Accounts API

Overview

Gigya’s Accounts API is the core of the Customer Identity (RaaS) platform, providing the basic building blocks of functionality. The API gives you complete flexibility in terms of design and maintaining full control of the server-side integration. The Accounts API provides access to the user account data stored by Gigya (see Profile Management for the precise account structure). The user data stored includes Gigya's predefined fields, social graph, and site's specific custom fields.

The Accounts APIs are supported both for client side (Web SDK) and server side (REST API). We encourage you to use server-side APIs whenever applicable.

The RaaS platform consists of a fully indexed cloud-hosted database that allows storing any user-related data alongside with social data that is au tomatically imported and stored.
Among the more commonly used are the following API methods:

- accounts.getAccountInfo - retrieves a user account.
- accounts.setAccountInfo - stores data in a specified user account.
- accounts.deleteAccount - deletes a user account.
- accounts.search - allows searching the Accounts Storage using simple SQL-like queries. Read more in the accounts.search page.
- accounts.getSchema - retrieves the current schema of the Accounts Storage.
- accounts.setSchema - allows specifying a schema for specified fields. The schema defines access rights and static data fields in the profile and site's custom fields.

In addition, Gigya provides the following web-based tools, giving your admin full control over the user database:

- Identity Access - a site admin tool for accessing and editing site users data.
- Identity Query Tool - an intuitive and dynamic tool for searching and retrieving users' profile data, enabling people without DB knowledge to create a search query using simple drop-down menus.

Major Features

- **Social Sync** - the user's social networks profile data, which is available after Social Login/Registration is automatically imported and stored in the Accounts Storage and available for searching. In addition, whenever your Facebook users update their profiles on Facebook, the changes will sync automatically to the Accounts Storage. Gigya automatically subscribes to the following Facebook fields: *birthday, books, education, email, first_name, hometown, last_name, likes, link, locale, location, movies, music, name, relationship_status, religion, verified, timezone* and *work*. Please note that Gigya complies with the social networks' platform policies.
- **Schemaless** - the storage is built with a dynamic-schema that can seamlessly process massive amounts of user data in an optimized way. Having a dynamic-schema means that you may store any custom data you have with no constraints on its structure. You don't have to know in advance how your custom data is going to look and it doesn't have to look the same for all objects. There is no need to go through schema creation or modification when the data structure changes.
- **Fully indexed** - the storage immediately indexes your data so you can search your entire user base using any combination of user attributes. Its SQL-like interface simplifies queries, providing a flexible way to retrieve user segments at scale. You don't need to plan your queries or create the appropriate index in advance.
- **Security and Performance** - Gigya’s platform is Safe Harbor Certified, ISO27001 compliant, regularly scanned by PCI-approved scanning vendors, proven in high-scale enterprise environments, and holds an industry leading uptime. Gigya also owns and operates its own hardware, with real-time data backup.
- **Data Import** - you may import legacy data from an existing database, as well as data collected via traditional user workflows such as registration and newsletter sign-up.

Note: The Accounts API is part of the Customer Identity package, which is a premium platform that requires separate activation. If it is not part of your site package please contact your Gigya account manager or contact us by filling in a support form on our site.
You can also access the support page by clicking "Support" on the upper menu of Gigya's site.

Unable to render {include} The included page could not be found.

Watch an Instructional Video

To watch a video about this subject, you can visit our Enablement portal with your approved SAP customer or partner ID (S user). Please visit the About page to find out how to get an S user.

Accounts REST API Error Codes and Messages

Gigya defines specific error codes and messages that may be received when using Accounts APIs. These errors are returned to indicate that some information is incorrect or missing. Occasionally, an error code is in fact an informational message and does not indicate an error. For a comprehensive list of Gigya error codes, see Response Codes and Errors.

The following section describes errors which are specifically related to the Accounts REST API implementation, including the reasons for each error, suggestions for handling these errors, and which APIs may return the specified error.

Account pending registration - 206001

When is this Error Returned?

The "Account pending registration" error (code 206001) is returned when you call a method that performs social login, and the registration process has not been finalized, or the schema defines fields as **required** and one or more of these fields are missing from the user profile or data.

What's the Expected Next Step?

If the schema defines fields that are required and one or more of these fields are missing from the user profile or data, call accounts.setAccountInfo.

If the registration process has not been finalized, call accounts.finalizeRegistration.

Which APIs May Return this Error?

The APIs that may return this error are: accounts.login, accounts.register, and accounts.socialLogin.

Account pending verification - 206002

When is this Error Returned?

- The "Account pending verification" error (code 206002) is returned when the account has already been verified, and a user tries to log in with a loginID (usually an email address) that we have not yet verified that actually belongs to this person.
- When the accountOptions policy states that ***verifyEmail*** is "true", the account must be validated by using the available email addresses. When the policy states that ***allowUnverifiedLogin*** is "false", users are not allowed to login before they have verified their emails. So, in this case, when a user tries to login, and his account has not been verified yet, and ***verifyEmail*** is "true" in the policy and ***allowUnverifiedLogin*** is "false" in the policy, the "Account pending verification" error is returned.

What's the Expected Next Step?

Call accounts.resendVerificationCode to resend a validation email to the unverified addresses associated with the account. The email format is according to the templates defined in the policy.

Which APIs May Return this Error?

The APIs that may return this error are: accounts.login, accounts.linkAccounts, accounts.finalizeRegistration, and accounts.socialLogin.

Account missing loginID - 206003

When is this Error Returned?

When the registration policy states that *requireLoginID* is "true", a loginID is required when a user uses Social Login to register to the site, so the "Account missing loginID" error is returned (error code 206003) if ***requireLoginID*** is configured in the registration policy and there are no login identifiers or a password associated with the account.

What's the Expected Next Step?

Call accounts.register to register the new user to your site, in accordance with the predefined site Policies and the Schema of the Accounts Storage.

Which APIs May Return this Error?

The API that may return this error is: accounts.socialLogin.

Unique identifier exists - 400003

When is this Error Returned?

The "Unique identifier exists" error (code 400003) is returned w hen the email or the username already exist in the accounts database when a user tries to register or to set the account info.

What's the Expected Next Step?

Call the API method again with a different identifier that does not exist in the account database.

Which APIs May Return this Error?

The APIs that may return this error are: accounts.register and accounts.setAccountInfo.

CAPTCHA verification failed - 400021

When is this Error Returned?

The "CAPTCHA verification failed" error (code 400021) is returned when the registration policy states that ***requireCaptcha*** is "true", and the CAPTCHA verification fails when a user tries to register.

What's the Expected Next Step?

Call the API method again with a new CAPTCHA.

Which APIs May Return this Error?

The API that may return this error is: accounts.register.

Account disabled - 403041

When is this Error Returned?

The "Account disabled" error (code 403041)) is returned when a user tries to login and the account is disabled.

Which APIs May Return this Error?

The API that may return this error is: accounts.login.

Invalid loginID - 403042

When is this Error Returned?

The "Invalid LoginID" error (code 403042) is returned when a user tries to perform an action that requires a login identifier (username or email) and the login ID doesn't exist in our accounts database. It is also returned if the password that is passed in the API is incorrect.

Which APIs May Return this Error?

The APIs that may return this error are: accounts.login, account.linkAccounts, and accounts.setAccountInfo.

Login identifier exists - 403043

When is this Error Returned?

The "Login identifier exists" error (code 403043) is returned when email is defined as the Login Identifier and the email address received from the provider exists in the system but is associated with a different user.

What's the Expected Next Step?

Call accounts.getConflictingAccount, passing the regToken of the new identity, to receive the existing identity with the conflicting email address. Then call accounts.login to login with the existing identity, while setting the **loginMode** parameter to **"link"** and passing the **regToke n** of the new account. This will merge the new account and the existing one so that in the future, the user can log into the same account using either identity.

For more information and code examples see Linking Accounts Using API.

Which APIs May Return this Error?

The APIs that may return this error are: accounts.login, accounts.socialLogin, accounts.register, and accounts.setAccountInfo.

Underage user - 403044

When is this Error Returned?

The "Underage user" error is returned (error code 403044) when a user under the age of 13 tries to login.

Which APIs May Return this Error?

The API that may return this error is: accounts.login.

Pending password change - 403100

When is this Error Returned?

The "Pending password change" error (code 403100) is returned when a user attempts to login and the password change interval has passed since the last password change. The interval is set in the security.passwordChangeInterval policy.

What's the Expected Next Step?

Change the password.

Which APIs May Return this Error?

The API that may return this error is: accounts.login.

Account pending TFA verification - 403101

When is this Error Returned?

The "Account pending TFA verification" error (code 403101) is returned when a user tries to login or finalize registration and the policy (in the site Policies) requires 2-factor authentication, and the device is not in the verified device list for the account.

What's the Expected Next Step?

Complete the two-factor authentication.

Which APIs May Return this Error?

The APIs that may return this error are: accounts.login, accounts.socialLogin, accounts.finalizeRegistration, socialize.notifyLogin, and socialize.login.

Account pending TFA registration - 403102

When is this Error Returned?

The "Account pending TFA registration" error is returned (error code 403102) when a user tries to login or finalize registration and the policy (in the site Policies) requires 2-factor authentication, and the device is not in the verified device list for the account.

What's the Expected Next Step?

Complete the two-factor authentication.

Which APIs May Return this Error?

The APIs that may return this error are: accounts.login, accounts.socialLogin, accounts.finalizeRegistration, socialize.notifyLogin, and socialize.login.

Account pending recent login - 403110

When is this Error Returned?

The "Account pending recent login " error is returned (error code 403110) when there is an attempt to deactivate a TFA provider for a user (with accounts.tfa.deactivateProvider) or to register a user (with accounts.tfa.initTFA) and the user did not login through the device in the last few minutes.

What's the Expected Next Step?

Login through the device again.

Which APIs May Return this Error?

The APIs that may return this error are: accounts.tfa.deactivateProvider and accounts.tfa.initTFA.

Account temporarily locked out - 403120

When is this Error Returned?

The "Account temporarily locked out" error (code 403120) is returned when a user attempts to login and the account is locked out or the originating IP is locked out. This occurs after a set number of failed login attempts. The number is defined in the site's RBA Policy.

What's the Expected Next Step?

Wait until the lockout time has ended and login again.

Which APIs May Return this Error?

The API that may return this error is: accounts.login.

Login failed CAPTCHA required - 401020

When is this Error Returned?

The "Login failed captcha required" error (code 401020) is returned when a user attempts to login and the CAPTCHA threshold has been reached. The CAPTCHA threshold is set in the site Policies (*security .captcha.failedLoginThreshold* policy).

What's the Expected Next Step?

Login with the CAPTCHA challenge.

Which APIs May Return this Error?

The API that may return this error is: accounts.login.

Login failed wrong CAPTCHA - 401021

When is this Error Returned?

The "Login failed wrong captcha" error (code 401021) is returned when a user attempts to login and the CAPTCHA threshold has been reached and the provided CAPTCHA text is wrong. The CAPTCHA threshold is set in the site Policies (*security .captcha.failedLoginThreshold* policy).

What's the Expected Next Step?

Login again with the CAPTCHA challenge.

Which APIs May Return this Error?

The API that may return this error is: accounts.login.

Old password used - 401030

When is this Error Returned?

The "Old password used" error is returned (error code 401030) when a user attempts to login with a password that doesn't match the current password but does match the previous one. The server will return this error with the message saying that "the password was modified on" the date when the current password was set.

What's the Expected Next Step?

Use the current password.

Which APIs May Return this Error?

The API that may return this error is: accounts.login.

Validation errors - 40002X

The validationErrors object is an array of validation errors; each validation error is made up of an ***errorCode*** , a ***message*** , and a ***fieldName*** . The errorCode and message specify what error occured and the fieldName specifies which field had a validation error.

When an invalid field type is used, i.e. a string instead of an integer, or if a wrong format is used, i.e. an email address that is not in a correct format, an "Invalid parameter value" error is returned (error code 400006).

Some of the possible validation errors are:

- Schema validation failed - error code 400020
- Captcha verification failed - error code 400021
- Unique index validation error - error code 400022
- Invalid type validation error - error code 400023
- Dynamic fields validation error - error code 400024
- Write access validation error - error code 400025
- Invalid format validation error - error code 400026
- Required value validation error - error code 400027

When is this Error Returned?

A validation error is returned whenever there is a data validation error regarding one of the following required fields: *username, password, secretQuestion, secretAnswer, email*. For example,

```
validationErrors: [
    {
      "errorCode": 400006,
      "message": "invalid password - minimum length of 6 characters is
required",
      "fieldName": "password"
    },
    {
      "errorCode": 400006,
      "message": "wrong format",
      "fieldName": "profile.email"
    }
  ],
```

What's the Expected Next Step?

Call the API method again with the missing info.

Which APIs May Return this Error?

The APIs that may return this error are: accounts.register, and accounts.setAccountInfo.

Schema validation failed - 400020

When is this Error Returned?

The "Schema validation failed" error (error code 400020) is returned when trying to write to fields from the client-side. By default all the data fields in the DB have a "s*erverOnly*" write access, which means that only signed requests coming from the server are allowed to write into these fields. This is defined in the default accounts storage schema.

You will receive the following response when attempting to write into the profile fields, if you have not changed the schema:

```
"errorMessage": "Schema validation failed",
  "errorDetails": "write access mode violation: email",
  "statusCode": 400,
  "errorCode": 400020,
```

What's the Expected Next Step?

Call the accounts.setSchema API method to change the schema, and change the ***writeAccess*** to " *clientModify* " to allow unsigned requests coming from the client to write into this field and modify existing values.