Article

Identifying rate limits

Recognize the rate limits that apply to App Store Server API endpoints and handle them in your code.

Overview

The App Store Server API limits the number of requests that you can submit to each endpoint within a specified timespan. The request limits apply per app.

The following table lists the rate limits for each endpoint in the production environment, expressed in requests per second. The system enforces rate limits on an hourly basis.

Endpoint	Rate limit (per second)
`Get App Transaction Info`	50
`Get Transaction Info`	50
`Get Transaction History`	50
`Get Transaction History V1`	50
`Get All Subscription Statuses`	50
`Send Consumption Information`	50
`Get Notification History`	50
`Extend a Subscription Renewal Date`	20

Endpoint	Rate limit (per second)
`Set App Account Token`	20
`Look Up Order ID`	10
`Get Refund History`	10
`Get Refund History V1`	10
`Extend Subscription Renewal Dates for All Active Subscribers`	1
`Request a Test Notification`	1
`Get Test Notification Status`	1

The rate limits in the sandbox environment are 10% of the limits in the table above.

The App Store server may make adjustments to reduce or increase these rate limits as needed at any time.

Handle exceeded rate limits gracefully

If you exceed a per-hour limit, the API rejects the request with an HTTP 429 response, with a `RateLimitExceededError` in the body. Consider the following as you integrate the API:

- If you periodically call the API, throttle your requests to avoid exceeding the per-hour limit for an endpoint.
- Manage the HTTP 429 `RateLimitExceededError` in your error-handling process. For example, log the failure and queue the job to process it again at a later time.
- Check the `Retry-After` header if you receive the HTTP 429 error. This header contains a UNIX time, in milliseconds, that informs you when you can next send a request.

See Also

Essentials

Simplifying your implementation by using the App Store Server Library

Use Apple's open source library to create JSON Web Tokens (JWT) to authorize your calls, verify transactions, extract transaction identifiers from receipts, and more.

Creating API keys to authorize API requests

Create API keys you use to sign JSON Web Tokens and authorize API requests.

Generating JSON Web Tokens for API requests

Create JSON Web Tokens signed with your private key to authorize requests for App Store Server API and External Purchase Server API.

App Store Server API changelog

Learn about new features and updates in the App Store Server API.