



---

# Salesforce CLI Command Reference

Salesforce, Spring '21





# CONTENTS

<a href="#">Salesforce CLI Command Reference</a> .....	1
alias Namespace .....	1
auth Namespace .....	4
config Namespace .....	13
force Namespace .....	17
Help for Salesforce CLI Commands .....	198
CLI Deprecation Policy .....	199



# SALESFORCE CLI COMMAND REFERENCE

The command reference contains information about the Salesforce CLI commands and their parameters. This version of the command reference includes details about version 7.81.0 of Salesforce CLI and the following plugin versions: `salesforcedx` plugin version 50.7.1 `@salesforce/plugin-config` plugin version 1.2.1 `@salesforce/plugin-alias` plugin version 1.1.3 `@salesforce/plugin-auth` plugin version 1.4.0

For information about installing Salesforce CLI, see the [Salesforce CLI Setup Guide](#).

For information about Salesforce CLI changes, see the [Salesforce CLI Release Notes](#).

## [alias Namespace](#)

Use the alias commands to manage username aliases.

## [auth Namespace](#)

Use the auth commands to authorize a Salesforce org for use with the Salesforce CLI.

## [config Namespace](#)

Use the config commands to view and set your Salesforce CLI configuration values. Set your default Dev Hub and scratch org, and your default instance URL, either globally or at the project level.

## [force Namespace](#)

Commands to develop on the Salesforce Platform.

## [Help for Salesforce CLI Commands](#)

The `-h` | `--help` parameter shows details about Salesforce CLI topics and their commands.

## [CLI Deprecation Policy](#)

Salesforce deprecates CLI commands and parameters when, for example, the underlying API changes.

## alias Namespace

---

Use the alias commands to manage username aliases.

### [alias:list](#)

List username aliases for the Salesforce CLI.

### [alias:set](#)

Set username aliases for the Salesforce CLI.

### [alias:unset](#)

Unsets aliases for the Salesforce CLI.

## **alias:list**

List username aliases for the Salesforce CLI.

## Command Syntax

```
sfdx alias:list  
  [--json]  
  [--loglevel LOGLEVEL]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

## Aliases for **alias:list**

```
force:alias:list
```

## **alias:set**

Set username aliases for the Salesforce CLI.

## Help for **alias:set**

You can associate an alias with only one username at a time. If you've set an alias multiple times, the alias points to the most recent username.

## Examples for **alias:set**

```
sfdx alias:set YourAlias=username@example.com
```

```
sfdx alias:set YourAlias=username@example.com YourOtherAlias=devhub@example.com
```

## Command Syntax

```
sfdx alias:set  
  [--json]  
  [--loglevel LOGLEVEL]
```

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

## Aliases for `alias:set`

```
force:alias:set
```

## `alias:unset`

Unsets aliases for the Salesforce CLI.

## Command Syntax

**sfdx alias:unset**

[--json]

[--loglevel LOGLEVEL]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

## auth Namespace

---

Use the auth commands to authorize a Salesforce org for use with the Salesforce CLI.

### [auth:list](#)

List auth connection information.

### [auth:logout](#)

Log out from authorized orgs.

### [device Commands](#)

Authorize an org using a device code.

### [jwt Commands](#)

Authorize an org using JWT.

### [sfdxurl Commands](#)

Authorize an org using sfdxurl.

### [web Commands](#)

Authorize an org using a web browser.

## auth:list

List auth connection information.

## Command Syntax

```
sfdx auth:list  
  [--json]  
  [--loglevel LOGLEVEL]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn



## Aliases for `auth:list`

```
force:auth:list
```

## `auth:logout`

Log out from authorized orgs.

## Help for `auth:logout`

By default, this command logs you out from your default scratch org.

## Examples for `auth:logout`

```
sfdx auth:logout -u me@my.org
```

```
sfdx auth:logout -a
```

```
sfdx auth:logout -p
```

## Command Syntax

**sfdx auth:logout**

`[--json]`

`[--loglevel LOGLEVEL]`

`[-u TARGETUSERNAME]`

`[--apiversion APIVERSION]`

`[-a]`

`[-p]`

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-a | --all**

Optional

Includes all authenticated orgs: for example, Dev Hubs, sandboxes, DE orgs, and expired, deleted, and unknown-status scratch orgs.

Type: boolean

**-p | --noprompt**

Optional

Do not prompt for confirmation.

Type: boolean

## Aliases for `auth:logout`

```
force:auth:logout
```

## device Commands

Authorize an org using a device code.

[auth:device:login](#)

Authorize an org using a device code.

### **auth:device:login**

Authorize an org using a device code.

### Examples for `auth:device:login`

```
sfdx auth:device:login -d -a TestOrg1
```

```
sfdx auth:device:login -i <OAuth client id>
```

```
sfdx auth:device:login -r https://test.salesforce.com
```

### Command Syntax

**sfdx auth:device:login**

`[--json]`

`[--loglevel LOGLEVEL]`

`[-i CLIENTID]`

`[-r INSTANCEURL]`

[-d]

[-s]

[-a SETALIAS]

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-i | --clientid CLIENTID**

Optional

OAuth client ID (sometimes called the consumer key).

Type: string

### **-r | --instanceurl INSTANCEURL**

Optional

The login URL of the instance the org lives on.

Type: url

### **-d | --setdefaultdevhubusername**

Optional

Set the authenticated org as the default dev hub org for scratch org creation.

Type: boolean

### **-s | --setdefaultusername**

Optional

Set the authenticated org as the default username that all commands run against.

Type: boolean

### **-a | --setalias SETALIAS**

Optional

Set an alias for the authenticated org.

Type: string

## Aliases for **auth:device:login**

```
force:auth:device:login
```

## jwt Commands

Authorize an org using JWT.

`auth:jwt:grant`

Authorize an org using the JWT flow.

### auth:jwt:grant

Authorize an org using the JWT flow.

### Help for auth:jwt:grant

Use a certificate associated with your private key that has been uploaded to a personal connected app.

If you specify an `--instanceurl` value, this value overrides the `sfdcLoginUrl` value in your `sfdx-project.json` file. To specify a My Domain URL, use the format `MyDomainName.my.salesforce.com` (not `MyDomainName.lightning.force.com`).

### Examples for auth:jwt:grant

```
sfdx auth:jwt:grant -u me@my.org -f <path to jwt key file> -i <OAuth client id>
```

```
sfdx auth:jwt:grant -u me@my.org -f <path to jwt key file> -i <OAuth client id> -s -a MyDefaultOrg
```

```
sfdx auth:jwt:grant -u me@acme.org -f <path to jwt key file> -i <OAuth client id> -r https://acme.my.salesforce.com
```

### Command Syntax

**sfdx auth:jwt:grant**

`[--json]`

`[--loglevel LOGLEVEL]`

`-u USERNAME`

`-f JWTKEYFILE`

`-i CLIENTID`

`[-r INSTANCEURL]`

`[-d]`

`[-s]`

`[-a SETALIAS]`

### Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --username USERNAME**

Required

Authentication username.

Type: string

**-f | --jwtkeyfile JWTKEYFILE**

Required

Path to a file containing the private key.

Type: filepath

**-i | --clientid CLIENTID**

Required

OAuth client ID (sometimes called the consumer key).

Type: string

**-r | --instanceurl INSTANCEURL**

Optional

The login URL of the instance the org lives on.

Type: url

**-d | --setdefaultdevhubusername**

Optional

Set the authenticated org as the default dev hub org for scratch org creation.

Type: boolean

**-s | --setdefaultusername**

Optional

Set the authenticated org as the default username that all commands run against.

Type: boolean

**-a | --setalias SETALIAS**

Optional

Set an alias for the authenticated org.

Type: string

### Aliases for `auth:jwt:grant`

```
force:auth:jwt:grant
```

## sfdxurl Commands

Authorize an org using sfdxurl.

[auth:sfdxurl:store](#)

Authorize an org using an SFDX auth URL.

## **auth:sfdxurl:store**

Authorize an org using an SFDX auth URL.

### Help for **auth:sfdxurl:store**

Authorize a Salesforce org using an SFDX auth URL stored within a file. The file must have the format "force://<refreshToken>@<instanceUrl>" or "force://<clientId>:<clientSecret>:<refreshToken>@<instanceUrl>".

The file must contain only the URL or be a JSON file that has a top-level property named sfdxAuthUrl.

Use this command to get the SFDX auth URL for a Dev Hub org you have already authorized:

```
$ sfdx force:org:display -u <DevHub> --verbose
```

### Examples for **auth:sfdxurl:store**

```
sfdx auth:sfdxurl:store -f <path to sfdxAuthUrl file>
```

```
sfdx auth:sfdxurl:store -f <path to sfdxAuthUrl file> -s -a MyDefaultOrg
```

## Command Syntax

**sfdx auth:sfdxurl:store**

[--json]

[--loglevel LOGLEVEL]

-f SFDXURLFILE

[-d]

[-s]

[-a SETALIAS]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-f | --sfdxurlfile SFDXURLFILE**

Required

Path to a file containing the sfdx url.

Type: filepath

**-d | --setdefaultdevhubusername**

Optional

Set the authenticated org as the default dev hub org for scratch org creation.

Type: boolean

**-s | --setdefaultusername**

Optional

Set the authenticated org as the default username that all commands run against.

Type: boolean

**-a | --setalias SETALIAS**

Optional

Set an alias for the authenticated org.

Type: string

**Aliases for `auth:sfdxurl:store`**

```
force:auth:sfdxurl:store
```

## web Commands

Authorize an org using a web browser.

[auth:web:login](#)

Authorize an org using the web login flow.

**auth:web:login**

Authorize an org using the web login flow.

**Help for `auth:web:login`**To log in to a sandbox, set `--instanceurl` to `https://test.salesforce.com`.**Examples for `auth:web:login`**

```
sfdx auth:web:login -a TestOrg1
```

```
sfdx auth:web:login -i <OAuth client id>
```

```
sfdx auth:web:login -r https://test.salesforce.com
```

## Command Syntax

```
sfdx auth:web:login  
  [--json]  
  [--loglevel LOGLEVEL]  
  [-i CLIENTID]  
  [-r INSTANCEURL]  
  [-d]  
  [-s]  
  [-a SETALIAS]
```

## Parameters

- json**  
Optional  
Format output as JSON.  
Type: boolean
- loglevel LOGLEVEL**  
Optional  
The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.  
Type: enum  
Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL  
Default value: warn
- i | --clientid CLIENTID**  
Optional  
OAuth client ID (sometimes called the consumer key).  
Type: string
- r | --instanceurl INSTANCEURL**  
Optional  
The login URL of the instance the org lives on.  
Type: url
- d | --setdefaultdevhubusername**  
Optional  
Set the authenticated org as the default dev hub org for scratch org creation.  
Type: boolean
- s | --setdefaultusername**  
Optional  
Set the authenticated org as the default username that all commands run against.  
Type: boolean
- a | --setalias SETALIAS**  
Optional



Set an alias for the authenticated org.

Type: string

### Aliases for `auth:web:login`

```
force:auth:web:login
```

## config Namespace

---

Use the config commands to view and set your Salesforce CLI configuration values. Set your default Dev Hub and scratch org, and your default instance URL, either globally or at the project level.

#### [config:get](#)

Get config var values for given names.

#### [config:list](#)

Lists the config variables that the Salesforce CLI uses for various commands and tasks.

#### [config:set](#)

Sets the configuration variables that the Salesforce CLI uses for various commands and tasks.

#### [config:unset](#)

Unsets the local and global configuration variables for the Salesforce CLI.

### config:get

Get config var values for given names.

### Help for config:get

Gets the Salesforce CLI configuration values for your default scratch org, your default Dev Hub org, your default instance URL, or any combination of the three. To see your default scratch org username, include 'defaultusername'.

To see your default Dev Hub, include 'defaultdevhubusername'.

To see your default instance URL, include 'instanceUrl'.

To see the locations where your values are set, include the --verbose flag.

### Examples for config:get

```
sfdx config:get defaultusername
```

```
sfdx config:get defaultusername defaultdevhubusername instanceUrl
```

```
sfdx config:get defaultusername defaultdevhubusername --verbose
```

### Command Syntax

```
sfdx config:get  
[--json]
```

```
[--loglevel LOGLEVEL]
```

```
[--verbose]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **--verbose**

Optional

Emit additional command output to stdout.

Type: boolean

## Aliases for `config:get`

```
force:config:get
```

## **config:list**

Lists the config variables that the Salesforce CLI uses for various commands and tasks.

## Command Syntax

```
sfdx config:list
```

```
[--json]
```

```
[--loglevel LOGLEVEL]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

## Aliases for `config:list`

```
force:config:list
```

## `config:set`

Sets the configuration variables that the Salesforce CLI uses for various commands and tasks.

## Help for `config:set`

Local variables apply only to your current project. Global variables apply in any directory.

## Examples for `config:set`

```
sfdx config:set defaultusername=me@my.org defaultdevhubusername=me@myhub.org
```

```
sfdx config:set defaultdevhubusername=me@myhub.org -g
```

## Command Syntax

```
sfdx config:set
```

```
  [--json]
```

```
  [--loglevel LOGLEVEL]
```

```
  [-g]
```

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-g | --global**

Optional

Sets the configuration variables globally, so they can be used from any directory.

Type: boolean

## Aliases for **config:set**

```
force:config:set
```

## **config:unset**

Unsets the local and global configuration variables for the Salesforce CLI.

## Help for **config:unset**

Local variables apply only to your current project. Global variables apply in any directory.

## Examples for **config:unset**

```
sfdx config:unset defaultusername defaultdevhubusername
```

```
sfdx config:unset defaultdevhubusername -g
```

## Command Syntax

```
sfdx config:unset
```

```
  [--json]
```

```
  [--loglevel LOGLEVEL]
```

```
  [-g]
```

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-g | --global**

Optional

Unsets the configuration variables globally, so they can be used from any directory.

Type: boolean

# force Namespace

---

Commands to develop on the Salesforce Platform.

## [analytics Commands](#)

Work with analytics assets.

## [apex Commands](#)

Use the apex commands to create Apex classes, execute anonymous blocks, view your logs, run Apex tests, and view Apex test results.

## [auth Commands](#)

Use the auth commands to authorize a Salesforce org for use with the Salesforce CLI.

## [cmdt Commands](#)

Create and update custom metadata types and their records.

## [community Commands](#)

Use the community commands to create and publish a community, and view a list of available templates in you org.

## [data Commands](#)

Use the data commands to manipulate records in your org. Commands are available to help you work with various APIs. Import CSV files with the Bulk API. Export and import data that includes master-detail relationships with the SObject Tree Save API. Perform simple CRUD operations on individual records with the REST API.

## [lightning Commands](#)

Use the lightning commands to create Aura components and Lightning web components. As of API version 45.0, you can build Lightning components using two programming models: Lightning Web Components, and the original model, Aura Components. Lightning web components and Aura components can coexist and interoperate on a page.

## [limits Commands](#)

Use the limits commands to view your org's limits and how close you are to reaching them.

## [mdapi Commands](#)

Use the mdapi commands to retrieve and deploy Metadata API–formatted files that represent components in an org, or to convert Metadata API–formatted metadata into the source format used in Salesforce DX projects.

## [org Commands](#)

Use the org commands to manage the orgs you use with Salesforce CLI. Create and delete scratch orgs, list your created and authorized orgs, and open orgs in your browser.

## [package Commands](#)

Use the package commands to develop and install packages.

## [package1 Commands](#)

Use the package1 commands to create and view first-generation package versions in your Dev Hub org.

## [project Commands](#)

Use the project commands to set up a Salesforce DX project.

## [schema Commands](#)

Use the schema commands to view information about the standard and custom objects in your org.

[source Commands](#)

Use the source commands to push and pull source to and from your scratch orgs, to deploy and retrieve source to and from non-source-tracked orgs, to see synchronization changes between your project and scratch orgs, and to convert your source to the metadata format for Metadata API deployments.

[user Commands](#)

Use the user commands to perform user-related admin tasks.

[visualforce Commands](#)

Use the visualforce commands to create Visualforce pages and components.

## analytics Commands

Work with analytics assets.

[force:analytics:template:create](#)

Creates a simple Analytics template in the specified directory. If you don't explicitly set the API version, it defaults to the current API version. The associated metadata files are created.

### **force:analytics:template:create**

Creates a simple Analytics template in the specified directory. If you don't explicitly set the API version, it defaults to the current API version. The associated metadata files are created.

### Help for **force:analytics:template:create**

If not supplied, the apiversion, template, and outputdir use default values.

The outputdir can be an absolute path or relative to the current working directory.

Examples:

```
$ sfdx force:analytics:template:create -n myTemplate -d outputdir
```

### Command Syntax

```
sfdx force:analytics:template:create
```

```
  [--json]
```

```
  [--loglevel LOGLEVEL]
```

```
  [-d OUTPUTDIR]
```

```
  [--apiversion APIVERSION]
```

```
  -n TEMPLATENAME
```

### Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-d | --outputdir OUTPUTDIR**

Optional

The directory to store the newly created files. The location can be an absolute path or relative to the current working directory. The default is the current directory.

Type: string

Default value: .

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-n | --templatename TEMPLATENAME**

Required

The name of the Analytics template.

Type: string

## apex Commands

Use the apex commands to create Apex classes, execute anonymous blocks, view your logs, run Apex tests, and view Apex test results.

[force:apex:class:create](#)

Creates an Apex class in the specified directory or the current working directory. If you don't explicitly set the API version, it defaults to the current API version. The .cls file and associated metadata file are created.

[force:apex:execute](#)

Executes one or more lines of anonymous Apex code entered on the command line, or executes the code in a local file.

[force:apex:log:get](#)

Fetches the specified log or given number of most recent logs from the scratch org. .

[force:apex:log:list](#)

Run this command in a project to list the IDs and general information for all debug logs in your default org.

[force:apex:log:tail](#)

Activates debug logging and displays logs in the terminal. You can also pipe the logs to a file.

[force:apex:test:report](#)

Displays the test results for a specific test run.

[force:apex:test:run](#)

Runs Apex tests.

### `force:apex:trigger:create`

Creates an Apex trigger in the specified directory or the current working directory. If you don't explicitly set the API version, it defaults to the current API version. The .trigger file and associated metadata file are created.

## **force:apex:class:create**

Creates an Apex class in the specified directory or the current working directory. If you don't explicitly set the API version, it defaults to the current API version. The .cls file and associated metadata file are created.

### Help for **force:apex:class:create**

If not supplied, the `apiversion`, `template`, and `outputdir` use default values.

The `outputdir` can be an absolute path or relative to the current working directory.

Examples:

```
$ sfdx force:apex:class:create -n MyClass
```

```
$ sfdx force:apex:class:create -n MyClass -d classes
```

## Command Syntax

### **sfdx force:apex:class:create**

`[--json]`

`[--loglevel LOGLEVEL]`

`-n CLASSNAME`

`[-t TEMPLATE]`

`[-d OUTPUTDIR]`

`[--apiversion APIVERSION]`

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: `trace`, `debug`, `info`, `warn`, `error`, `fatal`, `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, `FATAL`

Default value: `warn`

### **-n | --classname CLASSNAME**

Required

The name of the new Apex class. The name can be up to 40 characters and must start with a letter.



Type: string

**-t | --template TEMPLATE**

Optional

The template to use to create the file. Supplied parameter values or default values are filled into a copy of the template.

Type: string

Permissible values are: ApexException, ApexUnitTest, DefaultApexClass, InboundEmailService

Default value: DefaultApexClass

**-d | --outputdir OUTPUTDIR**

Optional

The directory to store the newly created files. The location can be an absolute path or relative to the current working directory. The default is the current directory.

Type: string

Default value: .

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

## **force:apex:execute**

Executes one or more lines of anonymous Apex code entered on the command line, or executes the code in a local file.

### Help for **force:apex:execute**

If you don't run this command from within a Salesforce DX project, `---targetusername` is required.

To execute your code interactively, run this command with no parameters. At the prompt, enter all your Apex code; press CTRL-D when you're finished. Your code is then executed in a single execute anonymous request.

For more information, see "Anonymous Blocks" in the Apex Developer Guide.

### Examples for **force:apex:execute**

```
$ sfdx force:apex:execute -u testusername@salesforce.org -f ~/test.apex
```

```
$ sfdx force:apex:execute -f ~/test.apex
```

```
$ sfdx force:apex:execute
```

Start typing Apex code. Press the Enter key after each line, then press CTRL+D when finished.

## Command Syntax

**sfdx force:apex:execute**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-f APEXCODEFILE]

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-f | --apexcodefile APEXCODEFILE**

Optional

Path to a local file that contains Apex code.

Type: filepath

## **force:apex:log:get**

Fetches the specified log or given number of most recent logs from the scratch org. .

### Help for **force:apex:log:get**

To get the IDs for your debug logs, run "sfdx force:apex:log:list".

Use the --logid parameter to return a specific log.

Use the --number parameter to return the specified number of recent logs.

Use the --outputdir parameter to specify the directory to store the logs in.

Executing this command without parameters returns the most recent log.

## Examples for `force:apex:log:get`

```
$ sfdx force:apex:log:get -i <log id>
```

```
$ sfdx force:apex:log:get -i <log id> -u me@my.org
```

```
$ sfdx force:apex:log:get -n 2 -c
```

```
$ sfdx force:apex:log:get -d Users/Desktop/logs -n 2
```

## Command Syntax

### `sfdx force:apex:log:get`

```
[--json]  
[--loglevel LOGLEVEL]  
[-u TARGETUSERNAME]  
[--apiversion APIVERSION]  
[-i LOGID]  
[-n NUMBER]  
[-d OUTPUTDIR]
```

## Parameters

### `--json`

Optional

Format output as JSON.

Type: boolean

### `--loglevel LOGLEVEL`

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### `-u | --targetusername TARGETUSERNAME`

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### `--apiversion APIVERSION`

Optional

Override the API version used for API requests made by this command.

Type: string

### `-i | --logid LOGID`

Optional

Id of the log to display.

Type: id

**-n | --number NUMBER**

Optional

Number of most recent logs to display.

Type: number

**-d | --outputdir OUTPUTDIR**

Optional

The location can be an absolute path or relative to the current working directory. The default is the current directory.

Type: string

## **force:apex:log:list**

Run this command in a project to list the IDs and general information for all debug logs in your default org.

### Help for **force:apex:log:list**

To fetch a specific log from your org, obtain the ID from this command's output, then run the “sfdx force:apex:log:get” command.

### Examples for **force:apex:log:list**

```
$ sfdx force:apex:log:list
```

```
$ sfdx force:apex:log:list -u me@my.org
```

## Command Syntax

**sfdx force:apex:log:list**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

## **force:apex:log:tail**

Activates debug logging and displays logs in the terminal. You can also pipe the logs to a file.

### Help for **force:apex:log:tail**

Tails logs from your target org for 30 minutes.

If a DEVELOPER\_LOG trace flag does not exist, this command creates one in the target org.

If the active trace flag's expiration date is within this command's timeout window, the command sets the trace flag's expiration date to 30 minutes from the current time.

The --debuglevel parameter assigns a debug level to the active DEVELOPER\_LOG trace flag.

Use --skiptraceflag to skip trace flag setup, including setting expiration date and debug level. Include this flag only if there is an active user-based trace flag for your user.

The --json parameter emits log lines in JSON, but does not follow the standard Salesforce CLI JSON format (which includes status and result values).

Examples:

```
$ sfdx force:apex:log:tail
```

```
$ sfdx force:apex:log:tail --debuglevel MyDebugLevel
```

```
$ sfdx force:apex:log:tail -c -s
```

## Command Syntax

**sfdx force:apex:log:tail**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-c]

[-d DEBUGLEVEL]

[-s]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-c | --color**

Optional

Applies default colors to noteworthy log lines.

Type: boolean

**-d | --debuglevel DEBUGLEVEL**

Optional

Debug level to set on the DEVELOPER\_LOG trace flag for your user.

Type: string

**-s | --skiptraceflag**

Optional

Skips trace flag setup. Assumes that a trace flag and debug level are fully set up.

Type: boolean

## **force:apex:test:report**

Displays the test results for a specific test run.

## Help for **force:apex:test:report**

Displays test results for an enqueued or completed asynchronous Apex test run.

Examples:

```
$ sfdx force:apex:test:report -i <test run id>
```

```
$ sfdx force:apex:test:report -i <test run id> -r junit
```

```
$ sfdx force:apex:test:report -i <test run id> -c --json
```

## Command Syntax

### **sfdx force:apex:test:report**

[--json]

[--loglevel LOGLEVEL]

[-r RESULTFORMAT]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

-i TESTRUNID

[-c]

[-d OUTPUTDIR]

[-w WAIT]

[--verbose]

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-r | --resultformat RESULTFORMAT**

Optional

Format to use when displaying results. If you also specify the --json flag, --json overrides this parameter.

Type: enum

Permissible values are: human, tap, junit, json

Default value: human

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-i | --testrunid TESTRUNID**

Required

The ID of test run.

Type: id

**-c | --codecoverage**

Optional

Retrieves code coverage results.

Type: boolean

**-d | --outputdir OUTPUTDIR**

Optional

Directory to store test run files.

Type: directory

**-w | --wait WAIT**

Optional

Sets the streaming client socket timeout, in minutes. If the streaming client socket has no contact from the server for a number of minutes, the client exits. Specify a longer wait time if timeouts occur frequently.

Type: minutes

Default value: 6 minutes

**--verbose**

Optional

Displays Apex test processing details. If JSON format is specified, processing details aren't displayed.

Type: boolean

## **force:apex:test:run**

Runs Apex tests.

### Help for **force:apex:test:run**

By default, runs all Apex tests in the org's namespace.

To run specific test classes, specify class names or suite names, or set a `--testlevel` value.

To run specific test methods, use `--tests`.



Examples:

```
$ sfdx force:apex:test:run
```

```
$ sfdx force:apex:test:run -n "MyClassTest,MyOtherClassTest" -r human
```

```
$ sfdx force:apex:test:run -s "MySuite,MyOtherSuite" -c --json
```

```
$ sfdx force:apex:test:run -t
"MyClassTest.testCoolFeature,MyClassTest.testAwesomeFeature,AnotherClassTest,namespace.TheirClassTest.testThis"
-r human
```

```
$ sfdx force:apex:test:run -l RunLocalTests -d <path to outputdir> -u me@my.org
```

## Command Syntax

### **sfdx force:apex:test:run**

```
[--json]
[--loglevel LOGLEVEL]
[-r RESULTFORMAT]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
[-n CLASSNAMES]
[-s SUITENAMES]
[-t TESTS]
[-c]
[-d OUTPUTDIR]
[-l TESTLEVEL]
[-w WAIT]
[-y]
[--verbose]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-r | --resultformat RESULTFORMAT**

Optional

Format to use when displaying results. If you also specify the --json flag, --json overrides this parameter.

Type: enum

Permissible values are: human, tap, junit, json

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-n | --classnames CLASSNAMES**

Optional

Comma-separated list of Apex test class names to run. You can specify only one of classnames, suite names, or tests.

Type: array

**-s | --suite names SUITENAMES**

Optional

Comma-separated list of Apex test suite names to run. You can only specify one of classnames, suite names, or tests.

Type: array

**-t | --tests TESTS**

Optional

Comma-separated list of Apex test class names or IDs and test methods, if applicable, to run. You can only specify one of classnames, suite names, or tests.

Type: array

**-c | --codecoverage**

Optional

Retrieves code coverage results.

Type: boolean

**-d | --outputdir OUTPUTDIR**

Optional

Directory to store test run files.

Type: directory

**-l | --testlevel TESTLEVEL**

Optional

Specifies which tests to run, using one of these TestLevel enum values:

RunSpecifiedTests—Only the tests that you specify are run.

RunLocalTests—All tests in your org are run, except the ones that originate from installed managed packages.

RunAllTestsInOrg—All tests are in your org and in installed managed packages are run.

Type: enum

Permissible values are: RunLocalTests, RunAllTestsInOrg, RunSpecifiedTests

**-w | --wait WAIT**

Optional

Sets the streaming client socket timeout, in minutes. If the streaming client socket has no contact from the server for a number of minutes, the client exits. Specify a longer wait time if timeouts occur frequently.

Type: minutes

**-y | --synchronous**

Optional

Runs test methods from a single Apex class synchronously. If you don't specify this flag, tests are run asynchronously.

Type: boolean

**--verbose**

Optional

Displays Apex test processing details. If JSON format is specified, processing details aren't displayed.

Type: boolean

## **force:apex:trigger:create**

Creates an Apex trigger in the specified directory or the current working directory. If you don't explicitly set the API version, it defaults to the current API version. The .trigger file and associated metadata file are created.

### Help for **force:apex:trigger:create**

If not supplied, the apiversion, template, and outputdir use default values.

The outputdir can be an absolute path or relative to the current working directory.

Examples:

```
$ sfdx force:apex:trigger:create -n MyTrigger
```

```
$ sfdx force:apex:trigger:create -n MyTrigger -s Account -e 'before insert,after insert'
```

```
$ sfdx force:apex:trigger:create -n MyTrigger -d triggers
```

## Command Syntax

**sfdx force:apex:trigger:create**

[--json]

[--loglevel LOGLEVEL]

-n TRIGGERNAME

[-t TEMPLATE]

[-d OUTPUTDIR]

[--apiversion APIVERSION]

`[-s SUBJECT]`

`[-e TRIGGEREVENTS]`

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-n | --triggername TRIGGERNAME**

Required

The name of the new Apex trigger. The name can be up to 40 characters and must start with a letter.

Type: string

### **-t | --template TEMPLATE**

Optional

The template to use to create the file. Supplied parameter values or default values are filled into a copy of the template.

Type: string

Permissible values are: ApexTrigger

Default value: ApexTrigger

### **-d | --outputdir OUTPUTDIR**

Optional

The directory to store the newly created files. The location can be an absolute path or relative to the current working directory. The default is the current directory.

Type: string

Default value: .

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-s | --subject SUBJECT**

Optional

The sObject to create an Apex trigger on.

Type: string

Default value: SUBJECT

**-e | --triggerevents TRIGGEREVENTS**

Optional

The events that cause the trigger to fire.

Type: array

Default value: before insert

## auth Commands

Use the auth commands to authorize a Salesforce org for use with the Salesforce CLI.

[force:auth:device:login](#)

Authorize an org using a device code. You must open a browser, navigate to the verification URL, and enter the code. Log in, if not already logged in, and you'll be prompted to allow the device to connect to the org.

[force:auth:jwt:grant](#)

Authorizes a Salesforce org using the JWT flow.

[force:auth:list](#)

List auth connection information.

[force:auth:logout](#)

Logs you out from one or all of your authorized Salesforce orgs.

[force:auth:sfdxurl:store](#)

Authorizes a Salesforce org using an SFDX auth URL.

[force:auth:web:login](#)

Authorizes a Salesforce org by opening a browser so you can log in through salesforce.com.

### **force:auth:device:login**

Authorize an org using a device code. You must open a browser, navigate to the verification URL, and enter the code. Log in, if not already logged in, and you'll be prompted to allow the device to connect to the org.

### Examples for **force:auth:device:login**

```
sfdx force:auth:device:login -d -a TestOrg1
```

```
sfdx force:auth:device:login -i <OAuth client id>
```

```
sfdx force:auth:device:login -r https://test.salesforce.com
```

### Command Syntax

**sfdx force:auth:device:login**

[--json]

[--loglevel LOGLEVEL]

[-i CLIENTID]

[-r INSTANCEURL]

[-d]

[-s]

[-a SETALIAS]

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-i | --clientid CLIENTID**

Optional

OAuth client ID (aka consumer key)

Type: string

### **-r | --instanceurl INSTANCEURL**

Optional

login URL of the instance the org is on

Type: url

### **-d | --setdefaultdevhubusername**

Optional

set the authenticated org as the default Dev Hub org for scratch org creation

Type: boolean

### **-s | --setdefaultusername**

Optional

set the authenticated org as the default username that all commands run against

Type: boolean

### **-a | --setalias SETALIAS**

Optional

set an alias for the authenticated org

Type: string

## **force:auth:jwt:grant**

Authorizes a Salesforce org using the JWT flow.

## Help for `force:auth:jwt:grant`

Authorizes a Salesforce org using a certificate associated with your private key that has been uploaded to a personal connected app.

If you specify an `--instanceurl` value, this value overrides the `sfdcLoginUrl` value in your `sfdx-project.json` file. To specify a My Domain URL, use the format `MyDomainName.my.salesforce.com` (not `MyDomainName.lightning.force.com`).

Examples:

```
$ sfdx force:auth:jwt:grant -u me@my.org -f <path to jwt key file> -i <OAuth client id>
```

```
$ sfdx force:auth:jwt:grant -u me@my.org -f <path to jwt key file> -i <OAuth client id>
-s -a MyDefaultOrg
```

```
$ sfdx force:auth:jwt:grant -u me@acme.org -f <path to jwt key file> -i <OAuth client id>
-r https://acme.my.salesforce.com
```

## Command Syntax

### `sfdx force:auth:jwt:grant`

```
[--json]
[--loglevel LOGLEVEL]
-u USERNAME
-f JWTKEYFILE
-i CLIENTID
[-r INSTANCEURL]
[-d]
[-s]
[-a SETALIAS]
```

## Parameters

### `--json`

Optional

Format output as JSON.

Type: boolean

### `--loglevel LOGLEVEL`

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### `-u` | `--username USERNAME`

Required

The authentication username.

Type: string

**-f | --jwtkeyfile JWTKEYFILE**

Required

Path to a file containing the private key.

Type: filepath

**-i | --clientid CLIENTID**

Required

The OAuth client ID (sometimes referred to as the consumer key).

Type: string

**-r | --instanceurl INSTANCEURL**

Optional

The login URL of the Salesforce instance that the org lives on.

Type: url

**-d | --setdefaultdevhubusername**

Optional

Sets the authenticated org as the default Dev Hub org for scratch org creation.

Type: boolean

**-s | --setdefaultusername**

Optional

Sets the authenticated org as the default username that all commands run against.

Type: boolean

**-a | --setalias SETALIAS**

Optional

Sets an alias for the authenticated org.

Type: string

## **force:auth:list**

List auth connection information.

## Command Syntax

**sfdx force:auth:list**

[--json]

[--loglevel LOGLEVEL]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean



**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**force:auth:logout**

Logs you out from one or all of your authorized Salesforce orgs.

**Help for force:auth:logout**

By default, this command logs you out from your default scratch org.

Examples:

```
$ sfdx force:auth:logout -u me@my.org
```

```
$ sfdx force:auth:logout -a
```

```
$ sfdx force:auth:logout -p
```

**Command Syntax****sfdx force:auth:logout**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-a]

[-p]

**Parameters****--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-a | --all**

Optional

Includes all authenticated orgs: for example, Dev Hubs, sandboxes, DE orgs, and expired, deleted, and unknown-status scratch orgs.

Type: boolean

**-p | --noprompt**

Optional

Do not prompt for confirmation.

Type: boolean

## **force:auth:sfdxurl:store**

Authorizes a Salesforce org using an SFDX auth URL.

### Help for **force:auth:sfdxurl:store**

Authorize a Salesforce org using an SFDX auth URL stored within a file.

The file must have the format "force://<refreshToken>@<instanceUrl>" or "force://<clientId>:<clientSecret>:<refreshToken>@<instanceUrl>".

The file must contain only the URL or be a JSON file that has a top-level property named sfdxAuthUrl.

Use this command to get the SFDX auth URL for a Dev Hub org you have already authorized:

```
$ sfdx force:org:display -u <DevHub> --verbose
```

Examples:

```
$ sfdx force:auth:sfdxurl:store -f <path to sfdxAuthUrl file>
```

```
$ sfdx force:auth:sfdxurl:store -f <path to sfdxAuthUrl file> -s -a MyDefaultOrg
```

## Command Syntax

**sfdx force:auth:sfdxurl:store**

[--json]

[--loglevel LOGLEVEL]

-f SFDXURLFILE

[-d]

[-s]

[-a SETALIAS]

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-f | --sfdxurlfile SFDXURLFILE**

Required

Path to a file containing the SFDX URL.

Type: filepath

### **-d | --setdefaultdevhubusername**

Optional

Sets the authenticated org as the default Dev Hub org for scratch org creation.

Type: boolean

### **-s | --setdefaultusername**

Optional

Sets the authenticated org as the default username that all commands run against.

Type: boolean

### **-a | --setalias SETALIAS**

Optional

Sets an alias for the authenticated org.

Type: string

## **force:auth:web:login**

Authorizes a Salesforce org by opening a browser so you can log in through salesforce.com.

### Help for **force:auth:web:login**

To log in to a sandbox, set --instanceurl to https://test.salesforce.com.

Examples:

```
$ sfdx force:auth:web:login -a TestOrg1
```

```
$ sfdx force:auth:web:login -i <OAuth client id>
```

```
$ sfdx force:auth:web:login -r https://test.salesforce.com
```

## Command Syntax

### **sfdx force:auth:web:login**

[--json]

[--loglevel LOGLEVEL]

[-i CLIENTID]

[-r INSTANCEURL]

[-d]

[-s]

[-a SETALIAS]

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-i | --clientid CLIENTID**

Optional

The OAuth client ID (sometimes referred to as the consumer key).

Type: string

### **-r | --instanceurl INSTANCEURL**

Optional

The login URL of the Salesforce instance that the org lives on.

Type: url

### **-d | --setdefaultdevhubusername**

Optional

Sets the authenticated org as the default Dev Hub org for scratch org creation.

Type: boolean

**-s | --setdefaultusername**

Optional

Sets the authenticated org as the default username that all commands run against.

Type: boolean

**-a | --setalias SETALIAS**

Optional

Sets an alias for the authenticated org.

Type: string

## cmdt Commands

Create and update custom metadata types and their records.

[force:cmdt:create](#)

Creates a new custom metadata type in the current project.

[force:cmdt:field:create](#)

Generate a custom metadata field based on the field type provided.

[force:cmdt:generate](#)

Generates a custom metadata type and all its records for the provided sObject.

[force:cmdt:record:create](#)

Create a new record for a given custom metadata type in the current project.

[force:cmdt:record:insert](#)

Create new custom metadata type records from a CSV file.

### force:cmdt:create

Creates a new custom metadata type in the current project.

### Examples for force:cmdt:create

Create a custom metadata type with developer name 'MyCustomType'; this name will also be used as the label:

```
$ sfdx force:cmdt:create --typename MyCustomType
```

Create a protected custom metadata type with a specific label:

```
$ sfdx force:cmdt:create --typename MyCustomType --label "Custom Type" --plurallabel "Custom Types" --visibility Protected
```

## Command Syntax

**sfdx force:cmdt:create**

[--json]

[--loglevel LOGLEVEL]

-n TYPENAME  
 [-l LABEL]  
 [-p PLURALLABEL]  
 [-v VISIBILITY]  
 [-d OUTPUTDIR]

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-n | --typename TYPENAME**

Required

The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your org. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores.

Type: string

### **-l | --label LABEL**

Optional

A label for the custom metadata type.

Type: string

### **-p | --plurallabel PLURALLABEL**

Optional

The plural version of the label value. If this flag is missing or blank, the singular label is used as the plural label.

Type: string

### **-v | --visibility VISIBILITY**

Optional

The visibility of the custom metadata type.

Type: enum

Permissible values are: PackageProtected, Protected, Public

Default value: Public

### **-d | --outputdir OUTPUTDIR**

Optional

The directory to store the newly-created custom metadata type files. The location can be an absolute path or relative to the current working directory. The default is the current directory.

Type: directory

## **force:cmdt:field:create**

Generate a custom metadata field based on the field type provided.

### Examples for **force:cmdt:field:create**

Create a metadata file for a custom checkbox field:

```
$ sfdx force:cmdt:field:create --fieldname MyField --fieldtype Checkbox
```

Create a metadata file for a custom picklist field:

```
$ sfdx force:cmdt:field:create --fieldname MyField --fieldtype Picklist --picklistvalues "A,B,C"
```

Create a metadata file for a custom number field:

```
$ sfdx force:cmdt:field:create --fieldname MyField --fieldtype Number --decimalplaces 2
```

## Command Syntax

### **sfdx force:cmdt:field:create**

```
[--json]
[--loglevel LOGLEVEL]
-n FIELDNAME
-f FIELDTYPE
[-p PICKLISTVALUES]
[-s DECIMALPLACES]
[-l LABEL]
[-d OUTPUTDIR]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-n | --fieldname FIELDNAME**

Required

The unique name for the field.

Type: string

**-f | --fieldtype FIELDTYPE**

Required

The unique name for the field.

Type: enum

Permissible values are: Checkbox, Date, DateTime, Email, Number, Percent, Phone, Picklist, Text, TextArea, LongTextArea, Url

**-p | --picklistvalues PICKLISTVALUES**

Optional

A comma-separated list of picklist values. These values are required when creating a Picklist field.

Type: array

**-s | --decimalplaces DECIMALPLACES**

Optional

The number of decimal places to use for Number or Percent fields. The value must be greater than or equal to zero.

Type: number

**-l | --label LABEL**

Optional

The label for the field.

Type: string

**-d | --outputdir OUTPUTDIR**

Optional

The directory to store the newly-created field definition files. The location can be an absolute path or relative to the current working directory. The default is the current directory.

Type: directory

## **force:cmdt:generate**

Generates a custom metadata type and all its records for the provided sObject.

### Examples for **force:cmdt:generate**

Generate a custom metadata type from an sObject in the default target org:

```
$ sfdx force:cmdt:generate --devname MyCMDT --subjectname MySourceObject__c
```

Generate a custom metadata type from an sObject in the specified target org; ignore unsupported field types instead of converting them to text:

```
$ sfdx force:cmdt:generate --devname MyCMDT --subjectname MySourceObject__c
--ignoreunsupported --targetusername 'alias or user email of the org containing the source
type'
```



Generate a protected custom metadata type from an sObject in the default target org:

```
$ sfdx force:cmt:generate --devname MyCMT --subjectname SourceCustomObject__c
--visibility Protected
```

Generate a protected custom metadata type with a specific label from an sObject in the default target org:

```
$ sfdx force:cmt:generate --devname MyCMT --label "My CMT" --plurallabel "My CMTs"
--subjectname SourceCustomSetting__c --visibility Protected
```

Generate a custom metadata type from an sObject in the default target org; put the resulting type metadata file in the specified directory:

```
$ sfdx force:cmt:generate --devname MyCMT --subjectname SourceCustomSetting__c
--typeoutputdir 'path/to/my/cmt/directory'
```

Generate a custom metadata type from an sObject in the default target org; put the resulting record metadata file(s) in the specified directory:

```
$ sfdx force:cmt:generate --devname MyCMT --subjectname SourceCustomSetting__c
--recordsoutputdir 'path/to/my/cmt/record/directory'
```

## Command Syntax

### **sfdx force:cmt:generate**

```
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-n DEVNAME
[-l LABEL]
[-p PLURALLABEL]
[-v VISIBILITY]
-s SUBJECTNAME
[-i]
[-d TYPEOUTPUTDIR]
[-r RECORDSOUTPUTDIR]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-n | --devname DEVNAME**

Required

The name of the custom metadata type.

Type: string

**-l | --label LABEL**

Optional

The label for the custom metadata type.

Type: string

**-p | --plurallabel PLURALLABEL**

Optional

The plural version of the label value. If this flag is missing or blank, the singular label is used as the plural label.

Type: string

**-v | --visibility VISIBILITY**

Optional

The visibility of the custom metadata type.

Type: enum

Permissible values are: PackageProtected, Protected, Public

Default value: Public

**-s | --subjectname SUBJECTNAME**

Required

The API name of the sObject source for custom metadata generation.

Type: string

**-i | --ignoreunsupported**

Optional

Ignore unsupported field types (these fields will not be created). The default is to create Text fields and convert the source value to text.

Type: boolean

**-d | --typeoutputdir TYPEOUTPUTDIR**

Optional

The directory to store newly-created custom metadata type files.

Type: directory

Default value: force-app/main/default/objects/

**-r | --recordsoutputdir RECORDSOUTPUTDIR**

Optional

The directory to store newly-created custom metadata record files.

Type: directory

Default value: force-app/main/default/customMetadata/

## **force:cmdt:record:create**

Create a new record for a given custom metadata type in the current project.

### Examples for **force:cmdt:record:create**

Create a record metadata file for custom metadata type 'MyCMT' with values specified for two custom fields:

```
$ sfdx force:cmdt:record:create --typename MyCMT__mdt --recordname MyRecord
My_Custom_Field_1=Foo My_Custom_Field_2=Bar
```

Create a protected record metadata file for custom metadata type 'MyCMT' with a specific label and values specified for two custom fields:

```
$ sfdx force:cmdt:record:create --typename MyCMT__mdt --recordname MyRecord --label
"My Record" --protected true My_Custom_Field_1=Foo My_Custom_Field_2=Bar
```

## Command Syntax

### **sfdx force:cmdt:record:create**

```
[--json]
[--loglevel LOGLEVEL]
-t TYPENAME
-n RECORDNAME
[-l LABEL]
[-p PROTECTED]
[-i INPUTDIR]
[-d OUTPUTDIR]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-t | --typename TYPENAME**

Required

The API name of the custom metadata type to create a record for.

Type: string

**-n | --recordname RECORDNAME**

Required

The name for the new record.

Type: string

**-l | --label LABEL**

Optional

The label for the new record.

Type: string

**-p | --protected PROTECTED**

Optional

Protect the record when it is in a managed package. Protected records can only be accessed by code in the same managed package namespace.

Type: string

Permissible values are: true, false

Default value: false

**-i | --inputdir INPUTDIR**

Optional

The directory to pull the custom metadata type definition from.

Type: directory

Default value: force-app/main/default/objects

**-d | --outputdir OUTPUTDIR**

Optional

The directory to store newly-created custom metadata record files.

Type: directory

Default value: force-app/main/default/customMetadata

**force:cmdt:record:insert**

Create new custom metadata type records from a CSV file.

## Examples for `force:cmdt:record:insert`

Create record metadata files for type 'My\_CMDT\_Name' (from your local project) based on values in a CSV file, using 'Name' as the column that specifies the record name:

```
$ sfdx force:cmdt:record:insert --filepath path/to/my.csv --typename My_CMDT_Name
```

Create record metadata files for type 'My\_CMDT\_Name' (from the specified directory) based on values in a CSV file, using 'PrimaryKey' as the column that specifies the record name:

```
$ sfdx force:cmdt:record:insert --filepath path/to/my.csv --typename My_CMDT_Name
--inputdir "path/to/my/cmdt/directory" --namecolumn "PrimaryKey"
```

## Command Syntax

**sfdx force:cmdt:record:insert**

```
[--json]
[--loglevel LOGLEVEL]
-f FILEPATH
-t TYPENAME
[-i INPUTDIR]
[-d OUTPUTDIR]
[-n NAMECOLUMN]
```

## Parameters

### **--json**

Optional  
Format output as JSON.  
Type: boolean

### **--loglevel LOGLEVEL**

Optional  
The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.  
Type: enum  
Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL  
Default value: warn

### **-f | --filepath FILEPATH**

Required  
The path to the CSV file.  
Type: string

### **-t | --typename TYPENAME**

Required  
The API Name of the custom metadata type. The '\_\_\_mdt' suffix will be appended to the end of the name if it is omitted.  
Type: string

**-i | --inputdir INPUTDIR**

Optional

The directory to pull the custom metadata type definition from.

Type: directory

Default value: force-app/main/default/objects

**-d | --outputdir OUTPUTDIR**

Optional

The directory to store newly-created custom metadata record files.

Type: directory

Default value: force-app/main/default/customMetadata

**-n | --namecolumn NAMECOLUMN**

Optional

The column that is used to determine the name of the record.

Type: string

Default value: Name

## community Commands

Use the community commands to create and publish a community, and view a list of available templates in you org.

[force:community:create](#)

Creates a community using a template.

[force:community:publish](#)

Publishes your Lightning community to make it live. Each time you publish it, you update the live community with the most recent updates.

[force:community:template:list](#)

Retrieves the list of templates available in your org for creating a community.

### **force:community:create**

Creates a community using a template.

#### Help for **force:community:create**

See Which Community Template Should I Use? in Salesforce Help for more information about the different template types.

When you execute this command, it creates the community in preview status, which means that it isn't yet live. After you finish building your community, you can make it live.

If you have a Lightning community, publish the community using the sfdx force:community:publish command to make it live.

If you have a Salesforce Tabs + Visualforce community, activate the community to make it live by updating the status field of the Network type in the Metadata API. Alternatively, in Experience Workspaces, go to Administration | Settings, and click Activate.

For Lightning communities, activating the community just sends out a welcome email to community members.

Examples:

```
$ sfdx force:community:create --name "My Customer Community" --templatename "Customer Service" --urlpathprefix customers --description "My customer community"
```

```
$ sfdx force:community:create -n partnercommunity -t "Partner Central" -p partners
```

## Command Syntax

### **sfdx force:community:create**

```
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-n NAME
-t TEMPLATENAME
-p URLPATHPREFIX
[-d DESCRIPTION]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-n | --name NAME**

Required

The name of the community to create.

Type: string

**-t | --templatename** **TEMPLATENAME**

Required

The template to use to create the community, such as the Customer Service template. Run `force:community:template:list` to see which templates are available in your org.

Type: string

**-p | --urlpathprefix** **URLPATHPREFIX**

Required

The URL to append to the domain that you created when you enabled Communities for this org. For example, if your Communities domain name is `UniversalTelco.force.com` and you're creating a customer community, enter "customers" to create the unique URL

`UniversalTelco.force.com/customers`.

Type: string

**-d | --description** **DESCRIPTION**

Optional

The description of the community. The description displays in All Communities in Setup and helps with community identification.

Type: string

## **force:community:publish**

Publishes your Lightning community to make it live. Each time you publish it, you update the live community with the most recent updates.

### Help for **force:community:publish**

When you publish a Lightning community for the first time, you make the community's URL live and enable login access for community members.

Additionally, to send a welcome email to all community members, you must activate the Lightning community. (Activation is also required to successfully set up SEO for Lightning communities.) To activate a community, update the status field of the Network type in the Metadata API. Alternatively, in Experience Workspaces, go to Administration | Settings, and click Activate.

Subsequently, each time you publish the community, you update the live site with all changes made to the community since it was last published.

An email notification informs you when your changes are live.

Example:

```
$ sfdx force:community:publish --name "My Customer Community"
```

## Command Syntax

**sfdx force:community:publish**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]



`-n NAME`

## Parameters

### `--json`

Optional

Format output as JSON.

Type: boolean

### `--loglevel LOGLEVEL`

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### `-u | --targetusername TARGETUSERNAME`

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### `--apiversion APIVERSION`

Optional

Override the API version used for API requests made by this command.

Type: string

### `-n | --name NAME`

Required

The name of the Lightning community that you want to publish.

Type: string

## **`force:community:template:list`**

Retrieves the list of templates available in your org for creating a community.

### Help for **`force:community:template:list`**

See Which Community Template Should I Use? in Salesforce Help for more information about the different template types.

Example:

```
$ sfdx force:community:template:list
```

## Command Syntax

```
sfdx force:community:template:list
```

```
[--json]
```

```
[--loglevel LOGLEVEL]
```

`[-u TARGETUSERNAME]`

`[--apiversion APIVERSION]`

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

## data Commands

Use the data commands to manipulate records in your org. Commands are available to help you work with various APIs. Import CSV files with the Bulk API. Export and import data that includes master-detail relationships with the SObject Tree Save API. Perform simple CRUD operations on individual records with the REST API.

### [force:data:bulk:delete](#)

Deletes a batch of records listed in a CSV file.

### [force:data:bulk:status](#)

Polls the Bulk API for job status or batch status.

### [force:data:bulk:upsert](#)

Creates a job and one or more batches for inserting new rows and updating existing rows by accessing the Bulk API.

### [force:data:record:create](#)

Creates and inserts a record.

### [force:data:record:delete](#)

Deletes a single record.

[force:data:record:get](#)

Displays a single record.

[force:data:record:update](#)

Updates a single record.

[force:data:soql:query](#)

Executes a SOQL query.

[force:data:tree:export](#)

Exports data from an org into sObject tree format for `force:data:tree:import` consumption.

[force:data:tree:import](#)

Imports data into an org using the SObject Tree Save API. This data can include master-detail relationships.

## **force:data:bulk:delete**

Deletes a batch of records listed in a CSV file.

### Help for **force:data:bulk:delete**

The file must be a CSV file with only one column: "Id".

One job can contain many batches, depending on the length of the CSV file.

Returns a job ID and a batch ID. Use these IDs to check job status with `data:bulk:status`.

Examples:

```
$ sfdx force:data:bulk:delete -s Account -f ./path/to/file.csv
```

```
$ sfdx force:data:bulk:delete -s MyObject__c -f ./path/to/file.csv
```

## Command Syntax

**sfdx force:data:bulk:delete**

`[--json]`

`[--loglevel LOGLEVEL]`

`[-u TARGETUSERNAME]`

`[--apiversion APIVERSION]`

`-s SUBJECTTYPE`

`-f CSVFILE`

`[-w WAIT]`

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-s | --subjecttype SUBJECTTYPE**

Required

The sObject type of the records you're deleting.

Type: string

**-f | --csvfile CSVFILE**

Required

The path to the CSV file that contains the IDs of the records to delete.

Type: filepath

**-w | --wait WAIT**

Optional

The number of minutes to wait for the command to complete before displaying the results.

Type: minutes

**force:data:bulk:status**

Polls the Bulk API for job status or batch status.

**Help for force:data:bulk:status**

Examples:

```
$ sfdx force:data:bulk:status -i 750xx000000005sAAA
```

```
$ sfdx force:data:bulk:status -i 750xx000000005sAAA -b 751xx000000005nAAA
```

**Command Syntax****sfdx force:data:bulk:status**

[--json]

```
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-i JOBID
[-b BATCHID]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-i | --jobid JOBID**

Required

The ID of the job you want to view or of the job whose batch you want to view.

Type: id

### **-b | --batchid BATCHID**

Optional

The ID of the batch whose status you want to view.

Type: id

## **force:data:bulk:upsert**

Creates a job and one or more batches for inserting new rows and updating existing rows by accessing the Bulk API.

## Help for `force:data:bulk:upsert`

Inserts or updates records from a CSV file.

One job can contain many batches, depending on the length of the CSV file.

Returns a job ID and a batch ID. Use these IDs to check job status with `data:bulk:status`.

For information about formatting your CSV file, see "Prepare CSV Files" in the Bulk API Developer Guide.

Examples:

```
$ sfdx force:data:bulk:upsert -s MyObject__c -f ./path/to/file.csv -i MyField__c
```

```
$ sfdx force:data:bulk:upsert -s MyObject__c -f ./path/to/file.csv -i Id -w 2
```

## Command Syntax

### `sfdx force:data:bulk:upsert`

```
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-s SUBJECTTYPE
-f CSVFILE
-i EXTERNALID
[-w WAIT]
```

## Parameters

### `--json`

Optional

Format output as JSON.

Type: boolean

### `--loglevel LOGLEVEL`

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### `-u` | `--targetusername TARGETUSERNAME`

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### `--apiversion APIVERSION`

Optional

Override the API version used for API requests made by this command.

Type: string

**-s | --subjecttype SUBJECTTYPE**

Required

The sObject type of the records you want to upsert.

Type: string

**-f | --csvfile CSVFILE**

Required

The path to the CSV file that defines the records to upsert.

Type: filepath

**-i | --externalid EXTERNALID**

Required

The column name of the external ID.

Type: string

**-w | --wait WAIT**

Optional

The number of minutes to wait for the command to complete before displaying the results.

Type: minutes

## force:data:record:create

Creates and inserts a record.

### Help for force:data:record:create

The format of a field-value pair is <fieldName>=<value>.

Enclose all field-value pairs in one set of double quotation marks, delimited by spaces.

Enclose values that contain spaces in single quotes.

To get data on API performance metrics, specify both --perflog and --json.

Examples:

```
$ sfdx force:data:record:create -s Account -v "Name=Acme"
```

```
$ sfdx force:data:record:create -s Account -v "Name='Universal Containers'"
```

```
$ sfdx force:data:record:create -s Account -v "Name='Universal Containers'
Website=www.example.com"
```

```
$ sfdx force:data:record:create -t -s TraceFlag -v "DebugLevelId=7d1170000008U36AAE
StartDate=2017-12-01T00:26:04.000+0000 ExpirationDate=2017-12-01T00:56:04.000+0000
LogType=CLASS_TRACING TracedEntityId=01p17000000R6bLAAS"
```

```
$ sfdx force:data:record:create -s Account -v "Name=Acme" --perflog --json
```

## Command Syntax

**sfdx force:data:record:create**

```
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-s SUBJECTTYPE
-v VALUES
[-t]
[--perflog]
```

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-s | --subjecttype SUBJECTTYPE**

Required

The sObject type of the record you're creating.

Type: string

**-v | --values VALUES**

Required

The <fieldName>=<value> pairs you're creating.

Type: string



**-t | --usetoolingapi**

Optional

Create the record using Tooling API.

Type: boolean

**--perflog**

Optional

Gets data on API performance metrics from the server. The data is stored in \$HOME/.sfdx/apiPerformanceLog.json

Type: boolean

**force:data:record:delete**

Deletes a single record.

**Help for force:data:record:delete**

Specify an sObject type and either an ID or a list of &lt;fieldName&gt;=&lt;value&gt; pairs.

The format of a field-value pair is &lt;fieldName&gt;=&lt;value&gt;.

Enclose all field-value pairs in one set of double quotation marks, delimited by spaces.

Enclose values that contain spaces in single quotes.

To get data on API performance metrics, specify both --perflog and --json.

Examples:

```
$ sfdx force:data:record:delete -s Account -i 001D000000Kv3d1
```

```
$ sfdx force:data:record:delete -s Account -w "Name=Acme"
```

```
$ sfdx force:data:record:delete -s Account -w "Name='Universal Containers'"
```

```
$ sfdx force:data:record:delete -s Account -w "Name='Universal Containers' Phone='(123) 456-7890'"
```

```
$ sfdx force:data:record:delete -t -s TraceFlag -i 7tf170000009cU6AAI --perflog --json
```

**Command Syntax****sfdx force:data:record:delete**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

-s SUBJECTTYPE

[-i SUBJECTID]

[-w WHERE]

[-t]

[`--perflog`]

## Parameters

### `--json`

Optional

Format output as JSON.

Type: boolean

### `--loglevel LOGLEVEL`

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### `-u | --targetusername TARGETUSERNAME`

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### `--apiversion APIVERSION`

Optional

Override the API version used for API requests made by this command.

Type: string

### `-s | --subjecttype SUBJECTTYPE`

Required

The sObject type of the record you're deleting.

Type: string

### `-i | --subjectid SUBJECTID`

Optional

The ID of the record you're deleting.

Type: id

### `-w | --where WHERE`

Optional

A list of `<fieldName>=<value>` pairs to search for.

Type: string

### `-t | --usetoolingapi`

Optional

Delete the record using Tooling API.

Type: boolean

### `--perflog`

Optional

Gets data on API performance metrics from the server. The data is stored in `$HOME/.sfdx/apiPerformanceLog.json`

Type: boolean

## **force:data:record:get**

Displays a single record.

### Help for **force:data:record:get**

Specify an sObject type and either an ID or a list of <fieldName>=<value> pairs.

The format of a field-value pair is <fieldName>=<value>.

Enclose all field-value pairs in one set of double quotation marks, delimited by spaces.

Enclose values that contain spaces in single quotes.

To get data on API performance metrics, specify both `--perflog` and `--json`.

Examples:

```
$ sfdx force:data:record:get -s Account -i 001D000000Kv3d1
```

```
$ sfdx force:data:record:get -s Account -w "Name=Acme"
```

```
$ sfdx force:data:record:get -s Account -w "Name='Universal Containers'"
```

```
$ sfdx force:data:record:get -s Account -w "Name='Universal Containers' Phone='(123) 456-7890'"
```

```
$ sfdx force:data:record:get -t -s TraceFlag -i 7tf170000009cUBAAY --perflog --json
```

## Command Syntax

### **sfdx force:data:record:get**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

-s SUBJECTTYPE

[-i SUBJECTID]

[-w WHERE]

[-t]

[--perflog]

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-s | --subjecttype SUBJECTTYPE**

Required

The sObject type of the record you're retrieving.

Type: string

**-i | --subjectid SUBJECTID**

Optional

The ID of the record you're retrieving.

Type: id

**-w | --where WHERE**

Optional

A list of <fieldName>=<value> pairs to search for.

Type: string

**-t | --usetoolingapi**

Optional

Retrieve the record using Tooling API.

Type: boolean

**--perflog**

Optional

Gets data on API performance metrics from the server. The data is stored in \$HOME/.sfdx/apiPerformanceLog.json

Type: boolean

## **force:data:record:update**

Updates a single record.

## Help for `force:data:record:update`

The format of a field-value pair is `<fieldName>=<value>`.

Enclose all field-value pairs in one set of double quotation marks, delimited by spaces.

Enclose values that contain spaces in single quotes.

To get data on API performance metrics, specify both `--perflog` and `--json`.

Examples:

```
$ sfdx force:data:record:update -s Account -i 001D000000Kv3dl -v "Name=NewAcme"
```

```
$ sfdx force:data:record:update -s Account -w "Name='Old Acme'" -v "Name='New Acme'"
```

```
$ sfdx force:data:record:update -s Account -i 001D000000Kv3dl -v "Name='Acme III'
Website=www.example.com"
```

```
$ sfdx force:data:record:update -t -s TraceFlag -i 7tf170000009cUBAAY -v
"ExpirationDate=2017-12-01T00:58:04.000+0000"
```

```
$ sfdx force:data:record:update -s Account -i 001D000000Kv3dl -v "Name=NewAcme" --perflog
--json
```

## Command Syntax

### `sfdx force:data:record:update`

```
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-s SUBJECTTYPE
[-i SUBJECTID]
[-w WHERE]
-v VALUES
[-t]
[--perflog]
```

## Parameters

### `--json`

Optional

Format output as JSON.

Type: boolean

### `--loglevel LOGLEVEL`

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-s | --subjecttype SUBJECTTYPE**

Required

The sObject type of the record you're updating.

Type: string

**-i | --subjectid SUBJECTID**

Optional

The ID of the record you're updating.

Type: id

**-w | --where WHERE**

Optional

A list of <fieldName>=<value> pairs to search for.

Type: string

**-v | --values VALUES**

Required

The <fieldName>=<value> pairs you're updating.

Type: string

**-t | --usetoolingapi**

Optional

Update the record using Tooling API.

Type: boolean

**--perflog**

Optional

Gets data on API performance metrics from the server. The data is stored in \$HOME/.sfdx/apiPerformanceLog.json

Type: boolean

## **force:data:soql:query**

Executes a SOQL query.

## Help for `force:data:soql:query`

When you execute this command in a project, it executes the query against the data in your default scratch org.

To get data on API performance metrics, specify both `--perflog` and `--json`.

Examples:

```
$ sfdx force:data:soql:query -q "SELECT Id, Name, Account.Name FROM Contact"
```

```
$ sfdx force:data:soql:query -q "SELECT Id, Name FROM Account WHERE ShippingState IN ('CA', 'NY')"
```

```
$ sfdx force:data:soql:query -q "SELECT Name FROM ApexTrigger" -t
```

```
$ sfdx force:data:soql:query -q "SELECT Name FROM ApexTrigger" --perflog --json
```

## Command Syntax

### `sfdx force:data:soql:query`

```
[--json]
[--loglevel LOGLEVEL]
[-r RESULTFORMAT]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-q QUERY
[-t]
[--perflog]
```

## Parameters

### `--json`

Optional

Format output as JSON.

Type: boolean

### `--loglevel LOGLEVEL`

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### `-r` | `--resultformat RESULTFORMAT`

Optional

Format to use when displaying results. If you also specify the `--json` flag, `--json` overrides this parameter.

Type: enum

Permissible values are: human, csv, json

Default value: human

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-q | --query QUERY**

Required

SOQL query to execute.

Type: string

**-t | --usetoolingapi**

Optional

Execute the query using Tooling API.

Type: boolean

**--perflog**

Optional

Gets data on API performance metrics from the server. The data is stored in \$HOME/.sfdx/apiPerformanceLog.json

Type: boolean

## **force:data:tree:export**

Exports data from an org into sObject tree format for force:data:tree:import consumption.

### Help for **force:data:tree:export**

Generates JSON files for use with the force:data:tree:import command.

Examples:

```
$ sfdx force:data:tree:export -q "SELECT Id, Name, (SELECT Name, Address__c FROM Properties__r) FROM Broker__c"
```

```
$ sfdx force:data:tree:export -q <path to file containing soql query> -x export-demo -d /tmp/sfdx-out -p
```

For more information and examples, run "sfdx force:data:tree:import -h".

The query for export can return a maximum of 2,000 records. For more information, see the REST API Developer Guide: [https://developer.salesforce.com/docs/atlas.en-us.api\\_rest.meta/api\\_rest/resources\\_composite\\_subject\\_tree.htm](https://developer.salesforce.com/docs/atlas.en-us.api_rest.meta/api_rest/resources_composite_subject_tree.htm)

## Command Syntax

**sfdx force:data:tree:export**

[--json]



```

[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-q QUERY
[-p]
[-x PREFIX]
[-d OUTPUTDIR]

```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-q | --query QUERY**

Required

A SOQL query statement or the path of a file containing a SOQL query statement to retrieve the records to export.

Type: string

### **-p | --plan**

Optional

Generates multiple sObject tree files and a plan definition file for aggregated import.

Type: boolean

### **-x | --prefix PREFIX**

Optional

Prefix of generated files.

Type: string

**-d | --outputdir OUTPUTDIR**

Optional

Directory to store generated files.

Type: directory

## **force:data:tree:import**

Imports data into an org using the SOject Tree Save API. This data can include master-detail relationships.

### Help for **force:data:tree:import**

To generate JSON files for use with `force:data:tree:import`, run `"sfdx force:data:tree:export"`.

Examples:

To import records as individual files, first run the export commands:

```
$ sfdx force:data:tree:export -q "SELECT Id, Name FROM Account"
```

```
$ sfdx force:data:tree:export -q "SELECT Id, LastName, FirstName FROM Contact"
```

Then run the import command:

```
$ sfdx force:data:tree:import -f Contact.json,Account.json -u me@my.org
```

To import multiple data files as part of a plan, first run the export command with the `-p | --plan` flag:

```
$ sfdx force:data:tree:export -p -q "SELECT Id, Name, (SELECT Id, LastName, FirstName FROM Contacts) FROM Account"
```

Then run the import command, supplying a filepath value for the `-p | --plan` parameter:

```
$ sfdx force:data:tree:import -p Account-Contact-plan.json -u me@my.org
```

The SOject Tree API supports requests that contain up to 200 records. For more information, see the REST API Developer Guide: [https://developer.salesforce.com/docs/atlas.en-us.api\\_rest.meta/api\\_rest/resources\\_composite\\_subject\\_tree.htm](https://developer.salesforce.com/docs/atlas.en-us.api_rest.meta/api_rest/resources_composite_subject_tree.htm)

## Command Syntax

**sfdx force:data:tree:import**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-f SUBJECTTREEFILES]

[-p PLAN]

[--confighelp]

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-f | --subjecttreefiles SUBJECTTREEFILES**

Optional

Comma-delimited, ordered paths of JSON files containing a collection of record trees to insert. Either --subjecttreefiles or --plan is required.

Type: array

### **-p | --plan PLAN**

Optional

Path to plan to insert multiple data files that have master-detail relationships. Either --subjecttreefiles or --plan is required.

Type: filepath

### **--confighelp**

Optional

Displays the schema information for the configuration file. If you use this option, all other options, except --json, are ignored.

Type: boolean

## lightning Commands

Use the lightning commands to create Aura components and Lightning web components. As of API version 45.0, you can build Lightning components using two programming models: Lightning Web Components, and the original model, Aura Components. Lightning web components and Aura components can coexist and interoperate on a page.

### `force:lightning:app:create`

Creates a Lightning app bundle in the specified directory or the current working directory. The bundle consists of multiple files in a folder with the designated name.

### `force:lightning:component:create`

Creates a bundle for an Aura component or a Lightning web component in the specified directory or the current working directory. The bundle consists of multiple files in a folder with the designated name.

### `force:lightning:event:create`

Creates a Lightning event bundle in the specified directory or the current working directory. The bundle consists of multiple files in a folder with the designated name.

### `force:lightning:interface:create`

Creates a Lightning interface bundle in the specified directory or the current working directory. The bundle consists of multiple files in a folder with the designated name.

### `force:lightning:lint`

Runs a static analysis, or “lint,” tool on your Aura component code. The default rules include recommended best practices and Lightning Locker requirements. For details, including how to customize the rules for your org, see the Lightning Aura Components Developer Guide.

### `force:lightning:lwc:test:create`

Creates a `__tests__` directory in the specified directory. Creates a `yourComponentName.test.js` file with boilerplate code in the `__tests__` directory.

### `force:lightning:lwc:test:run`

Invokes Lightning Web Components Jest unit tests.

### `force:lightning:lwc:test:setup`

Installs Jest unit testing tools for Lightning Web Components. For more information, see the Lightning Web Components Dev Guide: <https://developer.salesforce.com/docs/component-library/documentation/lwc/lwc.testing>.

### `force:lightning:test:create`

Creates a Lightning test in the specified directory or the current working directory. The `.resource` file and associated metadata file are created.

## **`force:lightning:app:create`**

Creates a Lightning app bundle in the specified directory or the current working directory. The bundle consists of multiple files in a folder with the designated name.

### **Help for `force:lightning:app:create`**

If not supplied, the `apiversion`, `template`, and `outputdir` use default values.

The `outputdir` can be an absolute path or relative to the current working directory.

If you don't specify an `outputdir`, we create a subfolder in your current working directory with the name of your bundle. For example, if the current working directory is `force-app` and your Lightning bundle is called `myBundle`, we create `force-app/myBundle/` to store the files in the bundle.

Examples:

```
$ sfdx force:lightning:app:create -n myapp
```

```
$ sfdx force:lightning:app:create -n myapp -d aura
```

## Command Syntax

```
sfdx force:lightning:app:create  
  [--json]  
  [--loglevel LOGLEVEL]  
  -n APPNAME  
  [-t TEMPLATE]  
  [-d OUTPUTDIR]  
  [--apiversion APIVERSION]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-n | --appname APPNAME**

Required

The Lightning app name. The name can be up to 40 characters and must start with a letter.

Type: string

### **-t | --template TEMPLATE**

Optional

The template to use to create the file. Supplied parameter values or default values are filled into a copy of the template.

Type: string

Permissible values are: DefaultLightningApp

Default value: DefaultLightningApp

### **-d | --outputdir OUTPUTDIR**

Optional

The directory to store the newly created files. The location can be an absolute path or relative to the current working directory. The default is the current directory.

Type: string

Default value: .

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

## **force:lightning:component:create**

Creates a bundle for an Aura component or a Lightning web component in the specified directory or the current working directory. The bundle consists of multiple files in a folder with the designated name.

### Help for **force:lightning:component:create**

If not supplied, the apiversion, template, and outputdir use default values.

The outputdir can be an absolute path or relative to the current working directory.

If you don't specify an outputdir, we create a subfolder in your current working directory with the name of your bundle. For example, if the current working directory is force-app and your Lightning bundle is called myBundle, we create force-app/myBundle/ to store the files in the bundle.

To create a Lightning web component, pass --type lwc to the command. If you don't include a --type value, Salesforce CLI creates an Aura component by default.

Examples:

```
$ sfdx force:lightning:component:create -n mycomponent
```

```
$ sfdx force:lightning:component:create -n mycomponent --type lwc
```

```
$ sfdx force:lightning:component:create -n mycomponent -d aura
```

```
$ sfdx force:lightning:component:create -n mycomponent --type lwc -d lwc
```

## Command Syntax

### **sfdx force:lightning:component:create**

```
[--json]
[--loglevel LOGLEVEL]
-n COMPONENTNAME
[-t TEMPLATE]
[-d OUTPUTDIR]
[--apiversion APIVERSION]
[--type TYPE]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-n | --componentname COMPONENTNAME**

Required

The Lightning component name. The name can be up to 40 characters and must start with a letter.

Type: string

**-t | --template TEMPLATE**

Optional

The template to use to create the file. Supplied parameter values or default values are filled into a copy of the template.

Type: string

Default value: DefaultLightningCmp

**-d | --outputdir OUTPUTDIR**

Optional

The directory to store the newly created files. The location can be an absolute path or relative to the current working directory. The default is the current directory.

Type: string

Default value: .

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**--type TYPE**

Optional

The type of the new Lightning component.

Type: string

Permissible values are: aura, lwc

Default value: aura

## **force:lightning:event:create**

Creates a Lightning event bundle in the specified directory or the current working directory. The bundle consists of multiple files in a folder with the designated name.

### **Help for force:lightning:event:create**

If not supplied, the apiversion, template, and outputdir use default values.

The outputdir can be an absolute path or relative to the current working directory.

If you don't specify an `outputdir`, we create a subfolder in your current working directory with the name of your bundle. For example, if the current working directory is `force-app` and your Lightning bundle is called `myBundle`, we create `force-app/myBundle/` to store the files in the bundle.

Examples:

```
$ sfdx force:lightning:event:create -n myevent
```

```
$ sfdx force:lightning:event:create -n myevent -d aura
```

## Command Syntax

**sfdx force:lightning:event:create**

```
[--json]  
[--loglevel LOGLEVEL]  
-n EVENTNAME  
[-t TEMPLATE]  
[-d OUTPUTDIR]  
[--apiversion APIVERSION]
```

## Parameters

### **--json**

Optional  
Format output as JSON.  
Type: boolean

### **--loglevel LOGLEVEL**

Optional  
The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.  
Type: enum  
Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL  
Default value: warn

### **-n | --eventname EVENTNAME**

Required  
The Lightning event name. The name can be up to 40 characters and must start with a letter.  
Type: string

### **-t | --template TEMPLATE**

Optional  
The template to use to create the file. Supplied parameter values or default values are filled into a copy of the template.  
Type: string  
Permissible values are: DefaultLightningEvt  
Default value: DefaultLightningEvt



**-d | --outputdir OUTPUTDIR**

Optional

The directory to store the newly created files. The location can be an absolute path or relative to the current working directory. The default is the current directory.

Type: string

Default value: .

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**force:lightning:interface:create**

Creates a Lightning interface bundle in the specified directory or the current working directory. The bundle consists of multiple files in a folder with the designated name.

**Help for force:lightning:interface:create**

If not supplied, the apiversion, template, and outputdir use default values.

The outputdir can be an absolute path or relative to the current working directory.

If you don't specify an outputdir, we create a subfolder in your current working directory with the name of your bundle. For example, if the current working directory is force-app and your Lightning bundle is called myBundle, we create force-app/myBundle/ to store the files in the bundle.

Examples:

```
$ sfdx force:lightning:interface:create -n myinterface
```

```
$ sfdx force:lightning:interface:create -n myinterface -d aura
```

**Command Syntax****sfdx force:lightning:interface:create**

[--json]

[--loglevel LOGLEVEL]

-n INTERFACENAME

[-t TEMPLATE]

[-d OUTPUTDIR]

[--apiversion APIVERSION]

**Parameters****--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-n | --interfacename INTERFACENAME**

Required

The Lightning interface name. The name can be up to 40 characters and must start with a letter.

Type: string

**-t | --template TEMPLATE**

Optional

The template to use to create the file. Supplied parameter values or default values are filled into a copy of the template.

Type: string

Permissible values are: DefaultLightningIntf

Default value: DefaultLightningIntf

**-d | --outputdir OUTPUTDIR**

Optional

The directory to store the newly created files. The location can be an absolute path or relative to the current working directory. The default is the current directory.

Type: string

Default value: .

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

## **force:lightning:lint**

Runs a static analysis, or "lint," tool on your Aura component code. The default rules include recommended best practices and Lightning Locker requirements. For details, including how to customize the rules for your org, see the Lightning Aura Components Developer Guide.

### Examples for **force:lightning:lint**

```
<%= config.bin %> <%= command.id %> ./path/to/be/linted/
```

### Command Syntax

```
sfdx force:lightning:lint
```

```
[--json]
```

```
[--loglevel LOGLEVEL]
[-i IGNORE]
[--files FILES]
[--config CONFIG]
[--verbose]
[--exit]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-i | --ignore IGNORE**

Optional

A "glob" pattern used to filter folders (and their contents) out of the analysis. For example: `--ignore **/foo/**`.

Type: string

### **--files FILES**

Optional

A "glob" pattern used to add specific files to the analysis. For example, to only analyse your controller files, use `--files **/*Controller.js`. When specified, this value overrides the `--ignore` flag. The default is all .js files.

Type: string

### **--config CONFIG**

Optional

Path to a custom ESLint configuration file. Only code style rules are used, while the rest are ignored. For example: `--config path/to/.eslintrc`.

Type: string

### **--verbose**

Optional

Report both warnings and errors. The default is to report only errors.

Type: boolean

### **--exit**

Optional

Exit with error code 1 if there are lint issues. The default exits without an error code.

Type: boolean

### **force:lightning:lwc:test:create**

Creates a `__tests__` directory in the specified directory. Creates a `yourComponentName.test.js` file with boilerplate code in the `__tests__` directory.

### Examples for **force:lightning:lwc:test:create**

```
$ sfdx force:lightning:lwc:test:create -f force-app/main/default/lwc/myButton/myButton.js
```

### Command Syntax

```
sfdx force:lightning:lwc:test:create
```

```
  [--json]
```

```
  [--loglevel LOGLEVEL]
```

```
  -f FILEPATH
```

### Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-f | --filepath FILEPATH**

Required

Path to Lightning web component .js file to create a test for.

Type: string

### **force:lightning:lwc:test:run**

Invokes Lightning Web Components Jest unit tests.

### Examples for **force:lightning:lwc:test:run**

```
$ sfdx force:lightning:lwc:test:run
```

```
$ sfdx force:lightning:lwc:test:run -w
```

## Command Syntax

```
sfdx force:lightning:lwc:test:run  
  [--json]  
  [--loglevel LOGLEVEL]  
  [-d]  
  [--watch]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-d | --debug**

Optional

Runs tests in a Node process that an external debugger can connect to. The run pauses until the debugger is connected. For more information, see: <https://jestjs.io/docs/en/troubleshooting>

Type: boolean

### **--watch**

Optional

Runs tests when a watched file changes. Watched files include the component under test and any files it references.

Type: boolean

## **force:lightning:lwc:test:setup**

Installs Jest unit testing tools for Lightning Web Components. For more information, see the Lightning Web Components Dev Guide: <https://developer.salesforce.com/docs/component-library/documentation/lwc/lwc.testing>.

## Examples for **force:lightning:lwc:test:setup**

```
$ sfdx force:lightning:lwc:test:setup
```

## Command Syntax

```
sfdx force:lightning:lwc:test:setup  
  [--json]
```

```
[--loglevel LOGLEVEL]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

## **force:lightning:test:create**

Creates a Lightning test in the specified directory or the current working directory. The .resource file and associated metadata file are created.

## Help for **force:lightning:test:create**

If not supplied, the apiversion, template, and outputdir use default values.

The outputdir can be an absolute path or relative to the current working directory.

Examples:

```
$ sfdx force:lightning:test:create -n MyLightningTest
```

```
$ sfdx force:lightning:test:create -n MyLightningTest -d lightningTests
```

## Command Syntax

### **sfdx force:lightning:test:create**

```
[--json]
```

```
[--loglevel LOGLEVEL]
```

```
-n TESTNAME
```

```
[-t TEMPLATE]
```

```
[-d OUTPUTDIR]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-n | --testname TESTNAME**

Required

The name of the new Lightning test. The name can be up to 40 characters and must start with a letter.

Type: string

**-t | --template TEMPLATE**

Optional

The template to use to create the file. Supplied parameter values or default values are filled into a copy of the template.

Type: string

Permissible values are: DefaultLightningTest

Default value: DefaultLightningTest

**-d | --outputdir OUTPUTDIR**

Optional

The directory to store the newly created files. The location can be an absolute path or relative to the current working directory. The default is the current directory.

Type: string

Default value: .

## limits Commands

Use the limits commands to view your org's limits and how close you are to reaching them.

[force:limits:api:display](#)

Displays remaining and maximum calls and events for your org.

### **force:limits:api:display**

Displays remaining and maximum calls and events for your org.

### Help for **force:limits:api:display**

When you execute this command in a project, it provides limit information for your default scratch org.

Examples:

```
$ sfdx force:limits:api:display
```

```
$ sfdx force:limits:api:display -u me@my.org
```

## Command Syntax

```
sfdx force:limits:api:display  
  [--json]  
  [--loglevel LOGLEVEL]  
  [-u TARGETUSERNAME]  
  [--apiversion APIVERSION]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

## mdapi Commands

Use the mdapi commands to retrieve and deploy Metadata API–formatted files that represent components in an org, or to convert Metadata API–formatted metadata into the source format used in Salesforce DX projects.

### [force:mdapi:convert](#)

Converts metadata retrieved via Metadata API into the source format used in Salesforce DX projects.

### [force:mdapi:deploy](#)

Deploys file representations of components into an org by creating or updating the components they represent. You can deploy and retrieve up to 10,000 files or 400 MB (39 MB compressed) at one time. The default target username is the admin user for the default scratch org.

### [force:mdapi:deploy:cancel](#)

Cancels an asynchronous metadata deployment.



[force:mdapi:deploy:report](#)

Checks the current status of an asynchronous metadata deployment.

[force:mdapi:describemetadata](#)

Displays details about metadata types enabled for your org. Use this information to identify the syntax needed for a <name> element in package.xml. The most recent API version is the default, or you can specify an older version.

[force:mdapi:listmetadata](#)

Displays properties of metadata components of a specified type. This call is useful when you want to identify individual components in your manifest file or if you want a high-level view of particular components in your organization. For example, you could use this target to return a list of names of all Layout components in your org, then use this information in a retrieve operation that returns a subset of these components.

[force:mdapi:retrieve](#)

Uses Metadata API to retrieve a .zip of XML files that represent metadata from the targeted org. The default target username is the admin user for the default scratch org. You can retrieve and deploy up to 10,000 files or 400 MB (39 MB compressed) at one time.

[force:mdapi:retrieve:report](#)

Check the status of an asynchronous metadata retrieval.

## **force:mdapi:convert**

Converts metadata retrieved via Metadata API into the source format used in Salesforce DX projects.

### Help for **force:mdapi:convert**

To use Salesforce CLI to work with components that you retrieved via Metadata API, first convert your files from the metadata format to the source format using "sfdx force:mdapi:convert".

To convert files from the source format back to the metadata format, so that you can deploy them using "sfdx force:mdapi:deploy", run "sfdx force:source:convert".

Examples:

```
$ sfdx force:mdapi:convert -r path/to/metadata
```

```
$ sfdx force:mdapi:convert -r path/to/metadata -d path/to/outputdir
```

## Command Syntax

### **sfdx force:mdapi:convert**

[--json]

[--loglevel LOGLEVEL]

-r ROOTDIR

[-d OUTPUTDIR]

[-x MANIFEST]

[-m METADATA]

[-p METADATAPATH]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-r | --rootdir ROOTDIR**

Required

The root directory that contains the metadata you retrieved using Metadata API.

Type: directory

**-d | --outputdir OUTPUTDIR**

Optional

The directory to store your files in after they're converted to the source format. Can be an absolute or relative path.

Type: directory

**-x | --manifest MANIFEST**

Optional

The complete path to the manifest (package.xml) file that specifies the metadata types to convert.

If you specify this parameter, don't specify --metadata or --sourcepath.

Type: string

**-m | --metadata METADATA**

Optional

A comma-separated list of metadata component names to convert.

Type: array

**-p | --metadatapath METADATAPATH**

Optional

A comma-separated list of paths to the local source files to convert. The supplied paths can be to a single file (in which case the operation is applied to only one file) or to a folder (in which case the operation is applied to all metadata types in the directory and its sub-directories).

If you specify this parameter, don't specify --manifest or --metadata. If the comma-separated list you're supplying contains spaces, enclose the entire comma-separated list in one set of double quotes.

Type: array

**force:mdapi:deploy**

Deploys file representations of components into an org by creating or updating the components they represent. You can deploy and retrieve up to 10,000 files or 400 MB (39 MB compressed) at one time. The default target username is the admin user for the default scratch org.

**Help for force:mdapi:deploy**

Specify the location of the files to deploy as a .zip file or by the root of the directory tree containing the files. To check the status of a deployment, specify its job ID. To run quick deploy of a recently validated package, use `--validateddeployrequestid` with the validated ID.

To wait for the command to finish running no matter how long the deployment takes, set `--wait` to `-1`: `"sfdx force mdapi:deploy -w -1 ..."`.

**Command Syntax**

```
sfdx force:mdapi:deploy
  [--json]
  [--loglevel LOGLEVEL]
  [-u TARGETUSERNAME]
  [--apiversion APIVERSION]
  [-c]
  [-d DEPLOYDIR]
  [-w WAIT]
  [-l TESTLEVEL]
  [-r RUNTESTS]
  [-o]
  [-g]
  [-q VALIDATEDDEPLOYREQUESTID]
  [--verbose]
  [-f ZIPFILE]
  [-s]
```

**Parameters****--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-c | --checkonly**

Optional

Validates the deployed metadata and runs all Apex tests, but prevents the deployment from being saved to the org.

If you change a field type from Master-Detail to Lookup or vice versa, that change isn't supported when using the `--checkonly` parameter to test a deployment (validation). This kind of change isn't supported for test deployments to avoid the risk of data loss or corruption. If a change that isn't supported for test deployments is included in a deployment package, the test deployment fails and issues an error.

If your deployment package changes a field type from Master-Detail to Lookup or vice versa, you can still validate the changes prior to deploying to Production by performing a full deployment to another test Sandbox. A full deployment includes a validation of the changes as part of the deployment process.

Note: A Metadata API deployment that includes Master-Detail relationships deletes all detail records in the Recycle Bin in the following cases.

1. For a deployment with a new Master-Detail field, soft delete (send to the Recycle Bin) all detail records before proceeding to deploy the Master-Detail field, or the deployment fails. During the deployment, detail records are permanently deleted from the Recycle Bin and cannot be recovered.
2. For a deployment that converts a Lookup field relationship to a Master-Detail relationship, detail records must reference a master record or be soft-deleted (sent to the Recycle Bin) for the deployment to succeed. However, a successful deployment permanently deletes any detail records in the Recycle Bin.

Type: boolean

**-d | --deploydir DEPLOYDIR**

Optional

The root of the directory tree that contains the files to deploy. The root must contain a valid `package.xml` file describing the entities in the directory structure. Required to initiate a deployment if you don't use `--zipfile`. If you specify both `--zipfile` and `--deploydir`, a zip file of the contents of the `--deploydir` directory is written to the location specified by `--zipfile`.

Type: directory

**-w | --wait WAIT**

Optional

The number of minutes to wait for the command to complete. The default is `-1` (no limit). `0`

Type: minutes

**-l | --testlevel TESTLEVEL**

Optional

Specifies which level of deployment tests to run. Valid values are:

**NoTestRun**—No tests are run. This test level applies only to deployments to development environments, such as sandbox, Developer Edition, or trial orgs. This test level is the default for development environments.

**RunSpecifiedTests**—Runs only the tests that you specify in the `--runtests` option. Code coverage requirements differ from the default coverage requirements when using this test level. Executed tests must comprise a minimum of 75% code coverage for each class and trigger in the deployment package. This coverage is computed for each class and trigger individually and is different than the overall coverage percentage.

**RunLocalTests**—All tests in your org are run, except the ones that originate from installed managed packages. This test level is the default for production deployments that include Apex classes or triggers.

**RunAllTestsInOrg**—All tests in your org are run, including tests of managed packages.

If you don't specify a test level, the default behavior depends on the contents of your deployment package. For more information, see "Running Tests in a Deployment" in the Metadata API Developer Guide.

Type: enum

Permissible values are: `NoTestRun`, `RunSpecifiedTests`, `RunLocalTests`, `RunAllTestsInOrg`

**-r | --runtests RUNTESTS**

Optional

Lists the Apex classes containing the deployment tests to run. Use this parameter when you set `--testlevel` to `RunSpecifiedTests`.

Type: array

**-o | --ignoreerrors**

Optional

Ignores the deploy errors, and continues with the deploy operation. The default is false. Keep this parameter set to false when deploying to a production org. If set to true, components without errors are deployed, and components with errors are skipped.

Type: boolean

**-g | --ignorewarnings**

Optional

If a warning occurs and `ignoreWarnings` is set to true, the `success` field in `DeployMessage` is true. When `ignoreWarnings` is set to false, `success` is set to false, and the warning is treated like an error.

This field is available in API version 18.0 and later. Prior to version 18.0, there was no distinction between warnings and errors. All problems were treated as errors and prevented a successful deployment.

Type: boolean

**-q | --validateddeployrequestid VALIDATEDDEPLOYREQUESTID**

Optional

Specifies the ID of a package with recently validated components to run a Quick Deploy. Deploying a validation helps you shorten your deployment time because tests aren't rerun. If you have a recent successful validation, you can deploy the validated components without running tests. A validation doesn't save any components in the org. You use a validation only to check the success or failure messages that you would receive with an actual deployment. To validate your components, add the `-c | --checkonly` flag when you run `"sfdx force:mdapi:deploy"`. This flag sets the `checkOnly="true"` parameter for your deployment. Before deploying a recent validation, ensure that the following requirements are met:

1. The components have been validated successfully for the target environment within the last 10 days.
2. As part of the validation, Apex tests in the target org have passed.
3. Code coverage requirements are met.

- If all tests in the org or all local tests are run, overall code coverage is at least 75%, and Apex triggers have some coverage.
- If specific tests are run with the RunSpecifiedTests test level, each class and trigger that was deployed is covered by at least 75% individually.

Type: id

#### **--verbose**

Optional

Indicates that you want verbose output from the deploy operation.

Type: boolean

#### **-f | --zipfile ZIPFILE**

Optional

The path to the .zip file of metadata files to deploy. You must indicate this option or --deploydir. If you specify both --zipfile and --deploydir, a .zip file of the contents of the deploy directory is created at the path specified for the .zip file.

Type: filepath

#### **-s | --singlepackage**

Optional

Indicates that the specified .zip file points to a directory structure for a single package. By default, the CLI assumes the directory is structured for a set of packages.

Type: boolean

## **force:mdapi:deploy:cancel**

Cancels an asynchronous metadata deployment.

### Help for **force:mdapi:deploy:cancel**

Use this command to cancel a specified asynchronous metadata deployment. You can also specify a wait time (in minutes) to check for updates to the canceled deploy status.

Examples:

Deploy a directory of files to the org

```
$ sfdx force:mdapi:deploy -d <directory>
```

Now cancel this deployment and wait two minutes

```
$ sfdx force:mdapi:deploy:cancel -w 2
```

If you have multiple deployments in progress and want to cancel a specific one, specify the job ID

```
$ sfdx force:mdapi:deploy:cancel -i <jobid>
```

Check the status of the cancel job

```
$ sfdx force:mdapi:deploy:report
```

## Command Syntax

```
sfdx force:mdapi:deploy:cancel
```

```
[--json]
```

```

[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
[-w WAIT]
[-i JOBID]

```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-w | --wait WAIT**

Optional

Number of minutes to wait for the command to complete and display results to the terminal window. If the command continues to run after the wait period, the CLI returns control of the terminal window to you. The default is 33 minutes.

Type: minutes

Default value: 33 minutes

### **-i | --jobid JOBID**

Optional

The job ID (requestId) of the deployment you want to cancel. If not specified, the default value is the ID of the most recent metadata deployment you ran using Salesforce CLI.

Type: id

## **force:mdapi:deploy:report**

Checks the current status of an asynchronous metadata deployment.

## Help for `force:mdapi:deploy:report`

Specify the job ID for the deploy you want to check. You can also specify a wait time (minutes) to check for updates to the deploy status.

### Command Syntax

```
sfdx force:mdapi:deploy:report
  [--json]
  [--loglevel LOGLEVEL]
  [-u TARGETUSERNAME]
  [--apiversion APIVERSION]
  [-w WAIT]
  [-i JOBID]
  [--verbose]
```

### Parameters

#### **--json**

Optional

Format output as JSON.

Type: boolean

#### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

#### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

#### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

#### **-w | --wait WAIT**

Optional

The number of minutes to wait for the command to complete. The default is -1 (no limit). 0

Type: minutes

#### **-i | --jobid JOBID**

Optional



The job ID (asyncId) of the deployment you want to check. If not specified, the default value is the ID of the most recent metadata deployment you ran using Salesforce CLI. Use with `-w` to resume waiting.

Type: id

**--verbose**

Optional

Indicates that you want verbose output for deploy results.

Type: boolean

## **force:mdapi:describemetadadata**

Displays details about metadata types enabled for your org. Use this information to identify the syntax needed for a `<name>` element in `package.xml`. The most recent API version is the default, or you can specify an older version.

### Help for **force:mdapi:describemetadadata**

The default target username is the admin user for the default scratch org. The username must have the Modify All Data permission or the Modify Metadata permission (Beta). For more information about permissions, see Salesforce Help.

Examples:

```
$ sfdx force:mdapi:describemetadadata -a 43.0
```

```
$ sfdx force:mdapi:describemetadadata -u me@example.com
```

```
$ sfdx force:mdapi:describemetadadata -f /path/to/outputfilename.txt
```

```
$ sfdx force:mdapi:describemetadadata -u me@example.com -f /path/to/outputfilename.txt
```

## Command Syntax

**sfdx force:mdapi:describemetadadata**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[-a APIVERSION]

[-f RESULTFILE]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**-a | --apiversion APIVERSION**

Optional

The API version to use. The default is the latest API version (50.0).

Type: string

**-f | --resultfile RESULTFILE**

Optional

Filters all the known metadata from the result such that all that is left are the types not yet fully supported by the CLI.

Type: filepath

## **force:mdapi:listmetadata**

Displays properties of metadata components of a specified type. This call is useful when you want to identify individual components in your manifest file or if you want a high-level view of particular components in your organization. For example, you could use this target to return a list of names of all Layout components in your org, then use this information in a retrieve operation that returns a subset of these components.

### **Help for force:mdapi:listmetadata**

The default target username is the admin user for the default scratch org.

Examples:

```
$ sfdx force:mdapi:listmetadata -m CustomObject
```

```
$ sfdx force:mdapi:listmetadata -m CustomObject -a 43.0
```

```
$ sfdx force:mdapi:listmetadata -m CustomObject -u me@example.com
```

```
$ sfdx force:mdapi:listmetadata -m CustomObject -f /path/to/outputfilename.txt
```

```
$ sfdx force:mdapi:listmetadata -m Dashboard --folder foldername
```

```
$ sfdx force:mdapi:listmetadata -m Dashboard --folder foldername -a 43.0
```

```
$ sfdx force:mdapi:listmetadata -m Dashboard --folder foldername -u me@example.com
```

```
$ sfdx force:mdapi:listmetadata -m Dashboard --folder foldername -f
/path/to/outputfilename.txt
```

```
$ sfdx force:mdapi:listmetadata -m CustomObject -u me@example.com -f
/path/to/outputfilename.txt
```

## Command Syntax

```
sfdx force:mdapi:listmetadata
  [--json]
  [--loglevel LOGLEVEL]
  [-u TARGETUSERNAME]
  [-a APIVERSION]
  [-f RESULTFILE]
  -m METADATATYPE
  [--folder FOLDER]
```

## Parameters

- json**  
Optional  
Format output as JSON.  
Type: boolean
- loglevel LOGLEVEL**  
Optional  
The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.  
Type: enum  
Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL  
Default value: warn
- u | --targetusername TARGETUSERNAME**  
Optional  
A username or alias for the target org. Overrides the default target org.  
Type: string
- a | --apiversion APIVERSION**  
Optional  
The API version to use. The default is the latest API version (50.0).  
Type: string
- f | --resultfile RESULTFILE**  
Optional  
The path to the file where the results of the command are stored. The default output destination is the console.  
Type: filepath
- m | --metadatatype METADATATYPE**  
Required  
The metadata type to be retrieved, such as CustomObject or Report. The metadata type value is case-sensitive.  
Type: string
- folder FOLDER**  
Optional

The folder associated with the component. This parameter is required for components that use folders, such as Dashboard, Document, EmailTemplate, or Report. The folder name value is case-sensitive.

Type: string

## **force:mdapi:retrieve**

Uses Metadata API to retrieve a .zip of XML files that represent metadata from the targeted org. The default target username is the admin user for the default scratch org. You can retrieve and deploy up to 10,000 files or 400 MB (39 MB compressed) at one time.

### Help for **force:mdapi:retrieve**

The default target username is the admin user for the default scratch org. You can retrieve and deploy up to 10,000 files or 400 MB (39 MB compressed) at one time.

### Command Syntax

#### **sfdx force:mdapi:retrieve**

```

[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[-a APIVERSION]
[-w WAIT]
-r RETRIEVETARGETDIR
[-k UNPACKAGED]
[--verbose]
[-d SOURCEDIR]
[-p PACKAGENAMES]
[-s]

```

### Parameters

#### **--json**

Optional

Format output as JSON.

Type: boolean

#### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

#### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**-a | --apiversion APIVERSION**

Optional

Use to override the default, which is the latest version supported by your CLI plug-in, with the version in your package.xml file.

Type: string

**-w | --wait WAIT**

Optional

The number of minutes to wait for the command to complete.

Type: minutes

**-r | --retrievetargetdir RETRIEVETARGETDIR**

Required

The root of the directory structure where the retrieved .zip or metadata files are put.

Type: directory

**-k | --unpacked UNPACKAGED**

Optional

The complete path for the manifest file that specifies the components to retrieve.

Type: filepath

**--verbose**

Optional

Indicates that you want verbose output from the retrieve operation.

Type: boolean

**-d | --sourcedir SOURCEDIR**

Optional

The source directory to use instead of the default package directory specified in sfdx-project.json

Type: directory

**-p | --packagenames PACKAGENAMES**

Optional

A comma-separated list of package names to retrieve.

Type: array

**-s | --singlepackage**

Optional

Indicates that the specified .zip file points to a directory structure for a single package. By default, the CLI assumes the directory is structured for a set of packages.

Type: boolean

## **force:mdapi:retrieve:report**

Check the status of an asynchronous metadata retrieval.

## Help for `force:mdapi:retrieve:report`

Specify the job ID and a target directory for the retrieve you want to check. You can also specify a wait time (minutes) to check for updates to the deploy status. If the retrieve was successful, the resulting zip file will be saved to the location passed in with the retrieve target parameter.

## Command Syntax

```
sfdx force:mdapi:retrieve:report
  [--json]
  [--loglevel LOGLEVEL]
  [-u TARGETUSERNAME]
  [--apiversion APIVERSION]
  [-w WAIT]
  [-r RETRIEVETARGETDIR]
  [--verbose]
  [-i JOBID]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-w | --wait WAIT**

Optional

The number of minutes to wait for the command to complete. -1

Type: minutes

**-r | --retrieve targetdir RETRIEVETARGETDIR**

Optional

The root of the directory structure where the retrieved .zip or metadata files are put.

Type: directory

**--verbose**

Optional

Indicates that you want verbose output from the retrieve operation.

Type: boolean

**-i | --jobid JOBID**

Optional

The job ID (asynclid) of the retrieve you want to check. If not specified, the default value is the ID of the most recent metadata retrieval you ran using Salesforce CLI. You must specify a `--retrieve targetdir`. Use with `--wait` to resume waiting.

Type: id

## org Commands

Use the org commands to manage the orgs you use with Salesforce CLI. Create and delete scratch orgs, list your created and authorized orgs, and open orgs in your browser.

[force:org:clone](#)

Clones a sandbox org using values specified in a configuration file or key=value pairs that you specify on the command line. Values specified on the command line override values in the configuration file.

[force:org:create](#)

Creates a scratch org or a sandbox org using the values specified in a configuration file or key=value pairs that you specify on the command line. Values specified on the command line override values in the configuration file.

[force:org:delete](#)

Marks a scratch or sandbox org for deletion.

[force:org:display](#)

Gets the description for the current or target org.

[force:org:list](#)

Lists all orgs that the Salesforce CLI has created or authenticated to.

[force:org:open](#)

Opens an org in your browser.

[force:org:shape:create \(Beta\)](#)

Create a scratch org configuration (shape) based on the specified source org.

[force:org:shape:delete \(Beta\)](#)

Deletes all org shapes that you've created for an org using the Salesforce CLI.

[force:org:shape:list \(Beta\)](#)

Lists all org shapes that you've created using the Salesforce CLI.

[force:org:snapshot:create \(Pilot\)](#)

Creates a snapshot of a scratch org.

[force:org:snapshot:delete \(Pilot\)](#)

Deletes a scratch org snapshot.

[force:org:snapshot:get \(Pilot\)](#)

Retrieves details about a scratch org snapshot.

[force:org:snapshot:list \(Pilot\)](#)

Lists scratch org snapshots for your Dev Hub.

[force:org:status](#)

Reports sandbox org creation status. If the org is ready, authenticates to the org.

## force:org:clone

Clones a sandbox org using values specified in a configuration file or key=value pairs that you specify on the command line. Values specified on the command line override values in the configuration file.

### Help for force:org:clone

Specify a configuration file with a sourceSandboxName or provide key=value pairs while cloning a sandbox. The --targetusername (-u) must be a production org with sandbox licenses. The --type (-t) is required if cloning a sandbox.

Examples:

```
$ sfdx force:org:clone -t sandbox -f config/dev-sandbox-def.json -u prodOrg -a MyDevSandbox
```

```
$ sfdx force:org:clone -t sandbox sandboxName=DevSbx1 sourceSandboxName=Sbx2Clone -u prodOrg -a MyDevSandbox
```

## Command Syntax

### sfdx force:org:clone

```
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-t TYPE
[-f DEFINITIONFILE]
[-s]
[-a SETALIAS]
[-w WAIT]
```

## Parameters

### --json

Optional

Format output as JSON.

Type: boolean



**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-t | --type TYPE**

Required

The type of org to create. Only "sandbox" is supported.

Type: enum

Permissible values are: sandbox

**-f | --definitionfile DEFINITIONFILE**

Optional

The path to an org definition file. Either --definitionfile or a vararg value for edition (for example, edition=Developer) is required. Varargs override the values in the scratch org definition file.

Type: filepath

**-s | --setdefaultusername**

Optional

Sets the created org as the default username.

Type: boolean

**-a | --setalias SETALIAS**

Optional

An alias for the created org.

Type: string

**-w | --wait WAIT**

Optional

Sets the streaming client socket timeout, in minutes. If the streaming client socket has no contact from the server for a number of minutes, the client exits. Specify a longer wait time if timeouts occur frequently.

Type: minutes

Default value: 6 minutes

**force:org:create**

Creates a scratch org or a sandbox org using the values specified in a configuration file or key=value pairs that you specify on the command line. Values specified on the command line override values in the configuration file.

**Help for force:org:create**

Specify a configuration file or provide key=value pairs while creating a scratch org or a sandbox. When creating scratch orgs, `—targetdevhubusername (-v)` must be a Dev Hub org. When creating sandboxes, the `--targetusername (-u)` must be a production org with sandbox licenses. The `—type (-t)` is required if creating a sandbox.

Examples:

```
$ sfdx force:org:create -f config/enterprise-scratch-def.json -a MyScratchOrg
```

```
$ sfdx force:org:create edition=Developer -a MyScratchOrg -s -v devHub
```

```
$ sfdx force:org:create -f config/enterprise-scratch-def.json -a ScratchOrgWithOverrides
username=testuser1@mycompany.org
```

```
$ sfdx force:org:create -t sandbox -f config/dev-sandbox-def.json -a MyDevSandbox -u
prodOrg
```

**Command Syntax****sfdx force:org:create**

```
[--json]
[--loglevel LOGLEVEL]
[-v TARGETDEVHUBUSERNAME]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
[-t TYPE]
[-f DEFINITIONFILE]
[-n]
[-c]
[-i CLIENTID]
[-s]
[-a SETALIAS]
[-w WAIT]
[-d DURATIONDAYS]
```

**Parameters****--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-t | --type TYPE**

Optional

The type of org to create.

Type: enum

Permissible values are: scratch, sandbox

Default value: scratch

**-f | --definitionfile DEFINITIONFILE**

Optional

The path to an org definition file. Either --definitionfile or a vararg value for edition (for example, edition=Developer) is required. Varargs override the values in the scratch org definition file.

Type: filepath

**-n | --nonamespace**

Optional

Creates the scratch org with no namespace. Scratch orgs without a namespace are useful when testing installations of packages with namespaces.

Type: boolean

**-c | --noancestors**

Optional

Prevents the inclusion of second-generation package ancestors in the scratch org.

Type: boolean

**-i | --clientid CLIENTID**

Optional

A connected app consumer key, as configured in your Dev Hub or production org. This parameter is not supported for sandbox org creation.

Type: string

**-s | --setdefaultusername**

Optional

Sets the created org as the default username.

Type: boolean

**-a | --setalias SETALIAS**

Optional

An alias for the created org.

Type: string

**-w | --wait WAIT**

Optional

Sets the streaming client socket timeout, in minutes. If the streaming client socket has no contact from the server for a number of minutes, the client exits. Specify a longer wait time if timeouts occur frequently.

Type: minutes

Default value: 6 minutes

**-d | --durationdays DURATIONDAYS**

Optional

Sets the duration of the scratch org, in days. Valid values are from 1-30. The default is 7 days.

Type: number

**force:org:delete**

Marks a scratch or sandbox org for deletion.

**Help for force:org:delete**

To mark the org for deletion without being prompted to confirm, specify `--noprompt`.

Examples:

```
$ sfdx force:org:delete -u me@my.org
```

```
$ sfdx force:org:delete -u MyOrgAlias -p
```

**Command Syntax**

**sfdx force:org:delete**

[--json]

[--loglevel LOGLEVEL]

[-v TARGETDEVHUBUSERNAME]

[-u TARGETUSERNAME]  
 [--apiversion APIVERSION]  
 [-p]

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-p | --noprompt**

Optional

No prompt to confirm deletion.

Type: boolean

## **force:org:display**

Gets the description for the current or target org.

### Help for **force:org:display**

Output includes your access token, client ID, connected status, org ID, instance URL, username, and alias, if applicable.

Use --verbose to include the SFDX auth URL. Including --verbose displays the sfdxAuthUrl property only if you authenticated to the org using force:auth:web:login (not force:auth:jwt:grant).

Examples:

```
$ sfdx force:org:display
```

```
$ sfdx force:org:display -u me@my.org
```

```
$ sfdx force:org:display -u TestOrg1 --json
```

```
$ sfdx force:org:display -u TestOrg1 --json > tmp/MyOrgDesc.json
```

## Command Syntax

### **sfdx force:org:display**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[--verbose]

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **--verbose**

Optional

Emit additional command output to stdout.

Type: boolean

## **force:org:list**

Lists all orgs that the Salesforce CLI has created or authenticated to.

### Help for **force:org:list**

Examples:

```
$ sfdx force:org:list
```

```
$ sfdx force:org:list --verbose --json
```

```
$ sfdx force:org:list --verbose --json > tmp/MyOrgList.json
```

### Command Syntax

#### **sfdx force:org:list**

```
[--json]  
[--loglevel LOGLEVEL]  
[--verbose]  
[--all]  
[--clean]  
[-p]  
[--skipconnectionstatus]
```

### Parameters

#### **--json**

Optional

Format output as JSON.

Type: boolean

#### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

#### **--verbose**

Optional

Lists more information about each org.

Type: boolean

#### **--all**

Optional

Lists all authenticated orgs, including expired, deleted, and unknown-status scratch orgs.

Type: boolean

**--clean**

Optional

Remove all local org authorizations for deleted or expired orgs.

Type: boolean

**-p | --noprompt**

Optional

Do not prompt for confirmation.

Type: boolean

**--skipconnectionstatus**

Optional

Skips retrieving the connection status of non-scratch orgs

Type: boolean

## **force:org:open**

Opens an org in your browser.

### Help for **force:org:open**

Opens your default scratch org, or another org that you specify.

To open a specific page, specify the portion of the URL after "yourInstance.salesforce.com/" as --path.

For example, specify "--path lightning" to open Lightning Experience, or specify "--path /apex/YourPage" to open a Visualforce page.

To generate a URL but not launch your browser, specify --urlonly.

Examples:

```
$ sfdx force:org:open
```

```
$ sfdx force:org:open -u me@my.org
```

```
$ sfdx force:org:open -u MyTestOrg1
```

```
$ sfdx force:org:open -r -p lightning
```

## Command Syntax

**sfdx force:org:open**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-p PATH]

[-r]



## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-p | --path PATH**

Optional

Navigation URL path (not including domain).

Type: string

**-r | --urlonly**

Optional

Displays a navigation URL, but doesn't launch your browser.

Type: boolean

## **force:org:shape:create (Beta)**

Create a scratch org configuration (shape) based on the specified source org.



**Note:** This release contains a beta version of the `force:org:shape:create` command, which means it's a high-quality feature with known limitations. The `force:org:shape:create` command isn't generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. We can't guarantee general availability within any particular time frame or at all. Make your purchase decisions only on the basis of generally available products and features. You can provide feedback and suggestions for the `force:org:shape:create` command in the [Org Shape for Scratch Orgs](#) group in the Trailblazer Community.

## Help for `force:org:shape:create`

Examples:

```
$ sfdx force:org:shape:create -u me@my.org
```

```
$ sfdx force:org:shape:create -u me@my.org --json --loglevel debug
```

## Command Syntax

**sfdx force:org:shape:create**

`[--json]`

`[--loglevel LOGLEVEL]`

`[-u TARGETUSERNAME]`

`[--apiversion APIVERSION]`

`[-f DEFINITIONFILE]`

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-f | --definitionfile DEFINITIONFILE**


Optional

Creates a scratch definition file based on the specified source org. This name must end in `scratch-def.json`. If you specify a name that doesn't end in `scratch-def.json`, we'll append it. The file is generated in the config directory.

Type: string

## **force:org:shape:delete** (Beta)

Deletes all org shapes that you've created for an org using the Salesforce CLI.

 **Note:** This release contains a beta version of the `force:org:shape:delete` command, which means it's a high-quality feature with known limitations. The `force:org:shape:delete` command isn't generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. We can't guarantee general availability within any particular time frame or at all. Make your purchase decisions only on the basis of generally available products and features. You can provide feedback and suggestions for the `force:org:shape:delete` command in the [Org Shape for Scratch Orgs](#) group in the Trailblazer Community.

### Help for **force:org:shape:delete**

Examples:

```
$ sfdx force:org:shape:delete -u me@my.org
```

```
$ sfdx force:org:shape:delete -u MyOrgAlias -p
```

```
$ sfdx force:org:shape:delete -u me@my.org --json
```

```
$ sfdx force:org:shape:delete -u me@my.org -p --json > tmp/MyOrgShapeDelete.json
```

### Command Syntax

**sfdx force:org:shape:delete**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-p]

### Parameters

#### **--json**

Optional

Format output as JSON.

Type: boolean

#### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

#### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-p | --noprompt**


Optional

Do not prompt for confirmation.

Type: boolean

## **force:org:shape:list (Beta)**

Lists all org shapes that you've created using the Salesforce CLI.

 **Note:** This release contains a beta version of the `force:org:shape:list` command, which means it's a high-quality feature with known limitations. The `force:org:shape:list` command isn't generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. We can't guarantee general availability within any particular time frame or at all. Make your purchase decisions only on the basis of generally available products and features. You can provide feedback and suggestions for the `force:org:shape:list` command in the [Org Shape for Scratch Orgs](#) group in the Trailblazer Community.

### Help for **force:org:shape:list**

Examples:

```
$ sfdx force:org:shape:list
```

```
$ sfdx force:org:shape:list --json
```

```
$ sfdx force:org:shape:list --json > tmp/MyOrgShapeList.json
```

### Command Syntax

**sfdx force:org:shape:list**

[--json]

[--loglevel LOGLEVEL]

[--verbose]

### Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**--verbose**

Optional

Lists more information about each org shape.

Type: boolean

**force:org:snapshot:create (Pilot)**

Creates a snapshot of a scratch org.



**Note:** We provide the `force:org:snapshot:create` command to selected customers through an invitation-only pilot program that requires agreement to specific terms and conditions. Pilot programs are subject to change, and we can't guarantee acceptance. The `force:org:snapshot:create` command isn't generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. We can't guarantee general availability within any particular time frame or at all. Make your purchase decisions only on the basis of generally available products and features. You can provide feedback and suggestions for the `force:org:snapshot:create` command in the [W19 Pilot: Scratch Org Snapshots](#) group in the Trailblazer Community.

**Help for force:org:snapshot:create**

A snapshot is a point-in-time export of a scratch org. The export is stored in Salesforce and referenced by its unique name in a scratch definition file.

Use "sfdx force:org:snapshot:get" to get details, including status, about a snapshot creation request.

With "snapshot" in your scratch org definition file, use "sfdx force:org:create" to create a scratch org from a snapshot.

Examples:

```
$ sfdx force:org:snapshot:create --sourceorg 00Dxx00000000000 --snapshotname Dependencies
--description "Contains PackageA v1.1.0"
```

```
$ sfdx force:org:snapshot:create -o myuser@myorg -n NightlyBranch -d "Contains PkgA
v2.1.1.0 and PkgB 3.3.0"
```

**Command Syntax****sfdx force:org:snapshot:create**

[--json]

[--loglevel LOGLEVEL]

[-v TARGETDEVHUBUSERNAME]

[--apiversion APIVERSION]

-o SOURCEORG

`-n SNAPSHOTNAME`  
`[-d DESCRIPTION]`

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-o | --sourceorg SOURCEORG**

Required

The org ID, or a locally authenticated username or alias, of the scratch org to snapshot.

Type: string

### **-n | --snapshotname SNAPSHOTNAME**

Required

The unique name of the snapshot. Use this name to create scratch orgs from the snapshot.

Type: string

### **-d | --description DESCRIPTION**


Optional

A description of the snapshot. Use this description to document the contents of the snapshot. We suggest that you include a reference point, such as a version control system tag or commit ID.

Type: string

## **force:org:snapshot:delete (Pilot)**

Deletes a scratch org snapshot.

 **Note:** We provide the `force:org:snapshot:delete` command to selected customers through an invitation-only pilot program that requires agreement to specific terms and conditions. Pilot programs are subject to change, and we can't guarantee acceptance. The `force:org:snapshot:delete` command isn't generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. We can't guarantee general availability within any particular time frame or at all. Make your purchase decisions only on the basis of generally available products and features. You can provide feedback and suggestions for the `force:org:snapshot:delete` command in the [W19 Pilot: Scratch Org Snapshots](#) group in the Trailblazer Community.

## Help for `force:org:snapshot:delete`

Examples:

```
$ sfdx force:org:snapshot:delete --snapshot 00o...
```

```
$ sfdx force:org:snapshot:delete -s BaseSnapshot
```

## Command Syntax

**sfdx force:org:snapshot:delete**

```
[--json]
[--loglevel LOGLEVEL]
[-v TARGETDEVHUBUSERNAME]
[--apiversion APIVERSION]
-s SNAPSHOT
```

## Parameters

### **--json**

Optional  
Format output as JSON.  
Type: boolean

### **--loglevel LOGLEVEL**

Optional  
The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.  
Type: enum  
Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL  
Default value: warn

### **-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional  
A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.  
Type: string

### **--apiversion APIVERSION**

Optional  
Override the API version used for API requests made by this command.

Type: string

**-s | --snapshot SNAPSHOT**


Required

The name or ID (starts with 00o) of the snapshot to delete.

Type: string

## **force:org:snapshot:get (Pilot)**

Retrieves details about a scratch org snapshot.

 **Note:** We provide the `force:org:snapshot:get` command to selected customers through an invitation-only pilot program that requires agreement to specific terms and conditions. Pilot programs are subject to change, and we can't guarantee acceptance. The `force:org:snapshot:get` command isn't generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. We can't guarantee general availability within any particular time frame or at all. Make your purchase decisions only on the basis of generally available products and features. You can provide feedback and suggestions for the `force:org:snapshot:get` command in the [W19 Pilot: Scratch Org Snapshots](#) group in the Trailblazer Community.

## Help for **force:org:snapshot:get**

Use "sfdx force:org:snapshot:create" to create a snapshot.

Use "sfdx force:org:snapshot:list" to retrieve all snapshots.

Examples:

```
$ sfdx force:org:snapshot:get --snapshot 00o...
```

```
$ sfdx force:org:snapshot:get -s Dependencies
```

## Command Syntax

**sfdx force:org:snapshot:get**

[--json]

[--loglevel LOGLEVEL]

[-v TARGETDEVHUBUSERNAME]

[--apiversion APIVERSION]

-s SNAPSHOT

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.



Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-s | --snapshot SNAPSHOT**


Required

The name or ID (starts with 00o) of the snapshot to retrieve.

Type: string

## **force:org:snapshot:list (Pilot)**

Lists scratch org snapshots for your Dev Hub.

 **Note:** We provide the `force:org:snapshot:list` command to selected customers through an invitation-only pilot program that requires agreement to specific terms and conditions. Pilot programs are subject to change, and we can't guarantee acceptance. The `force:org:snapshot:list` command isn't generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. We can't guarantee general availability within any particular time frame or at all. Make your purchase decisions only on the basis of generally available products and features. You can provide feedback and suggestions for the `force:org:snapshot:list` command in the [W19 Pilot: Scratch Org Snapshots](#) group in the Trailblazer Community.

## **Help for force:org:snapshot:list**

Use "sfdx force:org:snapshot:get" to get details about a snapshot request.

Use "sfdx force:org:snapshot:create" to create a snapshot.

Examples:

```
$ sfdx force:org:snapshot:list
```

```
$ sfdx force:org:snapshot:list -v OtherDevHub@example.com
```

## **Command Syntax**

**sfdx force:org:snapshot:list**

[--json]

[--loglevel LOGLEVEL]

[-v TARGETDEVHUBUSERNAME]

[--apiversion APIVERSION]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

## **force:org:status**

Reports sandbox org creation status. If the org is ready, authenticates to the org.

### Help for **force:org:status**

Use this command to check the status of your sandbox org creation and, if the org is ready, authenticate to the org.

To specify the number of minutes that the command will wait for the sandbox org creation to complete before returning control of the terminal to you, include a `-w | --wait` value.

Examples:

```
$ sfdx force:org:status -n YourSandboxName -w 10
```

## Command Syntax

**sfdx force:org:status**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

-n SANDBOXNAME

[-s]

`[-a SETALIAS]`

`[-w WAIT]`

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-n | --sandboxname SANDBOXNAME**

Required

The name of the sandbox org whose creation status you want to check.

Type: string

### **-s | --setdefaultusername**

Optional

Sets the created org as the default username.

Type: boolean

### **-a | --setalias SETALIAS**

Optional

An alias for the created org.

Type: string

### **-w | --wait WAIT**

Optional

The number of minutes to wait for the org to be ready while polling for its creation status.

Type: minutes

Default value: 6 minutes

## package Commands

Use the package commands to develop and install packages.

### [force:package:create](#)

Creates a package.

### [force:package:delete](#)

Delete unlocked and second-generation managed packages. Before you delete a package, first delete all associated package versions.

### [force:package:hammertest:list \(Pilot\)](#)

Lists the statuses of running and completed ISV Hammer tests.

### [force:package:hammertest:report \(Pilot\)](#)

Returns the status of a running ISV Hammer test or the results of a completed hammer test.

### [force:package:hammertest:run \(Pilot\)](#)

Runs ISV Hammer for the specified package version and subscriber orgs.

### [force:package:install](#)

Installs a package in the target org.

### [force:package:install:report](#)

Retrieves the status of a package installation request.

### [force:package:installed:list](#)

Lists all packages installed in the target org.

### [force:package:list](#)

Lists all packages in the Dev Hub org.

### [force:package:uninstall](#)

Uninstalls a second-generation package from the target org. To uninstall a first-generation package, use the Salesforce user interface.

### [force:package:uninstall:report](#)

Retrieves the status of a package uninstall request.

### [force:package:update](#)

Updates details about a package. Does not create a package version.

### [force:package:version:create](#)

Creates a package version in the Dev Hub org.

### [force:package:version:create:list](#)

Lists all requests to create second-generation package versions in the Dev Hub org.

### [force:package:version:create:report](#)

Retrieves details about a package version creation request in the Dev Hub org.

### [force:package:version:delete](#)

Delete unlocked and second-generation managed package versions.

### [force:package:version:displayancestry \(Beta\)](#)

Displays the ancestry tree for a 2GP managed package version.

### [force:package:version:list](#)

Lists all package versions in the Dev Hub org.

[force:package:version:promote](#)

Promotes a package version to released status.

[force:package:version:report](#)

Retrieves details about a package version in the Dev Hub org.

[force:package:version:update](#)

Updates a second-generation package version in the Dev Hub org.

## **force:package:create**

Creates a package.

### Help for **force:package:create**

First, use this command to create a package. Then create a package version.

If you don't have a namespace defined in your sfdx-project.json file, use --nonamespace.

Your --name value must be unique within your namespace.

Examples:

```
$ sfdx force:package:create -n YourPackageName -t Unlocked -r force-app
```

```
$ sfdx force:package:create -n YourPackageName -d "Your Package Descripton" -t Unlocked
-r force-app
```

Run 'sfdx force:package:list' to list all packages in the Dev Hub org.

## Command Syntax

**sfdx force:package:create**

```
[--json]
[--loglevel LOGLEVEL]
[-v TARGETDEVHUBUSERNAME]
[--apiversion APIVERSION]
-n NAME
-t PACKAGETYPE
[-d DESCRIPTION]
[-e]
-r PATH
[--orgdependent]
[-o ERRORNOTIFICATIONUSERNAME]
```

## Parameters

**--json**  
Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-n | --name NAME**

Required

Name of the package to create.

Type: string

**-t | --packagetype PACKAGETYPE**

Required

Package type for the package.

The options for package type are Managed and Unlocked (Managed=DeveloperManagedSubscriberManaged, Unlocked=DeveloperControlledSubscriberEditable).

These options determine upgrade and editability rules.

Type: enum

Permissible values are: Managed, Unlocked

**-d | --description DESCRIPTION**

Optional

Description of the package.

Type: string

**-e | --nonamespace**

Optional

Creates the package with no namespace. Available only for unlocked packages. Useful when you're migrating an existing org to packages. But, use a namespaced package for new metadata.

Type: boolean

**-r | --path PATH**

Required

The path to the directory that contains the contents of the package.

Type: directory

**--orgdependent**

Optional

Package depends on unpackaged metadata in the installation org. Applies to unlocked packages only. (Beta)

Use Source Tracking in Sandboxes (Beta), to develop your org-dependent unlocked package.

Type: boolean

**-o | --errornotificationusername ERRORNOTIFICATIONUSERNAME**

Optional

An active Dev Hub org user designated to receive email notifications for unhandled Apex exceptions, and install, upgrade, or uninstall failures associated with your package.

Type: string

## force:package:delete

Delete unlocked and second-generation managed packages. Before you delete a package, first delete all associated package versions.

### Help for force:package:delete

Specify the ID or alias of the package you want to delete.

Examples:

```
$ sfdx force:package:delete -p "Your Package Alias"
```

```
$ sfdx force:package:delete -p 0Ho...
```

## Command Syntax

**sfdx force:package:delete**

[--json]

[--loglevel LOGLEVEL]

[-v TARGETDEVHUBUSERNAME]

[--apiversion APIVERSION]

[-n]

-p PACKAGE

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-n | --noprompt**

Optional

don't prompt before deleting the package

Type: boolean

**-p | --package PACKAGE**


Required

The ID (starts with 0Ho) or alias of the package to delete.

Type: string

## **force:package:hammertest:list (Pilot)**

Lists the statuses of running and completed ISV Hammer tests.

 **Note:** We provide the `force:package:hammertest:list` command to selected customers through a pilot program that requires agreement to specific terms and conditions. To be nominated to participate in the program, contact Salesforce. Pilot programs are subject to change, and we can't guarantee acceptance. The `force:package:hammertest:list` command isn't generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. We can't guarantee general availability within any particular time frame or at all. Make your purchase decisions only on the basis of generally available products and features. You can provide feedback and suggestions for the `force:package:hammertest:list` command in the [ISV Hammer - Pilot](#) group in the Trailblazer Community.

### **Help for force:package:hammertest:list**

If you supply a `--packageversionid` value, this command returns test statuses for only the tests associated with the specified package version. If you don't include the `--packageversionid` parameter, the command lists all test statuses.

Examples:

```
$ sfdx force:package:hammertest:list -u you@yourpackagingorg.com -i 04t...
```

```
$ sfdx force:package:hammertest:list -u you@yourpackagingorg.com
```



## Command Syntax

```
sfdx force:package:hammertest:list
  [--json]
  [--loglevel LOGLEVEL]
  [-u TARGETUSERNAME]
  [--apiversion APIVERSION]
  [-i PACKAGEVERSIONID]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-i | --packageversionid PACKAGEVERSIONID**


Optional

The ID of the package version to list test results for (starts with 04t).

Type: id

## **force:package:hammertest:report (Pilot)**

Returns the status of a running ISV Hammer test or the results of a completed hammer test.

 **Note:** We provide the `force:package:hammertest:report` command to selected customers through a pilot program that requires agreement to specific terms and conditions. To be nominated to participate in the program, contact Salesforce. Pilot programs are subject to change, and we can't guarantee acceptance. The `force:package:hammertest:report` command isn't generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. We can't guarantee general availability within any particular time frame or at all. Make your purchase decisions

only on the basis of generally available products and features. You can provide feedback and suggestions for the `force:package:hammer:report` command in the [ISV Hammer - Pilot](#) group in the Trailblazer Community.

## Help for `force:package:hammer:report`

Example:

```
$ sfdx force:package:hammer:report -u you@yourpackagingorg.com -i OTW...
```

## Command Syntax

**sfdx force:package:hammer:report**

```
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-i REQUESTID
[-s]
```

## Parameters

### **--json**

Optional  
Format output as JSON.  
Type: boolean

### **--loglevel LOGLEVEL**

Optional  
The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.  
Type: enum  
Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL  
Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional  
A username or alias for the target org. Overrides the default target org.  
Type: string

### **--apiversion APIVERSION**

Optional  
Override the API version used for API requests made by this command.  
Type: string

### **-i | --requestid REQUESTID**

Required  
The ID of the ISV Hammer request to return the status or results for (starts with OTW).

To list your hammer request IDs, run "sfdx force:package:hammertest:list".

Type: id

**-s | --summary**


Optional

Hides Apex test failures, reporting only a results summary.

Type: boolean

## **force:package:hammertest:run (Pilot)**

Runs ISV Hammer for the specified package version and subscriber orgs.

 **Note:** We provide the `force:package:hammertest:run` command to selected customers through a pilot program that requires agreement to specific terms and conditions. To be nominated to participate in the program, contact Salesforce. Pilot programs are subject to change, and we can't guarantee acceptance. The `force:package:hammertest:run` command isn't generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. We can't guarantee general availability within any particular time frame or at all. Make your purchase decisions only on the basis of generally available products and features. You can provide feedback and suggestions for the `force:package:hammertest:run` command in the [ISV Hammer - Pilot](#) group in the Trailblazer Community.

### Help for **force:package:hammertest:run**

Use this command to run ISV Hammer for the specified package versions and subscriber orgs.

If Salesforce has enabled Multiple Managed Packages for your org, you can test more than one package version. Supply the IDs of the package versions you want to test in the desired order of execution.

The command creates a copy of each of the supplied subscriber orgs and tests the package upgrade in each of them.

If a scheduled date/time is supplied, validation and setup start immediately but the package upgrade test will not start until after the supplied date/time. Format the `--scheduledrundatetime` value as YYYY-MM-DD HH:mm:ss in the GMT timezone. This date cannot be more than 25 days from today.

To test against the Salesforce preview release version, during the sandbox preview window, include `--preview` and a `--scheduledrundatetime` value that's immediately after a staggered release, based on the Salesforce release calendar.

To run the package version's Apex tests in each subscriber org after successfully testing the package upgrade, include `--apextests`.

The command returns a request ID (starts with 0TW). Use this `IsvHammerRequest` ID to query for the test status by running "sfdx force:package:hammertest:report -i 0TW...".

Examples:

```
$ sfdx force:package:hammertest:run -u you@yourpackagingorg.com -i "04t...a,04t...b"
-s "00D...a,00Dx...b"
```

```
$ sfdx force:package:hammertest:run -u you@yourpackagingorg.com -i 04t... -f
subscriberOrgs.txt
```

### Command Syntax

**sfdx force:package:hammertest:run**

[--json]

[--loglevel LOGLEVEL]

```

[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-i PACKAGEVERSIONIDS
[-s SUBSCRIBERORGS]
[-f SUBSCRIBERFILE]
[-d SCHEDULEDRUNDATETIME]
[-p]
[-t]
[-n APEXTESTINTERFACE]

```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-i | --packageversionids PACKAGEVERSIONIDS**

Required

A comma-separated list of one or more package version IDs to upgrade or install (each starts with 04t). You can include more than one package version ID only if Salesforce has enabled Multiple Managed Packages for your org. The order of the list specifies the order of the upgrades or installations.

Type: array

### **-s | --subscriberorgs SUBSCRIBERORGS**

Optional

A comma-separated list of one or more subscriber org IDs (each starts with 00D) to run the hammer test on. Either -s or -f is required.

Type: array

**-f | --subscriberfile SUBSCRIBERFILE**

Optional

The path to a file containing a list of subscriber orgs IDs (each starts with 00D), one per line, to run the hammer test on. Either -s or -f required.

Type: string

**-d | --scheduledrundatetime SCHEDULEDRUNDATETIME**

Optional

The earliest date and time when the package upgrade test can run (in GMT, in the YYYY-MM-DD HH:mm:ss format).

Type: string

**-p | --preview**

Optional

Runs the package ISV Hammer test in the upcoming Salesforce preview version.

If you include this flag, include a --scheduledrundatetime value that's immediately after a staggered release, based on the Salesforce release calendar.

Type: boolean

**-t | --apextests**

Optional

After successfully testing the package upgrade, runs the Apex tests in the package version in each subscriber org.

Type: boolean

**-n | --apextestinterface APEXTESTINTERFACE**

Optional

After package upgrade validation, runs the Apex tests in the subscriber org that implement an interface exposed from a managed package. The parameter value for interface name must be fully qualified.

Type: string

**force:package:install**

Installs a package in the target org.

**Help for force:package:install**

Supply the ID of the package version to install. The package installs in your default target org unless you supply the username for a different target org.

For package upgrades, to specify options for component deprecation or deletion of removed components, include an --upgradetype value. To delete components that can be safely deleted and deprecate the others, specify --upgradetype Mixed (the default). To deprecate all removed components, specify --upgradetype DeprecateOnly. To delete all removed components, except for custom objects and custom fields, that don't have dependencies, specify --upgradetype Delete. (Note: This option can result in the loss of data that is associated with the deleted components.) The default is Mixed.

Examples:

```
$ sfdx force:package:install --package 04t... -u me@example.com
```

```
$ sfdx force:package:install --package awesome_package_alias
```

```
$ sfdx force:package:install --package "Awesome Package Alias"
```

```
$ sfdx force:package:install --package 04t... -t DeprecateOnly
```

## Command Syntax

### **sfdx force:package:install**

```
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
[-w WAIT]
[-k INSTALLATIONKEY]
[-b PUBLISHWAIT]
[-r]
[-p PACKAGE]
[-a APEXCOMPILE]
[-s SECURITYTYPE]
[-t UPGRADETYPE]
```

## Parameters

### **--json**

Optional  
Format output as JSON.  
Type: boolean

### **--loglevel LOGLEVEL**

Optional  
The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.  
Type: enum  
Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL  
Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional  
A username or alias for the target org. Overrides the default target org.  
Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-w | --wait WAIT**

Optional

Maximum number of minutes to wait for installation status. The default is 0.

Type: minutes

**-k | --installationkey INSTALLATIONKEY**

Optional

Installation key for installing a key-protected package. The default is null.

Type: string

**-b | --publishwait PUBLISHWAIT**

Optional

Maximum number of minutes to wait for the Subscriber Package Version ID to become available in the target org before canceling the install request. The default is 0.

Type: minutes

**-r | --noprompt**

Optional

Allows the following without an explicit confirmation response: 1) Remote Site Settings and Content Security Policy websites to send or receive data, and 2) --upgradetype Delete to proceed.

Type: boolean

**-p | --package PACKAGE**

Optional

The ID (starts with 04t) or alias of the package version to install.

Type: string

**-a | --apexcompile APEXCOMPILE**

Optional

Applies to unlocked packages only. Specifies whether to compile all Apex in the org and package, or only the Apex in the package.

Type: enum

Permissible values are: all, package

Default value: all

**-s | --securitytype SECURITYTYPE**

Optional

Security access type for the installed package.

Deprecation notice: The --securitytype parameter's default value will change from AllUsers to AdminsOnly in an upcoming release (v47.0 or later).

Type: enum

Permissible values are: AllUsers, AdminsOnly

Default value: AdminsOnly

**-t | --upgradetype UPGRADETYPE**

Optional

For package upgrades, specifies whether to mark all removed components as deprecated (DeprecateOnly), to delete removed components that can be safely deleted and deprecate the others (Mixed), or to delete all removed components, except for custom objects and custom fields, that don't have dependencies (Delete). The default is Mixed. Can specify DeprecateOnly or Delete only for unlocked package upgrades.

Type: enum

Permissible values are: DeprecateOnly, Mixed, Delete

Default value: Mixed

## **force:package:install:report**

Retrieves the status of a package installation request.

### Help for **force:package:install:report**

Examples:

```
$ sfdx force:package:install:report -i 0Hf...
```

```
$ sfdx force:package:install:report -i 0Hf... -u me@example.com
```

## Command Syntax

### **sfdx force:package:install:report**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

-i REQUESTID

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn



**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-i | --requestid REQUESTID**

Required

The ID of the package install request you want to check.

Type: id

## **force:package:installed:list**

Lists all packages installed in the target org.

### Help for **force:package:installed:list**

Examples:

```
$ sfdx force:package:installed:list
```

```
$ sfdx force:package:installed:list -u me@example.com
```

## Command Syntax

**sfdx force:package:installed:list**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

## **force:package:list**

Lists all packages in the Dev Hub org.

### Help for **force:package:list**

You can view the namespace, IDs, and other details for each package.

Examples:

```
$ sfdx force:package:list -v devhub@example.com
```

```
$ sfdx force:package:list -v devhub@example.com --verbose
```

## Command Syntax

**sfdx force:package:list**

[--json]

[--loglevel LOGLEVEL]

[-v TARGETDEVHUBUSERNAME]

[--apiversion APIVERSION]

[--verbose]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**--verbose**

Optional

Displays extended package details.

Type: boolean

## **force:package:uninstall**

Uninstalls a second-generation package from the target org. To uninstall a first-generation package, use the Salesforce user interface.

### Help for **force:package:uninstall**

Specify the package ID for a second-generation package.

Examples:

```
$ sfdx force:package:uninstall -p 04t... -u me@example.com
```

```
$ sfdx force:package:uninstall -p undesirable_package_alias
```

```
$ sfdx force:package:uninstall -p "Undesirable Package Alias"
```

To list the org's installed packages, run "sfdx force:package:installed:list".

To uninstall a first-generation package, from Setup, enter Installed Packages in the Quick Find box, then select Installed Packages.

## Command Syntax

**sfdx force:package:uninstall**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-w WAIT]

[-p PACKAGE]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-w | --wait WAIT**

Optional

Maximum number of minutes to wait for uninstall status. The default is 0.

Type: minutes

**-p | --package PACKAGE**

Optional

The ID (starts with 04t) or alias of the package version to uninstall.

Type: string

## **force:package:uninstall:report**

Retrieves the status of a package uninstall request.

### Help for **force:package:uninstall:report**

Examples:

```
$ sfdx force:package:uninstall:report -i 06y...
```

```
$ sfdx force:package:uninstall:report -i 06y... -u me@example.com
```

## Command Syntax

```
sfdx force:package:uninstall:report  
  [--json]  
  [--loglevel LOGLEVEL]  
  [-u TARGETUSERNAME]  
  [--apiversion APIVERSION]  
  -i REQUESTID
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-i | --requestid REQUESTID**

Required

The ID of the package uninstall request you want to check.

Type: id

## **force:package:update**

Updates details about a package. Does not create a package version.

## Help for **force:package:update**

Specify a new value for each option you want to update.

Examples:

```
$ sfdx force:package:update -p "Your Package Alias" -n "New Package Name"
```

```
$ sfdx force:package:update -p 0Ho... -d "New Package Description"
```

Run "sfdx force:package:list" to list all packages in the Dev Hub org.

## Command Syntax

### **sfdx force:package:update**

```
[--json]
[--loglevel LOGLEVEL]
[-v TARGETDEVHUBUSERNAME]
[--apiversion APIVERSION]
-p PACKAGE
[-n NAME]
[-d DESCRIPTION]
[-o ERRORNOTIFICATIONUSERNAME]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-p | --package PACKAGE**

Required

The ID (starts with 0Ho) or alias of the package to update.

Type: string

**-n | --name NAME**

Optional

New name of the package.

Type: string

**-d | --description DESCRIPTION**

Optional

New description of the package.

Type: string

**-o | --errornotificationusername ERRORNOTIFICATIONUSERNAME**

Optional

An active Dev Hub org user designated to receive email notifications for unhandled Apex exceptions, and install, upgrade, or uninstall failures associated with your package.

Type: string

## **force:package:version:create**

Creates a package version in the Dev Hub org.

### Help for **force:package:version:create**

The package version is based on the package contents in the specified directory.

To retrieve details about a package version create request, including status and package version ID (04t), run "sfdx force:package:version:create:report -i 08c...".

We recommend specifying the --installationkey to protect the contents of your package and to prevent unauthorized installation of your package.

To list package version creation requests in the org, run "sfdx force:package:version:create:list".

Managed packages must use the --codecoverage option and meet the code coverage requirements to promote a version to released.

Starting in Winter '21, only unlocked package versions that have met the minimum 75% code coverage requirement can be promoted. Code coverage minimums aren't enforced on org-dependent unlocked packages.

Examples:

```
$ sfdx force:package:version:create -d common -k password123
```

```
$ sfdx force:package:version:create -p "Your Package Alias" -k password123
```

```
$ sfdx force:package:version:create -p 0Ho... -k password123
```

```
$ sfdx force:package:version:create -d common -k password123 --skipvalidation
```

## Command Syntax

**sfdx force:package:version:create**

[--json]

[--loglevel LOGLEVEL]

```

[-v TARGETDEVHUBUSERNAME]
[--apiversion APIVERSION]
[-p PACKAGE]
[-d PATH]
[-f DEFINITIONFILE]
[-b BRANCH]
[-t TAG]
[-k INSTALLATIONKEY]
[-x]
[-w WAIT]
[-a VERSIONNAME]
[-n VERSIONNUMBER]
[-e VERSIONDESCRIPTION]
[-c]
[--releasenotesurl RELEASENOTESURL]
[--postinstallurl POSTINSTALLURL]
[--postinstallscript POSTINSTALLSCRIPT]
[--uninstallscript UNINSTALLSCRIPT]
[--skipvalidation]

```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.



Type: string

**-p | --package PACKAGE**

Optional

The ID (starts with 0Ho) or alias of the package to create a version of.

Type: string

**-d | --path PATH**

Optional

The path to the directory that contains the contents of the package.

Type: directory

**-f | --definitionfile DEFINITIONFILE**

Optional

The path to a definition file similar to scratch org definition file that contains the list of features and org preferences that the metadata of the package version depends on.

Type: filepath

**-b | --branch BRANCH**

Optional

The package version's branch.

Type: string

**-t | --tag TAG**

Optional

The package version's tag.

Type: string

**-k | --installationkey INSTALLATIONKEY**

Optional

Installation key for creating the key-protected package. Either an --installationkey value or the --installationkeybypass flag is required.

Type: string

**-x | --installationkeybypass**

Optional

Bypasses the installation key requirement. If you bypass this requirement, anyone can install your package. Either an --installationkey value or the --installationkeybypass flag is required.

Type: boolean

**-w | --wait WAIT**

Optional

The number of minutes to wait for the package version to be created.

Type: minutes

Default value: 0 minutes

**-a | --versionname VERSIONNAME**

Optional

The name of the package version to be created. Overrides the sfdx-project.json value.

Type: string

**-n | --versionnumber VERSIONNUMBER**

Optional

The version number of the package version to be created. Overrides the sfdx-project.json value.

Type: string

**-e | --versiondescription VERSIONDESCRIPTION**

Optional

The description of the package version to be created. Overrides the sfdx-project.json value.

Type: string

**-c | --codecoverage**

Optional

Calculate and store the code coverage percentage by running the Apex tests included in this package version. Managed packages must use this option and meet the code coverage requirements to promote the version.

Type: boolean

**--releasenotesurl RELEASENOTESURL**

Optional

The release notes URL. This link is displayed in the package installation UI to provide release notes for this package version to subscribers.

Type: url

**--postinstallurl POSTINSTALLURL**

Optional

The post-install instructions URL. The contents of the post-installation instructions URL are displayed in the UI after installation of the package version.

Type: url

**--postinstallscript POSTINSTALLSCRIPT**

Optional

Applies to managed packages only. The post-install script name. The post-install script is an Apex class within this package that is run in the installing org after installations or upgrades of this package version.

Type: string

**--uninstallscript UNINSTALLSCRIPT**

Optional

Applies to managed packages only. The uninstall script name. The uninstall script is an Apex class within this package that is run in the installing org after uninstalls of this package.

Type: string

**--skipvalidation**

Optional

Skips validation of dependencies, package ancestors, and metadata during package version creation. Skipping validation reduces the time it takes to create a new package version, but you can promote only validated package versions. Skipping validation can suppress important errors that can surface at a later stage. You can specify skip validation or code coverage, but not both. Code coverage is calculated during validation.

Type: boolean

**force:package:version:create:list**

Lists all requests to create second-generation package versions in the Dev Hub org.

**Help for force:package:version:create:list**

Shows the details of each request to create a package version in the Dev Hub org.

All filter parameters are applied using the AND logical operator (not OR).

To get information about a specific request, run "sfdx force:package:version:create:report" and supply the request ID.

Examples:

```
$ sfdx force:package:version:create:list
```

```
$ sfdx force:package:version:create:list --createdlastdays 3
```

```
$ sfdx force:package:version:create:list --status Error
```

```
$ sfdx force:package:version:create:list -s InProgress
```

```
$ sfdx force:package:version:create:list -c 3 -s Success
```

**Command Syntax**

**sfdx force:package:version:create:list**

[--json]

[--loglevel LOGLEVEL]

[-v TARGETDEVHUBUSERNAME]

[--apiversion APIVERSION]

[-c CREATEDLASTDAYS]

[-s STATUS]

**Parameters****--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-c | --createdlastdays CREATEDLASTDAYS**

Optional

Filters the list based on the specified maximum number of days since the request was created (starting at 00:00:00 of first day to now; 0 for today).

Type: number

**-s | --status STATUS**

Optional

Filters the list based on the status of version creation requests.

Type: enum

Permissible values are: Queued, InProgress, Success, Error

## **force:package:version:create:report**

Retrieves details about a package version creation request in the Dev Hub org.

### Help for **force:package:version:create:report**

Specify the request ID for which you want to view details. If applicable, the command displays errors related to the request.

Examples:

```
$ sfdx force:package:version:create:report -i 08c...
```

```
$ sfdx force:package:version:create:report -i 08c... -v devhub@example.com
```

To show all requests in the org, run "sfdx force:package:version:create:list".

## Command Syntax

### **sfdx force:package:version:create:report**

[--json]

[--loglevel LOGLEVEL]

[-v TARGETDEVHUBUSERNAME]

[--apiversion APIVERSION]

-i PACKAGECREATEREQUESTID

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-i | --packagecreaterequestid PACKAGECREATEREQUESTID**

Required

The ID (starts with 08c) of the package version creation request you want to display.

Type: id

## **force:package:version:delete**

Delete unlocked and second-generation managed package versions.

### Help for **force:package:version:delete**

Specify the ID or alias of the package version you want to delete.

Examples:

```
$ sfdx force:package:version:delete -p "Your Package Alias"
```

```
$ sfdx force:package:version:delete -p 04t...
```

## Command Syntax

**sfdx force:package:version:delete**

[--json]

[--loglevel LOGLEVEL]

[-v TARGETDEVHUBUSERNAME]

[--apiversion APIVERSION]

[-n]

`-p PACKAGE`

## Parameters

### `--json`

Optional

Format output as JSON.

Type: boolean

### `--loglevel LOGLEVEL`

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### `-v | --targetdevhubusername TARGETDEVHUBUSERNAME`

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

### `--apiversion APIVERSION`

Optional

Override the API version used for API requests made by this command.

Type: string

### `-n | --noprompt`

Optional

don't prompt before deleting the package version

Type: boolean

### `-p | --package PACKAGE`


Required

The ID (starts with 04t) or alias of the package version to delete.

Type: string

## `force:package:version:displayancestry` (Beta)

Displays the ancestry tree for a 2GP managed package version.

 **Note:** This release contains a beta version of the `force:package:version:displayancestry` command, which means it's a high-quality feature with known limitations. The `force:package:version:displayancestry` command isn't generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. We can't guarantee general availability within any particular time frame or at all. Make your purchase decisions only on the basis of generally available products and features. You can provide feedback and suggestions for the `force:package:version:displayancestry` command in the [Unlocked Packages and Managed 2GPs](#) group in the Trailblazer Community.

## Help for `force:package:version:displayancestry`

Examples:

```
$ sfdx force:package:version:displayancestry -p package_version_alias
```

```
$ sfdx force:package:version:displayancestry -p package_version_alias --dotcode
```

```
$ sfdx force:package:version:displayancestry -p OHo...
```

```
$ sfdx force:package:version:displayancestry -p 04t...
```

## Command Syntax

**sfdx force:package:version:displayancestry**

```
[--json]  
[--loglevel LOGLEVEL]  
[-v TARGETDEVHUBUSERNAME]  
[--apiversion APIVERSION]  
-p PACKAGE  
[--dotcode]  
[--verbose]
```

## Parameters

### **--json**

Optional  
Format output as JSON.  
Type: boolean

### **--loglevel LOGLEVEL**

Optional  
The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.  
Type: enum  
Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL  
Default value: warn

### **-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional  
A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.  
Type: string

### **--apiversion APIVERSION**

Optional  
Override the API version used for API requests made by this command.  
Type: string

**-p | --package PACKAGE**

Required

The ID or alias of the package or package version to display ancestry for. If you specify a package ID (starts with 0Ho) or alias, the ancestor tree for every package version associated with the package ID is displayed.

If you specify a package version (starts with 04t) or alias, the ancestry tree of the specified package version is displayed.

Type: string

**--dotcode**

Optional

Displays the ancestry tree in DOT code. You can use the DOT code output in graph visualization software to create tree visualizations.

Type: boolean

**--verbose**

Optional

Displays both the package version ID (starts with 04t) and the version number (major.minor.patch.build) in the ancestry tree.

Type: boolean

**force:package:version:list**

Lists all package versions in the Dev Hub org.

**Help for force:package:version:list**

Displays details of each package version in the org.

Use `--concise` or `--verbose` to display limited or additional details, respectively.

All filter parameters are applied using the AND logical operator (not OR).

Examples:

```
$ sfdx force:package:version:list --verbose --createdlastdays 3 --released --orderby PatchVersion
```

```
$ sfdx force:package:version:list --packages 0Ho000000000000,0Ho000000000001 --released --modifiedlastdays 0
```

```
$ sfdx force:package:version:list --released
```

```
$ sfdx force:package:version:list --concise --modifiedlastdays 0
```

```
$ sfdx force:package:version:list --concise -c 3 -r
```

```
$ sfdx force:package:version:list --packages exp-mgr,exp-mgr-util --released --modifiedlastdays 0
```

**Command Syntax**

**sfdx force:package:version:list**

[--json]

[--loglevel LOGLEVEL]



```

[-v TARGETDEVHUBUSERNAME]
[--apiversion APIVERSION]
[-c CREATEDLASTDAYS]
[--concise]
[-m MODIFIEDLASTDAYS]
[-p PACKAGES]
[-r]
[-o ORDERBY]
[--verbose]

```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-c | --createdlastdays CREATEDLASTDAYS**

Optional

Filters the list based on the specified maximum number of days since the request was created (starting at 00:00:00 of first day to now; 0 for today).

Type: number

### **--concise**

Optional

Displays limited package version details.

Type: boolean

**-m | --modifiedlastdays MODIFIEDLASTDAYS**

Optional

Lists the items modified in the specified last number of days, starting at 00:00:00 of first day to now. Use 0 for today.

Type: number

**-p | --packages PACKAGES**

Optional

Filters results on the specified comma-delimited packages (aliases or OHO IDs).

Type: array

**-r | --released**

Optional

Displays released versions only (IsReleased=true).

Type: boolean

**-o | --orderby ORDERBY**

Optional

Orders the list by the specified package version fields.

Type: array

**--verbose**

Optional

Displays extended package version details.

Type: boolean

**force:package:version:promote**

Promotes a package version to released status.

**Help for force:package:version:promote**

Supply the ID or alias of the package version you want to promote. Promotes the package version to released status.

Examples:

```
$ sfdx force:package:version:promote -p 04t...
```

```
$ sfdx force:package:version:promote -p awesome_package_alias
```

```
$ sfdx force:package:version:promote -p "Awesome Package Alias"
```

**Command Syntax****sfdx force:package:version:promote**

[--json]

[--loglevel LOGLEVEL]

[-v TARGETDEVHUBUSERNAME]

[--apiversion APIVERSION]

```
-p PACKAGE  
[-n]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-p | --package PACKAGE**

Required

The ID (starts with 04t) or alias of the package version to promote.

Type: string

### **-n | --noprompt**

Optional

Do not prompt to confirm setting the package version as released.

Type: boolean

## **force:package:version:report**

Retrieves details about a package version in the Dev Hub org.

## Help for **force:package:version:report**

Examples:

```
$ sfdx force:package:version:report -p 04t...
```

```
$ sfdx force:package:version:report -p "Your Package Alias"
```

To update package version values, run "sfdx force:package:version:update".

## Command Syntax

**sfdx force:package:version:report**

[--json]

[--loglevel LOGLEVEL]

[-v TARGETDEVHUBUSERNAME]

[--apiversion APIVERSION]

-p PACKAGE

[--verbose]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-p | --package PACKAGE**

Required

The ID (starts with 04t) or alias of the package to retrieve details for.

Type: string

**--verbose**

Optional

Displays extended package version details.

Type: boolean

**force:package:version:update**

Updates a second-generation package version in the Dev Hub org.

**Help for force:package:version:update**

Specify a new value for each option you want to update.

Examples:

```
$ sfdx force:package:version:update -p "Your Package Alias" -k password123
```

```
$ sfdx force:package:version:update -p 04t... -b master -t 'Release 1.0.7'
```

```
$ sfdx force:package:version:update -p 04t... -e "New Package Version Description"
```

To display details about a package version, run "sfdx force:package:version:report".

**Command Syntax****sfdx force:package:version:update**

```
[--json]
[--loglevel LOGLEVEL]
[-v TARGETDEVHUBUSERNAME]
[--apiversion APIVERSION]
-p PACKAGE
[-a VERSIONNAME]
[-e VERSIONDESCRIPTION]
[-b BRANCH]
[-t TAG]
[-k INSTALLATIONKEY]
```

**Parameters****--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-p | --package PACKAGE**

Required

The ID (starts with 04t) or alias of the package to update a version of.

Type: string

**-a | --versionname VERSIONNAME**

Optional

The new package version name.

Type: string

**-e | --versiondescription VERSIONDESCRIPTION**

Optional

The new package version description.

Type: string

**-b | --branch BRANCH**

Optional

The new package version branch.

Type: string

**-t | --tag TAG**

Optional

The new package version tag.

Type: string

**-k | --installationkey INSTALLATIONKEY**

Optional

The new installation key for the key-protected package. The default is null.

Type: string

## package1 Commands

Use the package1 commands to create and view first-generation package versions in your Dev Hub org.

[force:package1:version:create](#)

Creates a first-generation package version in the release org.

[force:package1:version:create:get](#)

Retrieves the status of a package version creation request.

[force:package1:version:display](#)

Displays detailed information about an individual first-generation package version.

[force:package1:version:list](#)

Lists the versions for the specified package or all first-generation packages in the org.

## **force:package1:version:create**

Creates a first-generation package version in the release org.

### Help for **force:package1:version:create**

The package version is based on the contents of the specified metadata package. Omit -m if you want to create an unmanaged package version.

### Command Syntax

**sfdx force:package1:version:create**

`[--json]`

`[--loglevel LOGLEVEL]`

`[-u TARGETUSERNAME]`

`[--apiversion APIVERSION]`

`-i PACKAGEID`

`-n NAME`

`[-d DESCRIPTION]`

`[-v VERSION]`

`[-m]`

`[-r RELEASENOTESURL]`

`[-p POSTINSTALLURL]`

`[-k INSTALLATIONKEY]`

`[-w WAIT]`

### Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-i | --packageid PACKAGEID**

Required

ID of the metadata package (starts with 033) of which you're creating a new version.

Type: id

**-n | --name NAME**

Required

Package version name.

Type: string

**-d | --description DESCRIPTION**

Optional

Package version description.

Type: string

**-v | --version VERSION**

Optional

Package version in major.minor format, for example, 3.2.

Type: string

**-m | --managedreleased**

Optional

Creates a managed package version. To create a beta version, don't include this parameter.

Type: boolean

**-r | --releasenotesurl RELEASENOTESURL**

Optional

The release notes URL. This link is displayed in the package installation UI to provide release notes for this package version to subscribers.

Type: url

**-p | --postinstallurl POSTINSTALLURL**

Optional

The post-install instructions URL. The contents of the post-installation instructions URL are displayed in the UI after installation of the package version.

Type: url



**-k | --installationkey INSTALLATIONKEY**

Optional

Installation key for creating the key-protected package. The default is null.

Type: string

**-w | --wait WAIT**

Optional

Minutes to wait for the package version to be created. The default is 2 minutes.

Type: number

**force:package1:version:create:get**

Retrieves the status of a package version creation request.

**Help for force:package1:version:create:get**

Examples:

```
$ sfdx force:package:version:create:report -i 08c...
```

```
$ sfdx force:package:version:create:report -i 08c... -v devhub@example.com
```

**Command Syntax****sfdx force:package1:version:create:get**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

-i REQUESTID

**Parameters****--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

- u | --targetusername TARGETUSERNAME**  
 Optional  
 A username or alias for the target org. Overrides the default target org.  
 Type: string
- apiversion APIVERSION**  
 Optional  
 Override the API version used for API requests made by this command.  
 Type: string
- i | --requestid REQUESTID**  
 Required  
 The ID of the PackageUploadRequest.  
 Type: id

## **force:package1:version:display**

Displays detailed information about an individual first-generation package version.

### Help for **force:package1:version:display**

Display detailed information about an individual package version, including metadata package ID, name, the release state, and build number.

## Command Syntax

```
sfdx force:package1:version:display
  [--json]
  [--loglevel LOGLEVEL]
  [-u TARGETUSERNAME]
  [--apiversion APIVERSION]
  -i PACKAGEVERSIONID
```

## Parameters

- json**  
 Optional  
 Format output as JSON.  
 Type: boolean
- loglevel LOGLEVEL**  
 Optional  
 The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.  
 Type: enum  
 Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-i | --packageversionid PACKAGEVERSIONID**

Required

ID (starts with 04t) of the metadata package version whose details you want to display.

Type: id

## **force:package1:version:list**

Lists the versions for the specified package or all first-generation packages in the org.

### Help for **force:package1:version:list**

If a metadata package ID is specified, lists all versions of the specified package. Otherwise, lists all package versions for the org. For each package version, the list includes the package version ID, metadata package ID, name, version number, and release state.

## Command Syntax

**sfdx force:package1:version:list**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-i PACKAGEID]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-i | --packageid PACKAGEID**

Optional

Metadata package ID (starts with 033) whose package versions you want to list. If not specified, shows all versions for all packages (managed and unmanaged) in the org.

Type: id

## project Commands

Use the project commands to set up a Salesforce DX project.

[force:project:create](#)

Creates a Salesforce DX project in the specified directory or the current working directory. The command creates the necessary configuration files and folders.

[force:project:upgrade](#)

Updates project configuration and definition files to the latest format.

### **force:project:create**

Creates a Salesforce DX project in the specified directory or the current working directory. The command creates the necessary configuration files and folders.

### Help for **force:project:create**

If not supplied, the apiversion, template, and outputdir use default values.

The outputdir can be an absolute path or relative to the current working directory.

Examples:

```
$ sfdx force:project:create --projectname mywork
```

```
$ sfdx force:project:create --projectname mywork --defaultpackagedir myapp
```

```
$ sfdx force:project:create --projectname mywork --defaultpackagedir myapp --manifest
```

```
$ sfdx force:project:create --projectname mywork --template empty
```

## Command Syntax

**sfdx force:project:create**

[--json]

[--loglevel LOGLEVEL]

-n PROJECTNAME

[-t TEMPLATE]

[-d OUTPUTDIR]

[-s NAMESPACE]

[-p DEFAULTPACKAGEDIR]

[-x]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-n | --projectname PROJECTNAME**

Required

The name for the new project. Any valid folder name is accepted.

Type: string

**-t | --template TEMPLATE**

Optional

The template to use to create the project. Supplied parameter values or default values are filled into a copy of the template.

Type: string

Permissible values are: standard, empty, analytics

Default value: standard

**-d | --outputdir OUTPUTDIR**

Optional

The directory to store the newly created files. The location can be an absolute path or relative to the current working directory. The default is the current directory.

Type: string

Default value: .

**-s | --namespace NAMESPACE**

Optional

The namespace associated with this project and any connected scratch orgs.

Type: string

**-p | --defaultpackagedir DEFAULTPACKAGEDIR**

Optional

The default package directory name. Metadata items such as classes and Lightning bundles are placed inside this folder.

Type: string

Default value: force-app

**-x | --manifest**

Optional

Generates a default manifest (package.xml) for fetching Apex, Visualforce, Lightning components, and static resources.

Type: boolean

**force:project:upgrade**

Updates project configuration and definition files to the latest format.

**Help for force:project:upgrade**

Examples:

\$ sfdx force:project:upgrade

\$ sfdx force:project:upgrade -f

**Command Syntax****sfdx force:project:upgrade**

[--json]

[--loglevel LOGLEVEL]

[-f]

**Parameters****--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-f | --forceupgrade**

Optional

Run all upgrades, even if the project definition files have already been upgraded.

Type: boolean

## schema Commands

Use the schema commands to view information about the standard and custom objects in your org.

[force:schema:object:describe](#)

Displays the metadata for a standard or custom object.

[force:schema:object:list](#)

List all objects of a specified category.

### **force:schema:object:describe**

Displays the metadata for a standard or custom object.

### Examples for **force:schema:object:describe**

```
sfdx force:schema:object:describe -s Account
```

```
sfdx force:schema:object:describe -s MyObject__c
```

```
sfdx force:schema:object:describe -s ApexClass -t
```

### Command Syntax

```
sfdx force:schema:object:describe
```

```
  [--json]
```

```
  [--loglevel LOGLEVEL]
```

```
  [-u TARGETUSERNAME]
```

```
  [--apiversion APIVERSION]
```

```
  -s SUBJECTTYPE
```

```
  [-t]
```

### Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-s | --subjecttype SUBJECTTYPE**

Required

The API name of the object to describe.

Type: string

**-t | --usetoolingapi**

Optional

Execute with Tooling API.

Type: boolean

## **force:schema:object:list**

List all objects of a specified category.

### Examples for **force:schema:object:list**

```
sfdx force:schema:object:list -c all
```

```
sfdx force:schema:object:list -c custom
```

```
sfdx force:schema:object:list -c standard
```

## Command Syntax

**sfdx force:schema:object:list**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-c SUBJECTTYPECATEGORY]



## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-c | --objecttypecategory OBJECTTYPECATEGORY**

Optional

The type of objects to list (all|custom|standard).

Type: string

Default value: ALL

## source Commands

Use the source commands to push and pull source to and from your scratch orgs, to deploy and retrieve source to and from non-source-tracked orgs, to see synchronization changes between your project and scratch orgs, and to convert your source to the metadata format for Metadata API deployments.

[force:source:convert](#)

Converts source-formatted files into metadata that you can deploy using Metadata API.

[force:source:delete](#)

Deletes source files from your project and from a non-source-tracked org, such as a sandbox.

[force:source:deploy](#)

Deploys metadata in source format to an org.

[force:source:deploy:cancel](#)

Cancels an asynchronous source deployment.

**force:source:deploy:report**

Checks the current status of an asynchronous metadata deployment.

**force:source:open**

Opens the specified Lightning Page in Lightning App Builder. Lightning Page files have the suffix `.flexipage-meta.xml`, and are stored in the `flexipages` directory. If you specify a different type of file, this command opens your org's home page.

**force:source:pull**

Pulls changed source from the scratch org to your project to keep them in sync.

**force:source:push**

Pushes changed source from your project to a scratch org to keep them in sync.

**force:source:retrieve**

Retrieves metadata in source format from an org to your local Salesforce DX project.

**force:source:status**

Lists changes that have been made locally, in a scratch org, or both.

**force:source:tracking:clear**

Clear all local source tracking information.

**force:source:tracking:reset**

Reset local and remote source tracking.

**force:source:convert**

Converts source-formatted files into metadata that you can deploy using Metadata API.

**Help for force:source:convert**

To convert source-formatted files into the metadata format, so that you can deploy them using Metadata API, run `"sfdx force:source:convert"`. Then deploy the metadata using `"sfdx force:mdapi:deploy"`.

To convert Metadata API-formatted files into the source format, run `"sfdx force:mdapi:convert"`.

To specify a package name that includes spaces, enclose the name in single quotes.

Examples:

```
$ sfdx force:source:convert -r path/to/source
```

```
$ sfdx force:source:convert -r path/to/source -d path/to/outputdir -n 'My Package'
```

**Command Syntax****sfdx force:source:convert**

`[--json]`

`[--loglevel LOGLEVEL]`

`[-r ROOTDIR]`

`[-d OUTPUTDIR]`

`[-n PACKAGENAME]`

`[-x MANIFEST]`

`[-p SOURCEPATH]`

`[-m METADATA]`

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-r | --rootdir ROOTDIR**

Optional

A directory other than the default package directory that contains the source-formatted files to convert.

Type: directory

### **-d | --outputdir OUTPUTDIR**

Optional

The output directory to store the Metadata API-formatted metadata files in.

Type: directory

### **-n | --packagename PACKAGENAME**

Optional

The name of the package to associate with the metadata-formatted files.

Type: string

### **-x | --manifest MANIFEST**

Optional

The complete path to the manifest (package.xml) file that specifies the metadata types to convert.

If you specify this parameter, don't specify `--metadata` or `--sourcepath`.

Type: string

### **-p | --sourcepath SOURCEPATH**

Optional

A comma-separated list of paths to the local source files to convert. The supplied paths can be to a single file (in which case the operation is applied to only one file) or to a folder (in which case the operation is applied to all metadata types in the directory and its sub-directories).

If you specify this parameter, don't specify `--manifest` or `--metadata`.

Type: array

**-m | --metadata METADATA**

Optional

A comma-separated list of metadata component names to convert.

Type: array

**force:source:delete**

Deletes source files from your project and from a non-source-tracked org, such as a sandbox.

**Help for force:source:delete**

Use this command to delete components from orgs that don't have source tracking, such as sandboxes.

To remove deleted items from scratch orgs, which have change tracking, use "sfdx force:source:push".

Examples:

```
$ sfdx force:source:delete -p path/to/source
```

```
$ sfdx force:source:delete -m <metadata>
```

**Command Syntax****sfdx force:source:delete**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-c]

[-r]

[-w WAIT]

[-p SOURCEPATH]

[-m METADATA]

**Parameters****--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-c | --checkonly**

Optional

Validates the deleted metadata and runs all Apex tests, but prevents the deletion from being saved to the org.

If you change a field type from Master-Detail to Lookup or vice versa, that change isn't supported when using the `--checkonly` parameter to test a deletion (validation). This kind of change isn't supported for test deletions to avoid the risk of data loss or corruption. If a change that isn't supported for test deletions is included in a deletion package, the test deletion fails and issues an error.

If your deletion package changes a field type from Master-Detail to Lookup or vice versa, you can still validate the changes prior to deploying to Production by performing a full deletion to another test Sandbox. A full deletion includes a validation of the changes as part of the deletion process.

Note: A Metadata API deletion that includes Master-Detail relationships deletes all detail records in the Recycle Bin in the following cases.

1. For a deletion with a new Master-Detail field, soft delete (send to the Recycle Bin) all detail records before proceeding to delete the Master-Detail field, or the deletion fails. During the deletion, detail records are permanently deleted from the Recycle Bin and cannot be recovered.
2. For a deletion that converts a Lookup field relationship to a Master-Detail relationship, detail records must reference a master record or be soft-deleted (sent to the Recycle Bin) for the deletion to succeed. However, a successful deletion permanently deletes any detail records in the Recycle Bin.

Type: boolean

**-r | --noprompt**

Optional

Do not prompt for delete confirmation.

Type: boolean

**-w | --wait WAIT**

Optional

Number of minutes to wait for the command to complete and display results to the terminal window. If the command continues to run after the wait period, the CLI returns control of the terminal window to you. The default is 33 minutes.

Type: minutes

Default value: 33 minutes

**-p | --sourcepath SOURCEPATH**

Optional

A comma-separated list of paths to the local metadata to delete. The supplied paths can be a single file (in which case the operation is applied to only one file) or a folder (in which case the operation is applied to all metadata types in the directory and its sub-directories).

If you specify this parameter, don't specify `--manifest` or `--metadata`.

Type: array

**-m | --metadata METADATA**

Optional

A comma-separated list of names of metadata components to delete from your project and your org.

Type: array

## **force:source:deploy**

Deploys metadata in source format to an org.

### Help for **force:source:deploy**

Use this command to deploy source (metadata that's in source format) to an org.

To take advantage of change tracking with scratch orgs, use "sfdx force:source:push".

To deploy metadata that's in metadata format, use "sfdx force:mdapi:deploy".

The source you deploy overwrites the corresponding metadata in your org. This command does not attempt to merge your source with the versions in your org.

To run the command asynchronously, set `--wait` to 0, which immediately returns the job ID. This way, you can continue to use the CLI.

To check the status of the job, use `force:source:deploy:report`.

If the comma-separated list you're supplying contains spaces, enclose the entire comma-separated list in one set of double quotes. On Windows, if the list contains commas, also enclose the entire list in one set of double quotes.

Examples:

To deploy the source files in a directory:

```
$ sfdx force:source:deploy -p path/to/source
```

To deploy a specific Apex class and the objects whose source is in a directory:

```
$ sfdx force:source:deploy -p "path/to/apex/classes/MyClass.cls,path/to/source/objects"
```

To deploy source files in a comma-separated list that contains spaces:

```
$ sfdx force:source:deploy -p
"path/to/objects/MyCustomObject/fields/MyField.field-meta.xml, path/to/apex/classes"
```

To deploy all Apex classes:

```
$ sfdx force:source:deploy -m ApexClass
```

To deploy a specific Apex class:

```
$ sfdx force:source:deploy -m ApexClass:MyApexClass
```

To deploy all custom objects and Apex classes:

```
$ sfdx force:source:deploy -m "CustomObject,ApexClass"
```

To deploy all Apex classes and two specific profiles (one of which has a space in its name):

```
$ sfdx force:source:deploy -m "ApexClass, Profile:My Profile, Profile: AnotherProfile"
```

To deploy all components listed in a manifest:

```
$ sfdx force:source:deploy -x path/to/package.xml
```

To run the tests that aren't in any managed packages as part of a deployment:

```
$ sfdx force:source:deploy -m ApexClass -l RunLocalTests
```

To check whether a deployment would succeed (to prepare for Quick Deploy):

```
$ sfdx force:source:deploy -m ApexClass -l RunAllTestsInOrg -c
```

To deploy an already validated deployment (Quick Deploy):

```
$ sfdx force:source:deploy -q 0Af9A00000FTM6pSAH
```

## Command Syntax

### **sfdx force:source:deploy**

```
[--json]  
[--loglevel LOGLEVEL]  
[-u TARGETUSERNAME]  
[--apiversion APIVERSION]  
[-c]  
[-w WAIT]  
[-l TESTLEVEL]  
[-r RUNTESTS]  
[-o]  
[-g]  
[-q VALIDATEDDEPLOYREQUESTID]  
[--verbose]  
[-m METADATA]  
[-p SOURCEPATH]  
[-x MANIFEST]
```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-c | --checkonly**

Optional

Validates the deployed metadata and runs all Apex tests, but prevents the deployment from being saved to the org.

If you change a field type from Master-Detail to Lookup or vice versa, that change isn't supported when using the `--checkonly` parameter to test a deployment (validation). This kind of change isn't supported for test deployments to avoid the risk of data loss or corruption. If a change that isn't supported for test deployments is included in a deployment package, the test deployment fails and issues an error.

If your deployment package changes a field type from Master-Detail to Lookup or vice versa, you can still validate the changes prior to deploying to Production by performing a full deployment to another test Sandbox. A full deployment includes a validation of the changes as part of the deployment process.

Note: A Metadata API deployment that includes Master-Detail relationships deletes all detail records in the Recycle Bin in the following cases.

1. For a deployment with a new Master-Detail field, soft delete (send to the Recycle Bin) all detail records before proceeding to deploy the Master-Detail field, or the deployment fails. During the deployment, detail records are permanently deleted from the Recycle Bin and cannot be recovered.
2. For a deployment that converts a Lookup field relationship to a Master-Detail relationship, detail records must reference a master record or be soft-deleted (sent to the Recycle Bin) for the deployment to succeed. However, a successful deployment permanently deletes any detail records in the Recycle Bin.

Type: boolean

**-w | --wait WAIT**

Optional

Number of minutes to wait for the command to complete and display results to the terminal window. If the command continues to run after the wait period, the CLI returns control of the terminal window to you. The default is 33 minutes.

Type: minutes

Default value: 33 minutes



**-l | --testlevel TESTLEVEL**

Optional

Specifies which level of deployment tests to run. Valid values are:

**NoTestRun**—No tests are run. This test level applies only to deployments to development environments, such as sandbox, Developer Edition, or trial orgs. This test level is the default for development environments.

**RunSpecifiedTests**—Runs only the tests that you specify in the `--runtests` option. Code coverage requirements differ from the default coverage requirements when using this test level. Executed tests must comprise a minimum of 75% code coverage for each class and trigger in the deployment package. This coverage is computed for each class and trigger individually and is different than the overall coverage percentage.

**RunLocalTests**—All tests in your org are run, except the ones that originate from installed managed packages. This test level is the default for production deployments that include Apex classes or triggers.

**RunAllTestsInOrg**—All tests in your org are run, including tests of managed packages.

If you don't specify a test level, the default behavior depends on the contents of your deployment package. For more information, see "Running Tests in a Deployment" in the Metadata API Developer Guide.

Type: enum

Permissible values are: NoTestRun, RunSpecifiedTests, RunLocalTests, RunAllTestsInOrg

**-r | --runtests RUNTESTS**

Optional

Lists the Apex classes containing the deployment tests to run. Use this parameter when you set `--testlevel` to `RunSpecifiedTests`.

Type: array

**-o | --ignoreerrors**

Optional

Ignores the deploy errors, and continues with the deploy operation. The default is false. Keep this parameter set to false when deploying to a production org. If set to true, components without errors are deployed, and components with errors are skipped.

Type: boolean

**-g | --ignorewarnings**

Optional

If a warning occurs and `ignoreWarnings` is set to true, the `success` field in `DeployMessage` is true. When `ignoreWarnings` is set to false, `success` is set to false, and the warning is treated like an error.

This field is available in API version 18.0 and later. Prior to version 18.0, there was no distinction between warnings and errors. All problems were treated as errors and prevented a successful deployment.

Type: boolean

**-q | --validateddeployrequestid VALIDATEDDEPLOYREQUESTID**

Optional

Specifies the ID of a package with recently validated components to run a Quick Deploy. Deploying a validation helps you shorten your deployment time because tests aren't rerun. If you have a recent successful validation, you can deploy the validated components without running tests. A validation doesn't save any components in the org. You use a validation only to check the success or failure messages that you would receive with an actual deployment. To validate your components, add the `-c | --checkonly` flag when you run `"sfdx force:mdapi:deploy"`. This flag sets the `checkOnly="true"` parameter for your deployment. Before deploying a recent validation, ensure that the following requirements are met:

1. The components have been validated successfully for the target environment within the last 10 days.

2. As part of the validation, Apex tests in the target org have passed.
3. Code coverage requirements are met.
  - If all tests in the org or all local tests are run, overall code coverage is at least 75%, and Apex triggers have some coverage.
  - If specific tests are run with the RunSpecifiedTests test level, each class and trigger that was deployed is covered by at least 75% individually.

Type: id

**--verbose**

Optional

Indicates that you want verbose output from the deploy operation.

Type: boolean

**-m | --metadata METADATA**

Optional

A comma-separated list of names of metadata components to deploy to the org.

Type: array

**-p | --sourcepath SOURCEPATH**

Optional

A comma-separated list of paths to the local source files to deploy. The supplied paths can be to a single file (in which case the operation is applied to only one file) or to a folder (in which case the operation is applied to all metadata types in the directory and its sub-directories).

If you specify this parameter, don't specify --manifest or --metadata.

Type: array

**-x | --manifest MANIFEST**

Optional

The complete path for the manifest (package.xml) file that specifies the components to deploy. All child components are included.

If you specify this parameter, don't specify --metadata or --sourcepath.

Type: filepath

## **force:source:deploy:cancel**

Cancels an asynchronous source deployment.

### **Help for force:source:deploy:cancel**

Use this command to cancel a specified asynchronous source deployment. You can also specify a wait time (in minutes) to check for updates to the canceled deploy status.

To run the command asynchronously, set --wait to 0, which immediately returns the job ID. This way, you can continue to use the CLI.

To check the status of the job, use force:source:deploy:report.

Examples:

Deploy a directory of files to the org

```
$ sfdx force:source:deploy -d <directory>
```

Now cancel this deployment and wait two minutes

```
$ sfdx force:source:deploy:cancel -w 2
```

If you have multiple deployments in progress and want to cancel a specific one, specify the job ID

```
$ sfdx force:source:deploy:cancel -i <jobid>
```

Check the status of the cancel job

```
$ sfdx force:source:deploy:report
```

## Command Syntax

**sfdx force:source:deploy:cancel**

`--json`

`--loglevel LOGLEVEL`

`-u TARGETUSERNAME`

`--apiversion APIVERSION`

`-w WAIT`

`-i JOBSITE`

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-w | --wait WAIT**

Optional

Number of minutes to wait for the command to complete and display results to the terminal window. If the command continues to run after the wait period, the CLI returns control of the terminal window to you. The default is 33 minutes.

Type: minutes

Default value: 33 minutes

### **-i | --jobid JOBID**

Optional

The job ID (requestId) of the deployment you want to cancel. If not specified, the default value is the ID of the most recent source deployment you ran using Salesforce CLI.

Type: id

## **force:source:deploy:report**

Checks the current status of an asynchronous metadata deployment.

### Help for **force:source:deploy:report**

Specify the job ID for the deploy you want to check. You can also specify a wait time (minutes) to check for updates to the deploy status.

Examples:

Deploy a directory of files to the org

```
$ sfdx force:source:deploy -d <directory>
```

Now cancel this deployment and wait two minutes

```
$ sfdx force:source:deploy:cancel -w 2
```

If you have multiple deployments in progress and want to cancel a specific one, specify the job ID

```
$ sfdx force:source:deploy:cancel -i <jobid>
```

Check the status of the cancel job

```
$ sfdx force:source:deploy:report
```

## Command Syntax

**sfdx force:source:deploy:report**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-w WAIT]

[-i JOBID]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-w | --wait WAIT**

Optional

Number of minutes to wait for the command to complete and display results to the terminal window. If the command continues to run after the wait period, the CLI returns control of the terminal window to you. The default is 33 minutes.

Type: minutes

Default value: 33 minutes

**-i | --jobid JOBID**

Optional

The job ID (asynclid) of the deployment you want to check. If not specified, the default value is the ID of the most recent metadata deployment you ran using Salesforce CLI. Use with -w to resume waiting.

Type: id

## **force:source:open**

Opens the specified Lightning Page in Lightning App Builder. Lightning Page files have the suffix .flexipage-meta.xml, and are stored in the flexipages directory. If you specify a different type of file, this command opens your org's home page.

### **Help for force:source:open**

The file opens in your default browser.

If no browser-based editor is available for the selected file, this command opens your org's home page.

To generate a URL for the browser-based editor but not open the editor, use --urlonly.

Examples:

```
$ sfdx force:source:open -f Property_Record_Page.flexipage-meta.xml
```

```
$ sfdx force:source:open -f Property_Record_Page.flexipage-meta.xml -r
```

## Command Syntax

**sfdx force:source:open**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

-f SOURCEFILE

[-r]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-f | --sourcefile SOURCEFILE**

Required

File to edit.

Type: filepath

**-r | --urlonly**

Optional

Generate a navigation URL path, but don't launch a browser-based editor.

Type: boolean

## **force:source:pull**

Pulls changed source from the scratch org to your project to keep them in sync.

### Help for **force:source:pull**

If the command detects a conflict, it displays the conflicts but does not complete the process. After reviewing the conflict, rerun the command with the `--forceoverwrite` parameter.

### Command Syntax

```
sfdx force:source:pull  
  [--json]  
  [--loglevel LOGLEVEL]  
  [-u TARGETUSERNAME]  
  [--apiversion APIVERSION]  
  [-w WAIT]  
  [-f]
```

### Parameters

#### **--json**

Optional

Format output as JSON.

Type: boolean

#### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

#### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

#### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-w | --wait WAIT**

Optional

The number of minutes to wait for the command to complete and display results to the terminal window. If the command continues to run after the wait period, the CLI returns control of the terminal window to you. The default is 33 minutes.

Type: minutes

Default value: 33 minutes

**-f | --forceoverwrite**

Optional

Runs the pull command even if conflicts exist. Changes in the scratch org overwrite changes in the project.

Type: boolean

**force:source:push**

Pushes changed source from your project to a scratch org to keep them in sync.

**Help for force:source:push**

If the command detects a conflict, it displays the conflicts but does not complete the process. After reviewing the conflict, rerun the command with the `--forceoverwrite` parameter.

**Command Syntax****sfdx force:source:push**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-f]

[-g]

[-w WAIT]

**Parameters****--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL



Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-f | --forceoverwrite**

Optional

Runs the push command even if conflicts exist. Changes in the project overwrite changes in the scratch org.

Type: boolean

**-g | --ignorewarnings**

Optional

Completes the deployment even if warnings are generated.

Type: boolean

**-w | --wait WAIT**

Optional

Number of minutes to wait for the command to complete and display results to the terminal window. If the command continues to run after the wait period, the CLI returns control of the terminal window to you. The default is 33 minutes.

Type: minutes

Default value: 33 minutes

## **force:source:retrieve**

Retrieves metadata in source format from an org to your local Salesforce DX project.

### **Help for force:source:retrieve**

Use this command to retrieve source (metadata that's in source format) from an org.

To take advantage of change tracking with scratch orgs, use "sfdx force:source:pull".

To retrieve metadata that's in metadata format, use "sfdx force:mdapi:retrieve".

The source you retrieve overwrites the corresponding source files in your local project. This command does not attempt to merge the source from your org with your local source files.

If the comma-separated list you're supplying contains spaces, enclose the entire comma-separated list in one set of double quotes. On Windows, if the list contains commas, also enclose it in one set of double quotes.

Examples:

To retrieve the source files in a directory:

```
$ sfdx force:source:retrieve -p path/to/source
```

To retrieve a specific Apex class and the objects whose source is in a directory:

```
$ sfdx force:source:retrieve -p "path/to/apex/classes/MyClass.cls,path/to/source/objects"
```

To retrieve source files in a comma-separated list that contains spaces:

```
$ sfdx force:source:retrieve -p
"path/to/objects/MyCustomObject/fields/MyField.field-meta.xml, path/to/apex/classes"
```

To retrieve all Apex classes:

```
$ sfdx force:source:retrieve -m ApexClass
```

To retrieve a specific Apex class:

```
$ sfdx force:source:retrieve -m ApexClass:MyApexClass
```

To retrieve all custom objects and Apex classes:

```
$ sfdx force:source:retrieve -m "CustomObject,ApexClass"
```

To retrieve all Apex classes and two specific profiles (one of which has a space in its name):

```
$ sfdx force:source:retrieve -m "ApexClass, Profile:My Profile, Profile: AnotherProfile"
```

To retrieve all metadata components listed in a manifest:

```
$ sfdx force:source:retrieve -x path/to/package.xml
```

To retrieve metadata from a package or multiple packages:

```
$ sfdx force:source:retrieve -n MyPackageName
```

```
$ sfdx force:source:retrieve -n "Package1, PackageName With Spaces, Package3"
```

To retrieve all metadata from a package and specific components that aren't in the package, specify both `-n | --packagenames` and one other scoping parameter:

```
$ sfdx force:source:retrieve -n MyPackageName -p path/to/apex/classes
```

```
$ sfdx force:source:retrieve -n MyPackageName -m ApexClass:MyApexClass
```

```
$ sfdx force:source:retrieve -n MyPackageName -x path/to/package.xml
```

## Command Syntax

### **sfdx force:source:retrieve**

```
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[-a APIVERSION]
[-w WAIT]
[-x MANIFEST]
[-m METADATA]
[-n PACKAGENAMES]
```

`[-p SOURCEPATH]`

`[--verbose]`

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **-a | --apiversion APIVERSION**

Optional

Use to override the default, which is the latest version supported by your CLI plug-in, with the version in your package.xml file.

Type: string

### **-w | --wait WAIT**

Optional

Number of minutes to wait for the command to complete and display results to the terminal window. If the command continues to run after the wait period, the CLI returns control of the terminal window to you. The default is 33 minutes.

Type: minutes

Default value: 33 minutes

### **-x | --manifest MANIFEST**

Optional

The complete path for the manifest (package.xml) file that specifies the components to retrieve.

If you specify this parameter, don't specify `--metadata` or `--sourcepath`.

Type: filepath

### **-m | --metadata METADATA**

Optional

A comma-separated list of names of metadata components to retrieve from the org.

Type: array

### **-n | --packagenames PACKAGENAMES**

Optional

A comma-separated list of package names to retrieve.

Type: array

**-p | --sourcepath SOURCEPATH**

Optional

A comma-separated list of file paths for source to retrieve from the org. The supplied paths can be to a single file (in which case the operation is applied to only one file) or to a folder (in which case the operation is applied to all source files in the directory and its sub-directories).

Type: array

**--verbose**

Optional

Indicates that you want verbose output from the retrieve operation.

Type: boolean

## **force:source:status**

Lists changes that have been made locally, in a scratch org, or both.

### Help for **force:source:status**

Examples:

```
$ sfdx force:source:status -l
```

```
$ sfdx force:source:status -r
```

```
$ sfdx force:source:status -a
```

```
$ sfdx force:source:status -a -u me@example.com --json
```

## Command Syntax

**sfdx force:source:status**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-a]

[-l]

[-r]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-a | --all**

Optional

Lists all the changes that have been made.

Type: boolean

**-l | --local**

Optional

Lists the changes that have been made locally.

Type: boolean

**-r | --remote**

Optional

Lists the changes that have been made in the scratch org.

Type: boolean

## **force:source:tracking:clear**

Clear all local source tracking information.

### **Help for force:source:tracking:clear**

WARNING: This command deletes or overwrites all existing source tracking files. Use with extreme caution.

Clears all local source tracking information. When you next run force:source:status, the CLI displays all local and remote files as changed, and any files with the same name are listed as conflicts.

## **Command Syntax**

**sfdx force:source:tracking:clear**

[--json]

```

[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
[-p]

```

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-p | --noprompt**

Optional

Do not prompt for source tracking override confirmation.

Type: boolean

## **force:source:tracking:reset**

Reset local and remote source tracking.

### Help for **force:source:tracking:reset**

WARNING: This command deletes or overwrites all existing source tracking files. Use with extreme caution.

Resets local and remote source tracking so that the CLI no longer registers differences between your local files and those in the org. When you next run `force:source:status`, the CLI returns no results, even though conflicts might actually exist. The CLI then resumes tracking new source changes as usual.

Use the `--revision` parameter to reset source tracking to a specific revision number of an org source member. To get the revision number, query the SourceMember Tooling API object with the `force:data:soql:query` command. For example:

```
$ sfdx force:data:soql:query -q "SELECT MemberName, MemberType, RevisionCounter FROM SourceMember" -t
```

## Command Syntax

**sfdx force:source:tracking:reset**

```
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
[-r REVISION]
[-p]
```

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-r | --revision REVISION**

Optional

Reset to a specific SourceMember revision counter number.

Type: integer

**-p | --noprompt**

Optional

Do not prompt for source tracking override confirmation.

Type: boolean

## user Commands

Use the user commands to perform user-related admin tasks.

### [force:user:create](#)

Creates a user for a scratch org.

### [force:user:display](#)

Displays information about a user of a scratch org that the Salesforce CLI has created or authenticated.

### [force:user:list](#)

Lists all users of a scratch org that the Salesforce CLI has created or authenticated.

### [force:user:password:generate](#)

Generates a password for scratch org users. Targets the usernames listed with the `--onbehalfof` parameter or the `--targetusername` parameter. Defaults to the `defaultusername`.

### [force:user:permset:assign](#)

Assigns a named permission set to one or more users of an org.

## **force:user:create**

Creates a user for a scratch org.

### Help for **force:user:create**

Create a user for a scratch org, optionally setting an alias for use by the CLI, assigning permission sets (e.g., `permsets=ps1,ps2`), generating a password (e.g., `generatepassword=true`), and setting User sObject fields.

Examples:

```
$ sfdx force:user:create
```

```
$ sfdx force:user:create -a testuser1 -f config/project-user-def.json
```

```
$ sfdx force:user:create username=testuser1@my.org email=me@my.org permsets=DreamHouse
```

```
$ sfdx force:user:create -f config/project-user-def.json email=me@my.org
generatepassword=true
```

## Command Syntax

### **sfdx force:user:create**

`[--json]`

`[--loglevel LOGLEVEL]`

`[-v TARGETDEVHUBUSERNAME]`

`[-u TARGETUSERNAME]`



[--apiversion APIVERSION]

[-f DEFINITIONFILE]

[-a SETALIAS]

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-f | --definitionfile DEFINITIONFILE**

Optional

File path to a user definition.

Type: filepath

### **-a | --setalias SETALIAS**

Optional

Sets an alias for the created username to reference within the CLI.

Type: string

## **force:user:display**

Displays information about a user of a scratch org that the Salesforce CLI has created or authenticated.

## Help for `force:user:display`

Output includes the profile name, org ID, access token, instance URL, login URL, and alias if applicable.

Examples:

```
$ sfdx force:user:display
```

```
$ sfdx force:user:display -u me@my.org --json
```

## Command Syntax

**sfdx force:user:display**

`[--json]`

`[--loglevel LOGLEVEL]`

`[-v TARGETDEVHUBUSERNAME]`

`[-u TARGETUSERNAME]`

`[--apiversion APIVERSION]`

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in `$HOME/.sfdx/sfdx.log`.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

### **-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**force:user:list**

Lists all users of a scratch org that the Salesforce CLI has created or authenticated.

**Help for force:user:list**

The original scratch org admin is marked with "(A)"

Examples:

```
$ sfdx force:user:list
```

```
$ sfdx force:user:list -u me@my.org --json
```

```
$ sfdx force:user:list --json > tmp/MyUserList.json
```

**Command Syntax****sfdx force:user:list**

```
[--json]
[--loglevel LOGLEVEL]
[-v TARGETDEVHUBUSERNAME]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
```

**Parameters****--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**force:user:password:generate**

Generates a password for scratch org users. Targets the usernames listed with the --onbehalf parameter or the --targetusername parameter. Defaults to the defaultusername.

**Help for force:user:password:generate**

Generates and sets a random password for one or more scratch org users.

If you haven't set a default Dev Hub, or if your scratch org isn't associated with your default Dev Hub, --targetdevhubusername is required.

To see a password that was previously generated, run "sfdx force:user:display".

Examples:

```
$ sfdx force:user:password:generate
```

```
$ sfdx force:user:password:generate -u me@my.org --json
```

```
$ sfdx force:user:password:generate -o "user1@my.org,user2@my.org,user3@my.org"
```

**Command Syntax****sfdx force:user:password:generate**

[--json]

[--loglevel LOGLEVEL]

[-v TARGETDEVHUBUSERNAME]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-o ONBEHALFOF]

**Parameters****--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-v | --targetdevhubusername TARGETDEVHUBUSERNAME**

Optional

A username or alias for the target Dev Hub org. Overrides the default Dev Hub org.

Type: string

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-o | --onbehalfof ONBEHALFOF**

Optional

A comma-separated list of usernames for which to generate passwords.

Type: array

## **force:user:permset:assign**

Assigns a named permission set to one or more users of an org.

### Help for **force:user:permset:assign**

Defaults to the defaultusername.

Examples:

```
$ sfdx force:user:permset:assign -n DreamHouse
```

```
$ sfdx force:user:permset:assign -n DreamHouse -u me@my.org
```

```
$ sfdx force:user:permset:assign -n DreamHouse -o "user1@my.org,user2,user3"
```

## Command Syntax

**sfdx force:user:permset:assign**

[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

-n PERMSETNAME

[-o ONBEHALFOF]

## Parameters

**--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-u | --targetusername TARGETUSERNAME**

Optional

A username or alias for the target org. Overrides the default target org.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-n | --permsetname PERMSETNAME**

Required

The name of the permission set to assign.

Type: string

**-o | --onbehalfof ONBEHALFOF**

Optional

Comma-separated list of usernames or aliases to assign the permission set to.

Type: array

## visualforce Commands

Use the visualforce commands to create Visualforce pages and components.

[force:visualforce:component:create](#)

Creates a Visualforce component in the specified directory or the current working directory. The command creates the .component file and associated metadata file.

[force:visualforce:page:create](#)

Creates a Visualforce page in the specified directory or the current working directory. The command creates the .page file and associated metadata file.

**force:visualforce:component:create**

Creates a Visualforce component in the specified directory or the current working directory. The command creates the .component file and associated metadata file.

**Help for force:visualforce:component:create**

If not supplied, the apiversion, template, and outputdir use default values.

The outputdir can be an absolute path or relative to the current working directory.

Examples:

```
$ sfdx force:visualforce:component:create -n mycomponent -l mylabel
```

```
$ sfdx force:visualforce:component:create -n mycomponent -l mylabel -d components
```

**Command Syntax**

**sfdx force:visualforce:component:create**

```
[--json]
[--loglevel LOGLEVEL]
[-t TEMPLATE]
[-d OUTPUTDIR]
-n COMPONENTNAME
[--apiversion APIVERSION]
-l LABEL
```

**Parameters****--json**

Optional

Format output as JSON.

Type: boolean

**--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

**-t | --template TEMPLATE**

Optional

The template to use to create the file. Supplied parameter values or default values are filled into a copy of the template.

Type: string

Permissible values are: DefaultVFComponent

Default value: DefaultVFComponent

**-d | --outputdir OUTPUTDIR**

Optional

The directory to store the newly created files. The location can be an absolute path or relative to the current working directory. The default is the current directory.

Type: string

Default value: .

**-n | --componentname COMPONENTNAME**

Required

The Visualforce component name. The name can be up to 40 characters and must start with a letter.

Type: string

**--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

**-l | --label LABEL**

Required

The label saved in the metadata for the Visualforce component.

Type: string

## **force:visualforce:page:create**

Creates a Visualforce page in the specified directory or the current working directory. The command creates the .page file and associated metadata file.

### Help for **force:visualforce:page:create**

If not supplied, the apiversion, template, and outputdir use default values.

The outputdir can be an absolute path or relative to the current working directory.

Examples:

```
$ sfdx force:visualforce:page:create -n mypage -l mylabel
```

```
$ sfdx force:visualforce:page:create -n mypage -l mylabel -d pages
```

## Command Syntax

### **sfdx force:visualforce:page:create**

[--json]

[--loglevel LOGLEVEL]

[-t TEMPLATE]

[-d OUTPUTDIR]

-n PAGENAME



[--apiversion APIVERSION]

-l LABEL

## Parameters

### **--json**

Optional

Format output as JSON.

Type: boolean

### **--loglevel LOGLEVEL**

Optional

The logging level for this command invocation. Logs are stored in \$HOME/.sfdx/sfdx.log.

Type: enum

Permissible values are: trace, debug, info, warn, error, fatal, TRACE, DEBUG, INFO, WARN, ERROR, FATAL

Default value: warn

### **-t | --template TEMPLATE**

Optional

The template to use to create the file. Supplied parameter values or default values are filled into a copy of the template.

Type: string

Permissible values are: DefaultVFPPage

Default value: DefaultVFPPage

### **-d | --outputdir OUTPUTDIR**

Optional

The directory to store the newly created files. The location can be an absolute path or relative to the current working directory. The default is the current directory.

Type: string

Default value: .

### **-n | --pagename PAGENAME**

Required

The Visualforce page name. The name can be up to 40 characters and must start with a letter.

Type: string

### **--apiversion APIVERSION**

Optional

Override the API version used for API requests made by this command.

Type: string

### **-l | --label LABEL**

Required

The label saved in the metadata for the Visualforce page.

Type: string

# Help for Salesforce CLI Commands

---

The `-h` | `--help` parameter shows details about Salesforce CLI topics and their commands.

For namespaces, the `-h` | `--help` parameter lists all topics in the namespace. For example, to see names and descriptions of all topics in the `force` namespace, run `sfdx force -h`.

For topics, the `-h` | `--help` parameter lists the commands and their descriptions. For example, to see all commands in the `org` topic, run `sfdx force:org -h`.

For commands, adding the `-h` | `--help` parameter shows parameters and usage information. For example, to see help for the `org:create` command, run `sfdx force:org:create -h`.

Help for commands has four parts.

## 1. Short Description of Command

At the top of the `--help` output (with no heading), a short description of the command is shown. For longer descriptions, see the *Salesforce CLI Command Reference*.

## 2. Usage

The command signature on the Usage line uses the docopt format.

- All available parameters are listed. Parameters that have short names are listed using their short names.
- Parameters that take a value show the value's type (for example, `<string>`) in angle brackets immediately after the parameter's name.
- Optional parameters are in square brackets (`[ ... ]`).
- Required parameters have no annotation.
- For parameters that accept a limited set of values, the values are shown after the parameter name, separated by pipes (`--parametername value1|value2|value3`).
- Mutually exclusive options are shown in parentheses, separated by a pipe (`( ... | ... )`).

If the command takes varargs (name-value pairs that aren't parameters), the usage signature includes `name=value...`



**Tip:** To see signatures for all commands in the `force` namespace, run `sfdx force:doc:commands:list --usage`.

## 3. Options

The Options section lists all the command's parameters, including their short name, long name, and purpose. For parameters that accept a value, the value name is written after an equals sign (`=`). The equals sign is optional when you run the command—for example, you could run `sfdx force:org:create -f=config/enterprise-scratch-def.json -a TestOrg1` or `sfdx force:org:create -f config/enterprise-scratch-def.json -a TestOrg1` with the same results.

Parameters that accept a limited list of values include the values in parentheses, with the default value indicated by an asterisk (\*).

For more information about the parameters, see the *Salesforce CLI Command Reference*.

## 4. Description

Usage notes and examples are below the list of parameters, in the Description section. This information is also available in the *Salesforce CLI Command Reference*.

## CLI Deprecation Policy

---

Salesforce deprecates CLI commands and parameters when, for example, the underlying API changes.

The Salesforce CLI deprecation policy is:

- Salesforce can deprecate a command or parameter in any major update of the `salesforcedx` plug-in.
- Salesforce removes the deprecated command or parameter in the next major release of the `salesforcedx` plug-in. For example, if Salesforce deprecates a command in version 41, it does not appear in version 42.
- If you use a command or parameter that's been deprecated but not yet removed, you get a warning message in `stderr` when you specify human-readable output. If you specify JSON output, the warning is presented as a property. The message includes the plug-in version of when the command or parameter will be removed. The command help also includes deprecation information when appropriate.
- When possible, Salesforce provides a functional alternative to the deprecated command or parameter.
- Salesforce announces new and upcoming deprecated commands and parameters in the release notes.