Dropbox

API home

Developer blog

My apps

Getting started

Core concepts

Setup

Authentication

Files and

folders

Reference

Development

kits

REST API

Best practices

Branding guide

Terms and

conditions

Developer

forum

Dropbox API

support

Authentication
/request_token
/authorize
/access_token
Dropbox accounts
/account/info
Files and metadata
/files (GET)
/files_put
/files (POST)
/metadata
/delta
/revisions
/restore
/search
/shares
/media
/copy_ref
/thumbnails
/chunked_upload
/commit_chunked_upload
File operations
/fileops/copy
/fileops/create_folder
/fileops/delete
/fileops/move

REST API

The REST API is the underlying interface for all of our official Dropbox mobile apps and our SDKs. It's the most direct way to access the API. This reference document is designed for those interested in developing for platforms not supported by the SDKs or for those interested in exploring API features in detail.

General notes

SSL only
We require that all requests are done over SSL.

App folder access type
The default root level access type, **app folder** (as described in core concepts), is referenced in API URLs by its codename `sandbox`. This is the only place where such a distinction is made.

UTF-8 encoding
Every string passed to and from the Dropbox API needs to be UTF-8 encoded. For maximum compatibility, normalize to Unicode Normalization Form C

(NFC) before UTF-8 encoding.

Version numbers

The current version of our API is version 1. Most version 0 methods will work for the time being, but some of its methods risk being removed (most notably, the version 0 API methods `/token` and `/account`).

Date format

All dates in the API are strings in the following format:

```
"Sat, 21 Aug 2010 22:31:20 +0000"
```

In code format, which can be used in all programming languages that support `strftime` or `strptime`:

```
"%a, %d %b %Y %H:%M:%S %z"
```

Locale global request parameter

Dropbox uses the `locale` parameter to specify language settings of content responses. If your app supports any language other than English, insert the appropriate IETF language tag. When a supported language is specified, Dropbox will returned translated `size` and/or `user_error` fields (where applicable).

Error handling

Errors are returned using standard HTTP error code syntax. Any additional info is included in the body of the return call, JSON-formatted. Error codes not listed here are in the REST API methods listed below.

Standard API errors

Code	Description
400	Bad input parameter. Error message should indicate which one and why.
401	Bad or expired token. This can happen if the user or Dropbox revoked or expired an access token. To fix, you should re-authenticate the user.
403	Bad OAuth request (wrong consumer key, bad nonce, expired timestamp...). Unfortunately, re-authenticating the user won't help here.
404	File or folder not found at the specified path.
405	Request method not expected (generally should be GET or POST).
503	Your app is making too many requests and is being rate limited. 503s can trigger on a per-app or per-user basis.
507	User is over Dropbox storage quota.
5xx	Server error. Check our ops Twitter feed @DropboxOps.

Authentication methods

Dropbox uses OAuth to authenticate all API requests.

/oauth/request_token

Python
Java
Ruby

Description
Step 1 of authentication. Obtain an OAuth *request token* to be used for the rest of the authentication process.

This method corresponds to Obtaining an Unauthorized Request Token in the OAuth Core 1.0 specification.

URL Structure

```
https://api.dropbox.com/1/oauth/request_token
```

Versions
0, 1
Method
POST
Parameters
There are no Dropbox-specific parameters for this method. See Consumer Obtains a Request Token in the OAuth Core 1.0 specification for a description of the OAuth parameters used for fetching a request token. Since this method is on behalf of an unauthenticated user, no access token or secret should be involved when signing or sending the request.

Returns
A request token and the corresponding request token secret, URL-encoded. This token/secret pair is meant to be used with /oauth/access_token to complete the authentication process and cannot be used for any other API calls. See Service Provider Issues an Unauthorized Request Token in the OAuth Core 1.0 specification for additional discussion of the values returned when fetching a request token.

Sample response

```
oauth_token_secret=b9q1n5il4lcc&oauth_token=mh7an9dkrg59
```

/oauth/authorize
Python
Java
Ruby
Description
Step 2 of authentication. Applications should direct the user to /oauth/authorize. This isn't an API call per se, but rather a web endpoint that lets the user sign in to Dropbox and choose whether to grant the application the ability to access files on their behalf. The page served by /oauth/authorize should be presented to the user through their web browser. Without the user's authorization in this step, it isn't possible for your application to obtain an access token from /oauth/access_token.

This method corresponds to Obtaining User Authorization in the OAuth Core 1.0 specification.

URL Structure

```
https://www.dropbox.com/1/oauth/authorize
```

Note: This is the only step that requires an endpoint on www.dropbox.com. All other API requests are done via api.dropbox.com or api-content.dropbox.com. For security's sake, please do not embed this webpage in your web app (e.g. via an iframe). The user must be redirected to the Dropbox website over HTTPS for unhindered and untampered authorization. As soon as authorization is accepted, the user will be redirected back to your website or to the website specified using oauth_callback (more on that below). If the user declines to authorize your app, no redirect to your site will happen.

Versions
0, 1
Method
GET
Parameters

oauth_token required The request token obtained via
`/oauth/request_token`.
oauth_callback After the user authorizes an application, the user is redirected
to the application-served URL provided by this parameter.
locale If the locale specified is a supported language, Dropbox will direct
users to a translated version of the authorization website. See the notes
above for more information about supported locales.
Returns
Because the application doesn't call `/oauth/authorize` directly, there is no
direct return value. After the user authorizes the application, use its request
token to retrieve an access token via the `/oauth/access_token` API call. If
the `oauth_callback` parameter is omitted, the application must find some
other way of determining when the authorization step is complete. For
example, the application can have the user explicitly indicate to it that this
step is complete, but this flow may be less intuitive for users than the redirect
flow.

If `oauth_callback` is specified, the application will know authorization is
complete when the user is redirected to the given URL. Dropbox will provide
the following return value via GET parameters in the URL:

oauth_token The request token that was just authorized. The request token
secret isn't sent back.
uid The user's unique Dropbox ID.

/oauth/access_token
Python
Java
Ruby
Description
Step 3 of authentication. After the `/oauth/authorize` step is complete, the
application can call `/oauth/access_token` to acquire an access token.

This method corresponds to Obtaining an Access Token in the OAuth Core 1.0
specification.

URL Structure

```
https://api.dropbox.com/1/oauth/access_token
```

Versions
0, 1
Method
POST
Parameters
There are no Dropbox-specific parameters for this method. See Consumer
Requests an Access Token in the OAuth Core 1.0 specification for a
description of the parameters used for fetching an access token. Note that
the `oauth_token` and `oauth_token_secret` for this method are the
request token and request token secret obtained previously via
`/oauth/request_token`.

Returns
URL-encoded access token, access token secret, and Dropbox user id. Upon
return, the authorization process is now complete and the access token and
corresponding secret can be used to sign requests for the main API calls. See
Service Provider Grants an Access Token in the OAuth Core 1.0 specification
for additional discussion of the values returned when fetching an access
token. If your app is configured to work within an app folder, that folder is
also created during this step.

Sample response

```
oauth_token_secret=95grkd9na7hm&oauth_token=ccl4li5n1q9b&ui
```

Dropbox accounts

/account/info
Python
Java
Ruby
Description
Retrieves information about the user's account.
URL Structure

```
https://api.dropbox.com/1/account/info
```

Versions
0, 1
Method
GET
Parameters
locale Use to specify language settings for user error messages and other language specific text. See the notes above for more information about supported locales.
Returns
User account information.

Sample JSON response

```
{
    "referral_link": "https://www.dropbox.com/referrals/r1a
    "display_name": "John P. User",
    "uid": 12345678,
    "country": "US",
    "quota_info": {
        "shared": 253738410565,
        "quota": 107374182400000,
        "normal": 680031877871
    }
}
```

Return value definitions

field	description
referral_link	The user's referral link.
display_name	The user's display name.
uid	The user's unique Dropbox ID.
country	The user's two-letter country code, if available.
quota_info/normal	The user's used quota outside of shared folders (bytes).
quota_info/shared	The user's used quota in shared folders (bytes).
quota_info/quota	The user's total quota allocation (bytes).

Files and metadata

/files (GET)
Python
Java
Ruby
Description

Downloads a file. Note that this call goes to **api-content.dropbox.com** instead of api.dropbox.com.

URL Structure

```
https://api-content.dropbox.com/1/files/<root>/<path>
```

root The root relative to which `path` is specified. Valid values are `sandbox` and `dropbox`.
path The path to the file you want to retrieve.
Versions
0, 1
Method
GET
Parameter
rev The revision of the file to retrieve. This defaults to the most recent revision.
Returns
The specified file's contents at the requested revision.

The HTTP response contains the content metadata in JSON format within an `x-dropbox-metadata` header.

Errors
404 The file wasn't found at the specified path, or wasn't found at the specified `rev`.
Notes
This method also supports HTTP Range Retrieval Requests to allow retrieving partial file contents.

/files_put
Python
Java
Ruby
Description
Uploads a file using PUT semantics. Note that this call goes to **api-content.dropbox.com** instead of api.dropbox.com.

This method is in most cases simpler to use than /files (POST).

The preferred HTTP method for this call is **PUT**. For compatibility with browser environments, the **POST** HTTP method is also recognized.

Note: Providing a `Content-Length` header set to the size of the uploaded file is required so that the server can verify that it has received the entire file contents.

URL Structure

```
https://api-content.dropbox.com/1/files_put/<root>/<path>?p
```

root The root relative to which `path` is specified. Valid values are `sandbox` and `dropbox`.
path The full path to the file you want to write to. This parameter should *not* point to a folder.
param=val The URL-encoded parameters for this request. They cannot be sent in the request body.
Version
1
Method
PUT, POST
Request body
required The file contents to be uploaded. Since the entire PUT body will be

treated as the file, any parameters must be passed as part of the request URL. The request URL should be signed just as you would sign any other OAuth request URL.

Parameters
locale The metadata returned on successful upload will have its `size` field translated based on the given locale.
overwrite This value, either `true` (default) or `false`, determines what happens when there's already a file at the specified path. If `true`, the existing file will be overwritten by the new one. If `false`, the new file will be automatically renamed (for example, `test.txt` might be automatically renamed to `test (1).txt`). The new name can be obtained from the returned metadata.
parent_rev The revision of the file you're editing. If `parent_rev` matches the latest version of the file on the user's Dropbox, that file will be replaced. Otherwise, the new file will be automatically renamed (for example, `test.txt` might be automatically renamed to `test (conflicted copy).txt`. If you specify a revision that doesn't exist, the file will not save (error 400). Get the most recent `rev` by performing a call to [/metadata](#).
Returns
The metadata for the uploaded file. More information on the returned metadata fields are available [here](#).

Sample JSON response

```
{
    "size": "225.4KB",
    "rev": "35e97029684fe",
    "thumb_exists": false,
    "bytes": 230783,
    "modified": "Tue, 19 Jul 2011 21:55:38 +0000",
    "path": "/Getting_Started.pdf",
    "is_dir": false,
    "icon": "page_white_acrobat",
    "root": "dropbox",
    "mime_type": "application/pdf",
    "revision": 220823
}
```

Errors
411 Missing `Content-Length` header (this endpoint doesn't support HTTP chunked transfer encoding).
Notes
`/files_put` has a maximum file size limit of 150 MB and does not support uploads with chunked encoding. To upload larger files, use [/chunked_upload](#) instead.

/files (POST)
[Python](#)
[Java](#)
[Ruby](#)
Description
Uploads a file. Note that this call goes to **api-content.dropbox.com** instead of api.dropbox.com.

We recommend you use [/files_put](#) instead due to its simpler interface.

URL Structure

```
https://api-content.dropbox.com/1/files/<root>/<path>
```

root The root relative to which `path` is specified. Valid values are `sandbox`

and `dropbox`.

path The path to the folder the file should be uploaded to. This parameter should *not* point to a file.

Versions

0, 1

Method

POST

Request body

required The file contents to be uploaded. Since the entire POST body will be treated as the file, any parameters must be passed as part of the request URL. The request URL should be signed just as you would sign any other OAuth request URL.

Parameters

locale The metadata returned on successful upload will have its `size` field translated based on the given locale.

overwrite This value, either `true` (default) or `false`, determines what happens when there's already a file at the specified path. If `true`, the existing file will be overwritten by the new one. If `false`, the new file will be automatically renamed (for example, `test.txt` might be automatically renamed to `test (1).txt`). The new name can be obtained from the returned metadata.

parent_rev The revision of the file you're editing. If `parent_rev` matches the latest version of the file on the user's Dropbox, that file will be replaced. Otherwise, the new file will be automatically renamed (for example, `test.txt` might be automatically renamed to `test (conflicted copy).txt`). If you specify a revision that doesn't exist, the file will not save (error 400). Get the most recent `rev` by performing a call to /metadata.

Returns

The metadata for the uploaded file. More information on the returned metadata fields are available here.

Sample JSON response

```
{
    "size": "225.4KB",
    "rev": "35e97029684fe",
    "thumb_exists": false,
    "bytes": 230783,
    "modified": "Tue, 19 Jul 2011 21:55:38 +0000",
    "path": "/Getting_Started.pdf",
    "is_dir": false,
    "icon": "page_white_acrobat",
    "root": "dropbox",
    "mime_type": "application/pdf",
    "revision": 220823
}
```

Errors

400 The file extension is on Dropbox's ignore list (e.g. **thumbs.db** or **.ds_store**).

404 The `parent_rev` of the file wasn't found.

411 Chunked encoding was attempted for this upload, but is not supported for this method. (For chunked encoding, use /chunked_upload instead.)

Notes

`/files` has a maximum file size limit of 150 MB. To upload larger files, use /chunked_upload instead.

/metadata

Python
Java
Ruby

Description
Retrieves file and folder metadata.
URL Structure

```
https://api.dropbox.com/1/metadata/<root>/<path>
```

root The root relative to which `path` is specified. Valid values are `sandbox`
and `dropbox`.
path The path to the file or folder.
Versions
0, 1
Method
GET
Parameters
file_limit Default is 10,000 (max is 25,000). When listing a folder, the service
will not report listings containing more than the specified amount of files and
will instead respond with a 406 (Not Acceptable) status response.
hash Each call to `/metadata` on a folder will return a `hash` field, generated by
hashing all of the metadata contained in that response. On later calls to
`/metadata`, you should provide that value via this parameter so that if
nothing has changed, the response will be a 304 (Not Modified) status code
instead of the full, potentially very large, folder listing. This parameter is
ignored if the specified path is associated with a file or if `list=false`. A
folder shared between two users will have the same hash for each user.
list The strings `true` and `false` are valid values. `true` is the default. If `true`,
the folder's metadata will include a `contents` field with a list of metadata
entries for the contents of the folder. If `false`, the `contents` field will be
omitted.
include_deleted Only applicable when `list` is set. If this parameter is set to
true, then `contents` will include the metadata of deleted children. Note that
the target of the metadata call is always returned even when it has been
deleted (with `is_deleted` set to true) regardless of this flag.
rev If you include a particular revision number, then only the metadata for
that revision will be returned.
locale The metadata returned will have its `size` field translated based on the
given locale. For more information see above.
Returns
The metadata for the file or folder at the given <path>. If <path> represents
a folder and the `list` parameter is `true`, the metadata will also include a
listing of metadata for the folder's contents.

Sample JSON return value for a file

```json
{
    "size": "225.4KB",
    "rev": "35e97029684fe",
    "thumb_exists": false,
    "bytes": 230783,
    "modified": "Tue, 19 Jul 2011 21:55:38 +0000",
    "client_mtime": "Mon, 18 Jul 2011 18:04:35 +0000",
    "path": "/Getting_Started.pdf",
    "is_dir": false,
    "icon": "page_white_acrobat",
    "root": "dropbox",
    "mime_type": "application/pdf",
    "revision": 220823
}
```

Sample JSON return value for a folder when `list` parameter is set to `true.` If `list` is `false` the `contents` key will simply be omitted from the result.

```
{
    "size": "0 bytes",
    "hash": "37eb1ba1849d4b0fb0b28caf7ef3af52",
    "bytes": 0,
    "thumb_exists": false,
    "rev": "714f029684fe",
    "modified": "Wed, 27 Apr 2011 22:18:51 +0000",
    "path": "/Public",
    "is_dir": true,
    "icon": "folder_public",
    "root": "dropbox",
    "contents": [
        {
            "size": "0 bytes",
            "rev": "35c1f029684fe",
            "thumb_exists": false,
            "bytes": 0,
            "modified": "Mon, 18 Jul 2011 20:13:43 +0000",
            "client_mtime": "Wed, 20 Apr 2011 16:20:19 +000
            "path": "/Public/latest.txt",
            "is_dir": false,
            "icon": "page_white_text",
            "root": "dropbox",
            "mime_type": "text/plain",
            "revision": 220191
        }
    ],
    "revision": 29007
}
```

Return value definitions

field	description
size	A human-readable description of the file size (translated by `locale`).
bytes	The file size in bytes.
path	Returns the canonical path to the file or directory.
is_dir	Whether the given entry is a folder or not.
is_deleted	Whether the given entry is deleted (only included if deleted files are being returned).
rev	A unique identifier for the current revision of a file. This field is the same `rev` as elsewhere in the API and can be used to detect changes and avoid conflicts.
hash	A folder's `hash` is useful for indicating changes to the folder's contents in later calls to `/metadata`. This is roughly the folder equivalent to a file's `rev`.
thumb_exists	True if the file is an image can be converted to a thumbnail via the `/thumbnails` call.
icon	The name of the icon used to illustrate the file type in Dropbox's icon library.
modified	The last time the file was modified on Dropbox, in the standard date format (not included for the root folder).
client_mtime	For files, this is the modification time set by the desktop client when the file was added to Dropbox, in the standard date format. Since this time is not verified (the Dropbox server stores whatever the desktop client sends up), this should only be used for display purposes (such as sorting) and not, for example, to determine if a file has changed or not.
root	The root or top-level folder depending on your access level. All paths returned are relative to this root level. Permitted values are either `dropbox` or `app_folder`.
	A **deprecated** field that semi-uniquely identifies a file. Use

revision   rev instead.

Note: modified, rev, and revision aren't returned in the metadata for the root/top-level path.

Errors
304The folder contents have not changed (relies on hash parameter).
406There are too many file entries to return.

/delta
[Python](#)
[Java](#)
[Ruby](#)
Description
A way of letting you keep up with changes to files and folders in a user's Dropbox. You can periodically call /delta to get a list of "delta entries", which are instructions on how to update your local state to match the server's state.
URL Structure

```
https://api.dropbox.com/1/delta
```

Version
1
Method
POST
Parameters
cursor A string that is used to keep track of your current state. On the next call pass in this value to return delta entries that have been recorded since the cursor was returned.
locale The metadata returned will have its size field translated based on the given locale. For more information see [above](#).
Returns
A JSON object with four fields:

entries A list of "delta entries" (described below).
reset If true, clear your local state before processing the delta entries. reset is always true on the initial call to /delta (i.e. when no cursor is passed in). Otherwise, it is true in rare situations, such as after server or account maintenance, or if a user deletes their app folder.
cursor A string that encodes the latest information that has been returned. On the next call to /delta, pass in this value.
has_more If true, then there are more entries available; you can call /delta again immediately to retrieve those entries. If 'false', then wait for at least five minutes (preferably longer) before checking again.
Delta Entries
Each delta entry is a 2-item list of one of the following forms:

[<path>, <metadata>] - Indicates that there is a file/folder at the given

path. You should add the entry to your local path. The metadata value is the

same as what would be returned by the /metadata call, except folder

metadata doesn't have hash or contents fields. To correctly process delta

entries:

If the new entry includes parent folders that don't yet exist in your local

state, create those parent folders in your local state.

If the new entry is a file, replace whatever your local state has at path

with the new entry.

If the new entry is a folder, check what your local state has at `<path>`. If it's a file, replace it with the new entry. If it's a folder, apply the new `<metadata>` to the folder, but do not modify the folder's children.

`[<path>, null]` - Indicates that there is no file/folder at the given path. To update your local state to match, anything at path and all its children should be deleted. Deleting a folder in your Dropbox will sometimes send down a single deleted entry for that folder, and sometimes separate entries for the folder and all child paths. If your local state doesn't have anything at path, ignore this entry.

Note: Dropbox treats file names in a case-insensitive but case-preserving way. To facilitate this, the `<path>` values above are lower-cased versions of the actual path. The `<metadata>` value has the original case-preserved path.

/revisions
[Python](#)
[Java](#)
[Ruby](#)
Description
Obtains metadata for the previous revisions of a file.

Only revisions up to thirty days old are available (or more if the Dropbox user has [Pack-Rat](#)). You can use the revision number in conjunction with the [/restore](#) call to revert the file to its previous state.

URL Structure

```
https://api.dropbox.com/1/revisions/<root>/<path>
```

root The root relative to which `path` is specified. Valid values are `sandbox` and `dropbox`.
path The path to the file.
Version
1
Method
GET
Parameters
rev_limit Default is 10. Max is 1,000. When listing a file, the service will not report listings containing more than the amount specified and will instead respond with a 406 (Not Acceptable) status response.
locale The metadata returned will have its `size` field translated based on the given locale. For more information see [above](#).
Returns
A list of all revisions formatted just like file metadata. More information on the returned metadata fields are available [here](#).

Sample JSON return value

```
[
{
    "is_deleted": true,
    "revision": 4,
    "rev": "40000000d",
    "thumb_exists": false,
    "bytes": 0,
    "modified": "Wed, 20 Jul 2011 22:41:09 +0000",
    "path": "/hi2",
```

```
        "is_dir": false,
        "icon": "page_white",
        "root": "app_folder",
        "mime_type": "application/octet-stream",
        "size": "0 bytes"
    },
    {
        "revision": 1,
        "rev": "10000000d",
        "thumb_exists": false,
        "bytes": 3,
        "modified": "Wed, 20 Jul 2011 22:40:43 +0000",
        "path": "/hi2",
        "is_dir": false,
        "icon": "page_white",
        "root": "app_folder",
        "mime_type": "application/octet-stream",
        "size": "3 bytes"
    }
    ]
```

Errors
406Too many file entries to return.

/restore
[Python](#)
[Java](#)
[Ruby](#)
Description
Restores a file path to a previous revision.

Unlike downloading a file at a given revision and then re-uploading it, this call is atomic. It also saves a bunch of bandwidth.

URL Structure

```
https://api.dropbox.com/1/restore/<root>/<path>
```

root The root relative to which `path` is specified. Valid values are `sandbox` and `dropbox`.
path The path to the file.
Version
1
Method
POST
Parameters
rev required The revision of the file to restore.
locale The metadata returned will have its `size` field translated based on the given locale. For more information see [above](#).
Returns
The metadata of the restored file. More information on the returned metadata fields are available [here](#).

Sample JSON response

```
{
    "is_deleted": true,
    "revision": 4,
    "rev": "40000000d",
    "thumb_exists": false,
    "bytes": 0,
    "modified": "Wed, 20 Jul 2011 22:41:09 +0000",
```

```
            "path": "/hi2",
            "is_dir": false,
            "icon": "page_white",
            "root": "sandbox",
            "mime_type": "application/octet-stream",
            "size": "0 bytes"
    }
```

Errors
404Unable to find the revision at that path

/search
Python
Java
Ruby
Description
Returns metadata for all files and folders whose filename contains the given search string as a substring.

Searches are limited to the folder path and its sub-folder hierarchy provided in the call.

URL Structure

```
https://api.dropbox.com/1/search/<root>/<path>
```

root The root relative to which `path` is specified. Valid values are `sandbox` and `dropbox`.
path The path to the folder you want to search from.
Versions
0, 1
Methods
GET, POST
Parameters
query required The search string. Must be at least three characters long.
file_limit The maximum and default value is 1,000. No more than `file_limit` search results will be returned.
include_deleted If this parameter is set to true, then files and folders that have been deleted will also be included in the search.
locale The metadata returned will have its `size` field translated based on the given locale. For more information see above.
Returns
List of metadata entries for any matching files and folders. More information on the returned metadata fields are available here.

Sample JSON return value for a search for `.txt` in Dropbox/Public folder

```
[
    {
        "size": "0 bytes",
        "rev": "35c1f029684fe",
        "thumb_exists": false,
        "bytes": 0,
        "modified": "Mon, 18 Jul 2011 20:13:43 +0000",
        "path": "/Public/latest.txt",
        "is_dir": false,
        "icon": "page_white_text",
        "root": "dropbox",
        "mime_type": "text/plain",
        "revision": 220191
    }
]
```

/shares
Python
Java
Ruby

Description
Creates and returns a Dropbox link to files or folders users can use to view a preview of the file in a web browser.

Note: Links created after April 23rd, 2012 no longer expire after thirty days.

URL Structure

```
https://api.dropbox.com/1/shares/<root>/<path>
```

root The root relative to which `path` is specified. Valid values are `sandbox` and `dropbox`.
path The path to the file or folder you want to link to.
Versions
0, 1
Method
POST
Parameters
locale Use to specify language settings for user error messages and other language specific text. See the notes above for more information about supported locales.
short_url When `true` (default), the url returned will be shortened using the Dropbox url shortener. If `false`, the url will link directly to the file's preview page.
Returns
A Dropbox link to the given path. The link can be used publicly and directs to a preview page of the file. For compatibility reasons, it returns the link's expiration date in Dropbox's usual date format. All links are currently set to expire far enough in the future so that expiration is effectively not an issue.

Sample JSON return value for a file

```
{
    "url": "http://db.tt/APqhX1",
    "expires": "Tue, 01 Jan 2030 00:00:00 +0000"
}
```

/media
Python
Java
Ruby

Description
Returns a link directly to a file.

Similar to `/shares`. The difference is that this bypasses the Dropbox webserver, used to provide a preview of the file, so that you can effectively stream the contents of your media.

URL Structure

```
https://api.dropbox.com/1/media/<root>/<path>
```

root The root relative to which `path` is specified. Valid values are `sandbox` and `dropbox`.
path The path to the media file you want a direct link to.
Version
1

Method
POST
Parameters
locale Use to specify language settings for user error messages and other language specific text. See the notes above for more information about supported locales.
Returns
A url that serves the media directly. Also returns the link's expiration date in Dropbox's usual date format.

Sample JSON return value for a file

```
{
    'url': 'http://www.dropbox.com/s/m/a2mbDa2',
    'expires': 'Fri, 16 Sep 2011 01:01:25 +0000'
}
```

Notes
The /media link expires after four hours, allotting enough time to stream files, but not enough to leave a connection open indefinitely.

/copy_ref
Python
Java
Ruby
Description
Creates and returns a copy_ref to a file. This reference string can be used to copy that file to another user's Dropbox by passing it in as the from_copy_ref parameter on /fileops/copy.

URL Structure

```
https://api.dropbox.com/1/copy_ref/<root>/<path>
```

root The root relative to which path is specified. Valid values are sandbox and dropbox.
path The path to the file you want a copy_ref to refer to.
Versions
1
Method
GET
Returns
A copy_ref to the specified file. For compatibility reasons, it returns the link's expiration date in Dropbox's usual date format. All links are currently set to expire far enough in the future so that expiration is effectively not an issue.

Sample JSON return value for a file

```
{
    "copy_ref": "z1X6ATl6aWtzOGq0c3g5Ng",
    "expires": "Fri, 31 Jan 2042 21:01:05 +0000"
}
```

/thumbnails
Python
Java
Ruby
Description
Gets a thumbnail for an image. Note that this call goes to **api-content.dropbox.com** instead of api.dropbox.com.

URL Structure

```
https://api-content.dropbox.com/1/thumbnails/<root>/<path>
```

root The root relative to which `path` is specified. Valid values are `sandbox` and `dropbox`.
path The path to the image file you want to thumbnail.
Versions
0, 1
Method
GET
Parameters
format `jpeg` (default) or `png`. For images that are photos, `jpeg` should be preferred, while `png` is better for screenshots and digital art.
size One of the following values (default: `s`):

valuedimensions (px)
`xs`   32x32
`s`   64x64
`m`   128x128
`l`   640x480
`xl`   1024x768

Returns
A thumbnail of the specified image's contents. The image returned may be larger or smaller than the size requested, depending on the size and aspect ratio of the original image.

The HTTP response contains the content metadata in JSON format within an `x-dropbox-metadata` header.

Errors
404 The file path wasn't found or the file extension doesn't allow conversion to a thumbnail.
415 The image is invalid and cannot be converted to a thumbnail.
Notes
This method currently supports files with the following file extensions: "jpg", "jpeg", "png", "tiff", "tif", "gif", and "bmp".

Photos that are larger than 20MB in size will not be converted to a thumbnail.

/chunked_upload
Python
Java
Ruby
Description
Uploads large files to Dropbox in mulitple chunks. Also has the ability to resume if the upload is interrupted. This allows for uploads larger than the `/files` and `/files_put` maximum of 150 MB.

Typical usage:

1. Send a PUT request to `/chunked_upload` with the first chunk of the file without setting upload_id, and receive an upload_id in return.
2. Repeatedly PUT subsequent chunks using the upload_id to identify the upload in progress and an offset representing the number of bytes transferred so far.
3. After each chunk has been uploaded, the server returns a new offset representing the total amount transferred.
4. After the last chunk, POST to `/commit_chunked_upload` to complete the upload.

Chunks can be any size up to 150 MB. A typical chunk is 4 MB. Using large

chunks will mean fewer calls to `/chunked_upload` and faster overall throughput. However, whenever a transfer is interrupted, you will have to resume at the beginning of the last chunk, so it is often safer to use smaller chunks.

If the offset you submit does not match the expected offset on the server, the server will ignore the request and respond with a 400 error that includes the current offset. To resume upload, seek to the correct offset (in bytes) within the file and then resume uploading from that point.

A chunked upload can take a maximum of 24 hours before expiring.

URL Structure

```
https://api-content.dropbox.com/1/chunked_upload?param=val
```

param=val The URL-encoded parameters for this request. They cannot be sent in the request body.
Version
1
Method
PUT
Request body
required A chunk of data from the file being uploaded. If resuming, the chunk should begin at the number of bytes into the file that equals the **offset**.

Parameters
upload_idThe unique ID of the in-progress upload on the server. If left blank, the server will create a new upload session.
offset The byte offset of this chunk, relative to the beginning of the full file. The server will verify that this matches the offset it expects. If it does not, the server will return an error with the expected offset.
Returns
Sample JSON response

```
{
    "upload_id": "v0k84B0AT9fYkfMUp0sBTA",
    "offset": 31337,
    "expires": "Tue, 19 Jul 2011 21:55:38 +0000"
}
```

Errors
404The `upload_id` does not exist or has expired.
     The `offset` parameter does not match up with what the server expects.
400The body of the error response will be JSON similar to the above,
     indicating the correct `offset` to upload.

/commit_chunked_upload
Description
Completes an upload initiated by the `/chunked_upload` method. Saves a file uploaded via `/chunked_upload` to a user's Dropbox.

`/commit_chunked_upload` is similar to `/files_put`. The main difference is that while `/files_put` takes the file contents in the request body, `/commit_chunked_upload` takes a parameter `upload_id`, which is obtained when the file contents are uploaded via `/chunked_upload`.

Note that this call goes to **api-content.dropbox.com** instead of api.dropbox.com.

URL Structure

```
https://api-content.dropbox.com/1/commit_chunked_upload/<ro
```

root The root relative to which `path` is specified. Valid values are `sandbox` and `dropbox`.

path The full path to the file you want to write to. This parameter should *not* point to a folder.

Version
1

Method
POST

Parameters

locale The metadata returned on successful upload will have its `size` field translated based on the given locale.

overwrite This value, either `true` (default) or `false`, determines what happens when there's already a file at the specified path. If `true`, the existing file will be overwritten by the new one. If `false`, the new file will be automatically renamed (for example, `test.txt` might be automatically renamed to `test (1).txt`). The new name can be obtained from the returned metadata.

parent_rev The revision of the file you're editing. If `parent_rev` matches the latest version of the file on the user's Dropbox, that file will be replaced. Otherwise, the new file will be automatically renamed (for example, `test.txt` might be automatically renamed to `test (conflicted copy).txt`). If you specify a revision that doesn't exist, the file will not save (error 400). Get the most recent `rev` by performing a call to /metadata.

upload_id Used to identify the chunked upload session you'd like to commit.

Returns

The metadata for the uploaded file. More information on the returned metadata fields are available here.

Sample JSON response

```
{
    "size": "225.4KB",
    "rev": "35e97029684fe",
    "thumb_exists": false,
    "bytes": 230783,
    "modified": "Tue, 19 Jul 2011 21:55:38 +0000",
    "path": "/Getting_Started.pdf",
    "is_dir": false,
    "icon": "page_white_acrobat",
    "root": "dropbox",
    "mime_type": "application/pdf",
    "revision": 220823
}
```

Errors

400 Returned if the request does not contain an upload_id or if there is no chunked upload matching the given upload_id.

File operations

The various `fileops` calls provide the standard file operations. Files and folders can be moved, copied, or deleted. Folders can be created.

/fileops/copy
Python
Java
Ruby
Description

Copies a file or folder to a new location.
URL Structure

```
https://api.dropbox.com/1/fileops/copy
```

Versions
0, 1
Method
POST
Parameters
root required The root relative to which `from_path` and `to_path` are
specified. Valid values are `sandbox` and `dropbox`.
from_path Specifies the file or folder to be copied from relative to `root`.
to_path required Specifies the destination path, *including the new name for
the file or folder*, relative to `root`.
locale The metadata returned will have its `size` field translated based on the
given locale. For more information see above.
from_copy_ref Specifies a copy_ref generated from a previous /copy_ref call.
Must be used instead of the `from_path` parameter.
Returns
Metadata for the copy of the file or folder. More information on the returned
metadata fields are available here.

Sample JSON response

```
{
    "size": "15 bytes",
    "rev": "1f0a503351f",
    "thumb_exists": false,
    "bytes": 15,
    "modified": "Wed, 10 Aug 2011 18:21:29 +0000",
    "path": "/test1.txt",
    "is_dir": false,
    "icon": "page_white_text",
    "root": "dropbox",
    "mime_type": "text/plain",
    "revision": 496342
}
```

Errors
403 An invalid copy operation was attempted (e.g. there is already a file at the
given destination, or trying to copy a shared folder).
404 The source file wasn't found at the specified path.
406 Too many files would be involved in the operation for it to complete
successfully. The limit is currently 10,000 files and folders.

/fileops/create_folder
Python
Java
Ruby
Description
Creates a folder.
URL Structure

```
https://api.dropbox.com/1/fileops/create_folder
```

Versions
0, 1
Method
POST
Parameters

root required The root relative to which `path` is specified. Valid values are `sandbox` and `dropbox`.
path required The path to the new folder to create relative to `root`.
locale The metadata returned will have its `size` field translated based on the given locale. For more information see above.
Returns
Metadata for the new folder. More information on the returned metadata fields are available here.

Sample JSON response

```
{
    "size": "0 bytes",
    "rev": "1f477dd351f",
    "thumb_exists": false,
    "bytes": 0,
    "modified": "Wed, 10 Aug 2011 18:21:30 +0000",
    "path": "/new_folder",
    "is_dir": true,
    "icon": "folder",
    "root": "dropbox",
    "revision": 5023410
}
```

Errors
403There is already a folder at the given destination.

/fileops/delete
Python
Java
Ruby
Description
Deletes a file or folder.
URL Structure

```
https://api.dropbox.com/1/fileops/delete
```

Versions
0, 1
Method
POST
Parameters
root required The root relative to which `path` is specified. Valid values are `sandbox` and `dropbox`.
path required The path to the file or folder to be deleted.
locale The metadata returned will have its `size` field translated based on the given locale. For more information see above.
Returns
Metadata for the deleted file or folder. More information on the returned metadata fields are available here.

Sample JSON response

```
{
    "size": "0 bytes",
    "is_deleted": true,
    "bytes": 0,
    "thumb_exists": false,
    "rev": "1f33043551f",
    "modified": "Wed, 10 Aug 2011 18:21:30 +0000",
    "path": "/test .txt",
```

```
        "is_dir": false,
        "icon": "page_white_text",
        "root": "dropbox",
        "mime_type": "text/plain",
        "revision": 492341
}
```

Errors
404 No file wasn't found at the specified path.
406 Too many files would be involved in the operation for it to complete
    successfully. The limit is currently 10,000 files and folders.

/fileops/move
[Python](#)
[Java](#)
[Ruby](#)
Description
Moves a file or folder to a new location.
URL Structure

```
https://api.dropbox.com/1/fileops/move
```

Versions
0, 1
Method
POST
Parameters
root required The root relative to which `from_path` and `to_path` are
specified. Valid values are `sandbox` and `dropbox`.
from_path required Specifies the file or folder to be moved from relative to
`root`.
to_path required Specifies the destination path, *including the new name for
the file or folder*, relative to `root`.
locale The metadata returned will have its `size` field translated based on the
given locale. For more information see [above](#).
Returns
Metadata for the moved file or folder. More information on the returned
metadata fields are available [here](#).

Sample JSON response

```
{
        "size": "15 bytes",
        "rev": "1e0a503351f",
        "thumb_exists": false,
        "bytes": 15,
        "modified": "Wed, 10 Aug 2011 18:21:29 +0000",
        "path": "/test2.txt",
        "is_dir": false,
        "icon": "page_white_text",
        "root": "dropbox",
        "mime_type": "text/plain",
        "revision": 496342
}
```

Errors
403 An invalid move operation was attempted (e.g. there is already a file at the
    given destination, or moving a shared folder into a shared folder).
404 The source file wasn't found at the specified path.
406 Too many files would be involved in the operation for it to complete
    successfully. The limit is currently 10,000 files and folders.

Dropbox

Install

Mobile

Pricing

Teams

Tour

PYXL

Community

Referrals

Twitter

Facebook

Developers

About us

Dropbox Blog

Our team

News

Jobs

Support

Help Center

Privacy & Terms

Copyright

Contact us

English