiary details
ReadBeneficiaries**Detail**	/beneficiaries /accounts/{AccountId}/beneficiaries	Access to additional elements in the payload	Ability to read account identification details for the beneficiary
ReadDirectDebits	/direct-debits /accounts/{AccountId}/direct-debits		Ability to read **all** direct debit information
ReadStandingOrders**Basic**	/standing-orders /accounts/{AccountId}/standing-orders		Ability to read basic standing order information
ReadStandingOrders**Detail**	/standing-orders /accounts/{AccountId}/standing-orders	Access to additional elements in the payload	Ability to read account identification details for beneficiary of the standing order

ReadTransactions**Basic**	/transactions /accounts/{AccountId}/transactions /accounts/{AccountId}/statements/{StatementId}/transactions	Permissions must also include at least one of: • ReadTransactions**Credits** • ReadTransactions**Debits**	Ability to read basic transaction information
ReadTransactions**Detail**	/transactions /accounts/{AccountId}/transactions /accounts/{AccountId}/statements/{StatementId}/transactions	Access to additional elements in the payload Permissions must also include at least one of • ReadTransactions**Credits** • ReadTransactions**Debits**	Ability to read transaction data elements which may hold silent party details
ReadTransactions**Credits**	/transactions /accounts/{AccountId}/transactions /accounts/{AccountId}/statements/{StatementId}/transactions	Access to credit transactions. Permissions must also include one of: • ReadTransactions**Basic** • ReadTransactions**Detail**	Ability to read **only** credit transactions
ReadTransactions**Debits**	/transactions /accounts/{AccountId}/transactions /accounts/{AccountId}/statements/{StatementId}/transactions	Access to debit transactions. Permissions must also include one of: • ReadTransactions**Basic** • ReadTransactions**Detail**	Ability to read **only** debit transactions
ReadStatements**Basic**	/statements /accounts/{AccountId}/statements		Ability to read basic statement details
ReadStatements**Detail**	/statements /accounts/{AccountId}/statements /accounts/{AccountId}/statements/{StatementId}/file	Access to additional elements in the payload Access to download the statement file (if the ASPSP makes this available).	Ability to read statement data elements which may leak other information about the account
ReadProducts	/products /accounts/{AccountId}/product		Ability to read **all** product information relating to the account
ReadOffers	/offers /accounts/{AccountId}/offers		Ability to read **all** offer information
ReadParty	/accounts/{AccountId}/party		Ability to read party information on the account owner.
ReadParty**PSU**	/party		Ability to read party information on the PSU logged in.
ReadScheduledPayments**Basic**	/scheduled-payments /accounts/{AccountId}/scheduled-payments		Ability to read basic statement details
ReadScheduledPayments**Detail**	/scheduled-payments /accounts/{AccountId}/scheduled-payments	Access to additional elements in the payload	

ReadPAN	All API endpoints where PAN is available as a structured field	Request to access to PAN in the clear	Request to access **PAN** in the clear across the available endpoints. If this permission code is not in the account-access-consent, the AISP will receive a masked PAN. While an AISP may request to access PAN in the clear, an ASPSP may still respond with a masked PAN if: • The ASPSP does not display PAN in the clear in existing online channels • The ASPSP takes a legal view to respond with only the masked PAN

DETAIL PERMISSIONS

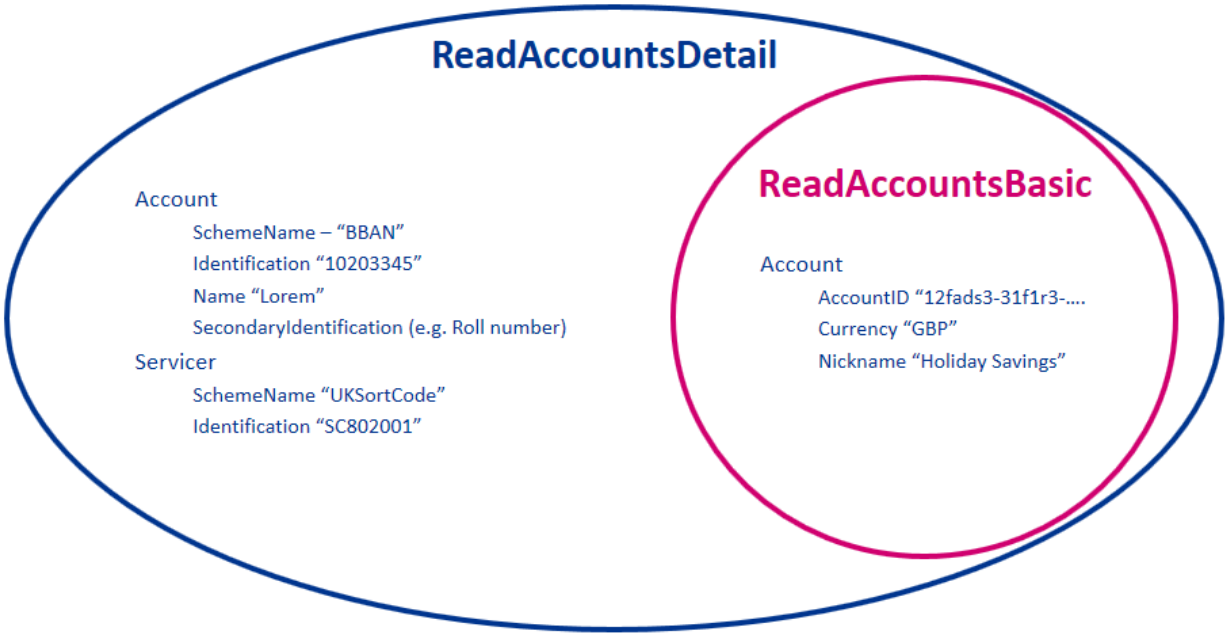
The additional elements that are granted for "Detail" permissions are listed in this section.

All other fields (other than these fields listed) are available with the "Basic" Permission access.

Permission - Detail Codes	Data Element Name	Occurrence	XPath
ReadAccountsDetail	Account	0..1	OBReadAccount3/Data/Account/Account
ReadAccountsDetail	Servicer	0..1	OBReadAccount3/Data/Account/Servicer
ReadBeneficiariesDetail	CreditorAgent	0..1	OBReadBeneficiary3/Data/Beneficiary/CreditorAgent
ReadBeneficiariesDetail	CreditorAccount	0..1	OBReadBeneficiary3/Data/Beneficiary/CreditorAccount
ReadStandingOrdersDetail	CreditorAgent	0..1	OBReadStandingOrder4/Data/StandingOrder/CreditorAgent
ReadStandingOrdersDetail	CreditorAccount	0..1	OBReadStandingOrder4/Data/StandingOrder/CreditorAccount
ReadTransactionsDetail	TransactionInformation	0..1	OBReadTransaction4/Data/Transaction/TransactionInformation
ReadTransactionsDetail	Balance	0..1	OBReadTransaction4/Data/Transaction/Balance
ReadTransactionsDetail	MerchantDetails	0..1	OBReadTransaction4/Data/Transaction/MerchantDetails
ReadTransactionsDetail	CreditorAgent	0..1	OBReadTransaction4/Data/Transaction/CreditorAgent
ReadTransactionsDetail	CreditorAccount	0..1	OBReadTransaction4/Data/Transaction/CreditorAccount
ReadTransactionsDetail	DebtorAgent	0..1	OBReadTransaction4/Data/Transaction/DebtorAgent
ReadTransactionsDetail	DebtorAccount	0..1	OBReadTransaction4/Data/Transaction/DebtorAccount
ReadStatementsDetail	StatementAmount	0..*	OBReadStatement1/Data/Statement/StatementAmount
ReadScheduledPaymentsDetail	CreditorAgent	0..1	OBReadScheduledPayment2/Data/ScheduledPayment/CreditorAgent
ReadScheduledPaymentsDetail	CreditorAccount	0..1	OBReadScheduledPayment2/Data/ScheduledPayment/CreditorAccount

In addition the ReadStatementsDetail is required to access the statement file download via: /accounts/{AccountId}/statements/{StatementId}/file

Example behaviour of the Permissions for the ReadAccountsBasic and ReadAccountsDetail codes is as follows:



REVERSING ENTRIES

It is expected that transactions will be returned in the payload irrespective of whether they are reversing entries, as long as the PSU has provided consent for that type of transaction.

If the PSU has provided permission for ReadTransactionsCredits, the ASPSP **must** include all credits, including debit reversals.

If the PSU has provided permission for ReadTransactionsDebits, the ASPSP **must** include all debits, including credit reversals.

Expiration Date Time

The ExpirationDateTime is an optional field which specifies the expiration for AISP access to the PSU's data.

The field is optional as the consent for AISP access to a PSU's data may be indefinite. The ExpirationDateTime is different from the RTS requirement for a PSU to re-authenticate after 90 days. The same account-access-consent resource will be re-authenticated with the same ExpirationDateTime as the original request.

The ExpirationDateTime applies to all Permissions (data clusters) being consented.

Transaction To/From Date Time

The TransactionToDateTime and the TransactionFromDateTime specify the period for consented transaction and/or statement history. Both the fields are optional and one may be specified without the other.

The AISP **must** be restricted to accessing transactions within this period when accessing the transactions resource.

The AISP **must** be restricted to accessing statements which are **completely** within this period when accessing the statements resource.

Account Access Consent Status

The Account Access Consent resource may have one of the following status codes after authorisation has taken place:

	Status	Description
1	Authorised	The account access consent has been successfully authorised.
2	Rejected	The account access consent has been rejected.
3	Revoked	The account access consent has been revoked via the ASPSP interface.

Consent Re-authentication

Account Access Consents are long-lived consents.

A PSU can re-authenticate an Account Access Consent if:

- The account-access-consent has a status of `Authorised` and
- The `ExpirationDateTime` of the account-access-consent, if specified, has not elapsed.

The accounts bound to the account-access-consent are selected in the ASPSP domain.

An ASPSP **may** allow the PSU to change the selected accounts during consent re-authentication.

Consent Revocation

A PSU may revoke consent for accessing account information at any point in time.

A PSU **may** revoke authorisation directly with the ASPSP. The mechanisms for this are in the competitive space and are up to each ASPSP to implement in the ASPSP's banking interface. If the PSU revokes authorisation with the ASPSP, the Status of the **account-access-consent** resou rce must be set to *Revoked.*

The PSU may request the AISP to revoke consent that it has authorised. If consent is revoked with the AISP:

- The AISP **must** cease to access the APIs at that point.
- The AISP **must** call the **DELETE** operation on the account-access-consent resource (before confirming consent revocation with the PSU) to indicate to the ASPSP that the PSU has revoked consent.

Changes to Selected Account(s)

The PSU **must** select the accounts to which the consent should be applied at the point of consent authorisation.

Subsequent changes to the set of accounts to which the consent authorisation applies **may** be carried out directly with the ASPSP. The method for doing this lies in the competitive space and is not part of this specification.

Additionally, the set of selected accounts may also change due to external factors. This includes (but is not limited to):

- The account being closed.
- The PSU's mandate to operate the account is revoked.
- The account is barred or frozen.
- The PSU changes the selected accounts during consent re-authentication.

In these scenarios, only the affected account is removed from the list of selected accounts. The ASPSP **must not** revoke authorisation to other accounts.

Risk Scoring Information

Information for risk scoring and assessment will come via:

- FAPI HTTP headers. These are defined in Section 6.3 of the FAPI specification and in the Headers section above.
- Additional fields identified by the industry as business logic security concerns - which will be passed in the Risk section of the payload in the JSON object.

No fields for business logic security concerns have been identified for the Account Info APIs.

Data Model

Using Meta to identify Available Transaction Period

For Accounts & Transaction APIs, the `Meta` section in API responses may contain two additional fields to indicate the date range for which data has been returned.

The transactions or statements for a particular range of dates may be excluded from the response because:

- The ASPSP does not provide historical transactions / statements for that date range.
- The PSU has not consented to transactions / statements for that date range.

The absence of transactions / statements in the payload does not indicate that there were no transactions / statements during that period.

To ensure that the data is interpreted correctly, the ASPSP **may** provide the date of the first available transaction and last available transaction as

part of the response in the Meta section in the FirstAvailableDateTime and LastAvailableDateTime fields.

Example Meta

```
"Meta": {
    "TotalPages": 1,
 "FirstAvailableDateTime": "2017-05-03T00:00:00+00:00",
 "LastAvailableDateTime": "2017-12-03T00:00:00+00:00"
  }
```

Mapping to Schemes & Standards

The Account Info API resources, where possible, have been borrowed from the ISO 20022 camt.052 XML standard. However, has been adapted for APIs based as per our design principles.

Deviations from the camt.052 XML standard are:

- The camt.052 header section and trailer sections have been removed as these are not required for a RESTful API.
- Resources have been identified and payload structures have been designed for these resources rather than a full message (i.e., camt.052) that encompasses all resources in a report format. This has meant we have designed separate endpoints and payloads to cover:
    - accounts
    - balances
    - beneficiaries
    - direct-debits
    - offers
    - party
    - products
    - standing-orders
    - statements
    - transactions
    - scheduled-payments
- New payloads have been designed for beneficiaries, direct-debits, standing-orders, and products resources as these are not in the ISO 20022 standard (or the camt.052 message).
- A DateTime element has been used instead of a complex choice element of Date and DateTime (across all API endpoints). Where time elements do not exist in ASPSP systems, the expectation is the time portion of the DateTime element will be defaulted to 00:00:00+00:00 .
- Variations for the accounts structure include:
    - Standardised inline with the Payment API account structures.
    - Contains elements to identify an account Nickname, SecondaryIdentification.
- Variations for the balances structure include:
    - Adding a Type into the CreditLine section to allow for multiple credit line types affecting the available balance.
    - DateTime element has been specified instead of a complex choice of Date and DateTime.
- Variations for the transactions structure include:
    - Renaming "entry" to "transaction" for consistency as this is the language used in the CMA Order and PSD2.
    - DateTime elements used instead of a complex choice of Date and DateTime.
    - Flattening of the structure for BankTransactionCode and ProprietaryBankTransactionCode.
    - Additional information for an AddressLine, MerchantDetails and a running Balance.

Resources

Each of the Account and Transaction API resources are documented in sub-pages of this specification. Each resource is documented with:

- Endpoints
    - The API endpoints available for the resource.
- Data Model
    - Resource definition.
    - UML diagram.
    - Permissions as they relate to accessing the resource.
    - Data dictionary - which defines fields, re-usable classes, mandatory (1..1) or conditional (0..1) as defined in the Design Principles section, and enumerations.
- Usage Examples

Enumerations

Static Enumerations

Code Class	Name	Definition
OBAddressTypeCode	Business	Address is the business address.
OBAddressTypeCode	Correspondence	Address is the address where correspondence is sent.
OBAddressTypeCode	DeliveryTo	Address is the address to which delivery is to take place.
OBAddressTypeCode	MailTo	Address is the address to which mail is sent.
OBAddressTypeCode	POBox	Address is a postal office (PO) box.
OBAddressTypeCode	Postal	Address is the complete postal address.
OBAddressTypeCode	Residential	Address is the home address.
OBAddressTypeCode	Statement	Address is the address where statements are sent.
OBBalanceType1Code	ClosingAvailable	Closing balance of amount of money that is at the disposal of the account owner on the date specified.
OBBalanceType1Code	ClosingBooked	Balance of the account at the end of the pre-agreed account reporting period. It is the sum of the opening booked balance at the beginning of the period and all entries booked to the account during the pre-agreed account reporting period.
OBBalanceType1Code	ClosingCleared	Closing balance of amount of money that is cleared on the date specified.
OBBalanceType1Code	Expected	Balance, composed of booked entries and pending items known at the time of calculation, which projects the end of day balance if everything is booked on the account and no other entry is posted.
OBBalanceType1Code	ForwardAvailable	Forward available balance of money that is at the disposal of the account owner on the date specified.
OBBalanceType1Code	Information	Balance for informational purposes.
OBBalanceType1Code	InterimAvailable	Available balance calculated in the course of the account servicer's business day, at the time specified, and subject to further changes during the business day. The interim balance is calculated on the basis of booked credit and debit items during the calculation time/period specified.
OBBalanceType1Code	InterimBooked	Balance calculated in the course of the account servicer's business day, at the time specified, and subject to further changes during the business day. The interim balance is calculated on the basis of booked credit and debit items during the calculation time/period specified.
OBBalanceType1Code	InterimCleared	Cleared balance calculated in the course of the account servicer's business day, at the time specified, and subject to further changes during the business day.
OBBalanceType1Code	OpeningAvailable	Opening balance of amount of money that is at the disposal of the account owner on the date specified.
OBBalanceType1Code	OpeningBooked	Book balance of the account at the beginning of the account reporting period. It always equals the closing book balance from the previous report.
OBBalanceType1Code	OpeningCleared	Opening balance of amount of money that is cleared on the date specified.

OBBalanceType1Code	PreviouslyClosedBooked	Balance of the account at the previously closed account reporting period. The opening booked balance for the new period has to be equal to this balance. Usage: the previously booked closing balance should equal (inclusive date) the booked closing balance of the date it references and equal the actual booked opening balance of the current date.
OBCreditDebitCode	Credit	Operation is a credit
OBCreditDebitCode	Debit	Operation is a debit
OBEntryStatus1Code	Booked	Booked means that the transfer of money has been completed between account servicer and account owner Usage: Status Booked does not necessarily imply finality of money as this depends on other factors such as the payment system used, the completion of the end- to-end transaction and the terms agreed between account servicer and owner. Status Booked is the only status that can be reversed.
OBEntryStatus1Code	Pending	Booking on the account owner's account in the account servicer's ledger has not been completed. Usage: this can be used for expected items, or for items for which some conditions still need to be fulfilled before they can be booked. If booking takes place, the entry will be included with status Booked in subsequent account report or statement. Status Pending cannot be reversed.
OBExternalAccountSubType1Code	ChargeCard	Account sub-type is a Charge Card.
OBExternalAccountSubType1Code	CreditCard	Account sub-type is a Credit Card.
OBExternalAccountSubType1Code	CurrentAccount	Account sub-type is a Current Account.
OBExternalAccountSubType1Code	EMoney	Account sub-type is an EMoney.
OBExternalAccountSubType1Code	Loan	Account sub-type is a Loan.
OBExternalAccountSubType1Code	Mortgage	Account sub-type is a Mortgage.
OBExternalAccountSubType1Code	PrePaidCard	Account sub-type is a PrePaid Card.
OBExternalAccountSubType1Code	Savings	Account sub-type is a Savings.
OBExternalAccountType1Code	Business	Account type is for business.
OBExternalAccountType1Code	Personal	Account type is for personal.
OBExternalCardAuthorisationType1Code	ConsumerDevice	Card authorisation was via a Consumer Device Cardholder Verification Method (CDCVM).
OBExternalCardAuthorisationType1Code	Contactless	Card authorisation was via Contactless.
OBExternalCardAuthorisationType1Code	None	No card authorisation was used.
OBExternalCardAuthorisationType1Code	PIN	Card authorisation was via PIN.
OBExternalCardSchemeType1Code	AmericanExpress	AmericanExpress scheme.
OBExternalCardSchemeType1Code	Diners	Diners scheme.
OBExternalCardSchemeType1Code	Discover	Discover scheme.
OBExternalCardSchemeType1Code	MasterCard	MasterCard scheme.
OBExternalCardSchemeType1Code	VISA	VISA scheme.
OBExternalLimitType1Code	Available	The amount of credit limit available to the account holder
OBExternalLimitType1Code	Credit	The amount of a credit limit that has been agreed with the account holder

OBExternalLimitType1Code	Emergency	The amount of an arranged lending limit that can be borrowed on top of pre-agreed lending, that has been agreed with the account holder
OBExternalLimitType1Code	Pre-Agreed	The amount of an arranged lending limit that has been agreed with the account holder
OBExternalLimitType1Code	Temporary	The amount of a temporary lending limit that has been agreed with the account holder
OBExternalOfferType1Code	BalanceTransfer	Offer is a balance transfer.
OBExternalOfferType1Code	LimitIncrease	Offer is a limit increase.
OBExternalOfferType1Code	MoneyTransfer	Offer is a money transfer.
OBExternalOfferType1Code	Other	Offer is of an other type.
OBExternalOfferType1Code	PromotionalRate	Offer is a promotional rate.
OBExternalPartyType1Code	Delegate	Party that has delegated access.
OBExternalPartyType1Code	Joint	Party is a joint owner of the account.
OBExternalPartyType1Code	Sole	Party is a sole owner of the account.
OBExternalScheduleType1Code	Arrival	Scheduled payment date is specified as the arrival date for the recipient.
OBExternalScheduleType1Code	Execution	Scheduled payment date is specified as the execution date.
OBExternalStandingOrderStatus1Code	Active	The standing order is active.
OBExternalStandingOrderStatus1Code	Inactive	The standing order is inactive.
OBExternalStatementType1Code	AccountClosure	Final account closure statement.
OBExternalStatementType1Code	AccountOpening	First statement provided for an account.
OBExternalStatementType1Code	Annual	Annual statement report.
OBExternalStatementType1Code	Interim	Adhoc or customised statement period.
OBExternalStatementType1Code	RegularPeriodic	Regular pre-agreed reporting statement.

ISO Enumerations

These following ISO Enumerations are used in the Accounts APIs.

ISO Data Type	Fields	ISO Enumeration Values URL
Min3Max4Text	MerchantCategoryCode	https://www.iso.org/standard/33365.html
ActiveOrHistoricCurrencyCode	Currency	https://www.iso20022.org/external_code_list.page
CountryCode	Country	https://en.wikipedia.org/wiki/ISO_3166-1_alpha-2#Officially_assig
ExternalBankTransactionFamily1Code	BankTransactionCode/Code	https://www.iso20022.org/external_code_list.page
ExternalBankTransactionSubFamily1Code	BankTransactionCode/SubCode	https://www.iso20022.org/external_code_list.page

Namespaced Enumerations

The enumerated values specified by Open Banking are documented in Swagger specification and Namespaced Enumerations page.

Swagger Specification

The Swagger Specification for Account Information APIs can be downloaded from the following links:

- JSON
- YAML