Salesforce CLI Command Reference

Salesforce, Summer '19

@salesforcedocs

Last updated: August 28, 2019

CONTENTS

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 46.12.1 of the `salesforcedx` CLI plug-in, for version 7.19.0 of Salesforce CLI.

For information about installing Salesforce CLI, see the *Salesforce DX Setup Guide*.

For information about Salesforce CLI changes, see the *Salesforce CLI Release Notes*.

`force` Namespace

Use commands in the `force` namespace to develop on the Salesforce Platform.

`alias` Commands

Use the alias commands to manage username aliases.

`alias:list`

Lists the aliases that the Salesforce CLI can use for various commands and tasks.

Command Syntax

```
sfdx force: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: string

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

Default value: warn

Help for `alias:list`

Example:

```
$ sfdx force:alias:list
```

`alias:set`

Sets an alias that the Salesforce CLI can use for various commands and tasks.

Command Syntax

sfdx force: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: string

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

Default value: warn

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.

To delete an alias, run "sfdx force:alias:set" with no username.

Examples:

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

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

```
$ sfdx force:alias:set AliasToDelete=
```

`apex` Commands

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

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.

apex:execute

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

apex:log:get

Fetches the last debug log.

apex:log:list

Displays a list of debug log IDs, along with general information about the logs.

apex:log:tail

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

apex:test:report

Displays the test results for a specific test run.

apex:test:run

Runs Apex tests.

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.

`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.

Command Syntax

`sfdx force:apex:class:create`
`[--json]`

```
[--loglevel LOGLEVEL]
-n CLASSNAME
[-t TEMPLATE]
[-d OUTPUTDIR]
[-a APIVERSION]
```

Parameters

--json
Optional

Formats output as JSON.

Type: boolean

--loglevel LOGLEVEL
Optional

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

Type: string

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: DefaultApexClass, ApexException, ApexUnitTest, 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

-a | --apiversion APIVERSION
Optional

The API version of the created source.

Type: string

Permissible values are: 46.0, 45.0

Default value: 46.0

Help for `apex:class:create`

`apex:execute`

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

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: string

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

Help for `apex:execute`

Executes one or more lines of Apex code, or executes the code in a local file.

Before you enter code, run this command with no parameters to get a prompt.

From the prompt, all commands are executed in a single execute anonymous request.

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

Examples:

```
$ 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.
```

`apex:log:get`

Fetches the last debug log.

Command Syntax

`sfdx force:apex:log:get`
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
[-c]
[-i LOGID]
[-n NUMBER]

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: string

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

-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

Help for `apex:log:get`

When you execute this command in a project, it fetches the specified log or given number of last logs from your default scratch org.

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

To specify a count of logs to return, use the --number parameter to return the most recent logs.

Executing this command without parameters returns the most recent log.

Examples:

```
$ 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
```

`apex:log:list`

Displays a list of debug log IDs, along with general information about the logs.

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: string

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

Help for `apex:log:list`

When you execute this command in a project, it lists the log IDs for your default scratch org.

Examples:

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

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

`apex:log:tail`

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

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: string

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

Help for `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
```

`apex:test:report`

Displays the test results for a specific test run.

Command Syntax

`sfdx force:apex:test:report`
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-i TESTRUNID
[-c]
[-d OUTPUTDIR]
[-w WAIT]
[--verbose]
[-r RESULTFORMAT]

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: string

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 | --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

--verbose
Optional

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

Type: boolean

-r | --resultformat RESULTFORMAT
Optional

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

Type: string

Permissible values are: human, tap, junit, json

Default value: human

Help for `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
```

`apex:test:run`

Runs Apex tests.

Command Syntax

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

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: string

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 | --classnames CLASSNAMES
Optional

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

Type: array

-s | --suitenames SUITENAMES
Optional

Comma-separated list of Apex test suite names to run. You can only specify one of classnames, suitenames, 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, suitenames, 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: string

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

-r | --resultformat RESULTFORMAT
Optional

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

Type: string

Permissible values are: human, tap, junit, json

Help for `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
```

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.

Command Syntax

```
sfdx force:apex:trigger:create
   [--json]
   [--loglevel LOGLEVEL]
   -n TRIGGERNAME
   [-t TEMPLATE]
   [-d OUTPUTDIR]
   [-a APIVERSION]
   [-s SOBJECT]
   [-e TRIGGEREVENTS]
```

Parameters

--json
Optional

Formats output as JSON.

Type: boolean

--loglevel LOGLEVEL
Optional

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

Type: string

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

-a | --apiversion APIVERSION
Optional

The API version of the created source.

Type: string

Permissible values are: 46.0, 45.0

Default value: 46.0

-s | --sobject SOBJECT
Optional

The sObject to create an Apex trigger on.

Type: string

Default value: SOBJECT

-e | --triggerevents TRIGGEREVENTS
Optional

The events that cause the trigger to fire.

Type: array

Default value: before insert

Help for `apex:trigger:create`

`auth` Commands

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

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.

auth:jwt:grant
Authorizes a Salesforce org using the JWT flow.

auth:list
list auth connection information

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

auth:sfdxurl:store
Authorizes a Salesforce org using an SFDX auth URL.

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

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.

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: string

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

Help for `auth:device:login`

`auth:jwt:grant`

Authorizes a Salesforce org using the JWT flow.

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: string

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

Help for `auth:jwt:grant`

Authorizes a Salesforce org using a private key file 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 <yourdomain>.my.salesforce.com (not <yourdomain>.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
```

`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: string

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

Default value: warn

Help for `auth:list`

`auth:logout`

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

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: string

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

Help for `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
```

`auth:sfdxurl:store`

Authorizes a Salesforce org using an SFDX auth URL.

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: string

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

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.

Examples:

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

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

auth:web:login

Authorizes a Salesforce org by opening a browser so you can log in through 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: string

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

Help for `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
```

`config` Commands

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`

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.

Command Syntax

sfdx force: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: string

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

Help for `config:get`

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:

```
$ sfdx force:config:get defaultusername
```

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

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

`config:list`

Lists the configuration variables for the Salesforce CLI.

Command Syntax

`sfdx force: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: string

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

Default value: warn

Help for `config:list`

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

`config:set`

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

Command Syntax

sfdx force: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: string

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

Help for `config:set`

Sets the configuration variables that the Salesforce CLI uses for various commands and tasks. Local variables apply only to your current project. Global variables apply in any directory.

Examples:

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

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

`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.

data:bulk:delete
Deletes a batch of records listed in a CSV file.

data:bulk:status
Polls the Bulk API for job status or batch status.

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

data:record:create
Creates and inserts a record.

data:record:delete
Deletes a single record.

data:record:get
Displays a single record.

data:record:update
Updates a single record.

data:soql:query
Executes a SOQL query.

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

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

`data:bulk:delete`

Deletes a batch of records listed in a CSV file.

Command Syntax

sfdx force:data:bulk:delete
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-s SOBJECTTYPE
-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: string

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 | --sobjecttype SOBJECTTYPE
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

Help for `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
```

`data:bulk:status`

Polls the Bulk API for job status or batch status.

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: string

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

Help for `data:bulk:status`

Examples:

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

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

`data:bulk:upsert`

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

Command Syntax

sfdx force:data:bulk:upsert
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-s SOBJECTTYPE
-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: string

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 | --sobjecttype SOBJECTTYPE
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

Help for `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
```

data:record:create

Creates and inserts a record.

Command Syntax

sfdx force:data:record:create
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-s SOBJECTTYPE
-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: string

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 | --sobjecttype SOBJECTTYPE
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

Help for `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=7dl170000008U36AAE
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
```

`data:record:delete`

Deletes a single record.

Command Syntax

```
sfdx force:data:record:delete
   [--json]
   [--loglevel LOGLEVEL]
   [-u TARGETUSERNAME]
   [--apiversion APIVERSION]
   -s SOBJECTTYPE
   [-i SOBJECTID]
   [-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: string

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 | --sobjecttype SOBJECTTYPE
Required

The sObject type of the record you're deleting.

Type: string

-i | --sobjectid SOBJECTID
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

Help for `data:record:delete`

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:delete -s Account -i 001D000000Kv3dl
```

```
   $ 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
```

`data:record:get`

Displays a single record.

Command Syntax

sfdx force:data:record:get
[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

```
-s SOBJECTTYPE
[-i SOBJECTID]
[-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: string

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 | --sobjecttype SOBJECTTYPE
Required

The sObject type of the record you're retrieving.

Type: string

-i | --sobjectid SOBJECTID
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

Help for `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 001D000000Kv3dl
```

```
   $ 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
```

`data:record:update`

Updates a single record.

Command Syntax

`sfdx force:data:record:update`

```
   [--json]
   [--loglevel LOGLEVEL]
   [-u TARGETUSERNAME]
   [--apiversion APIVERSION]
   -s SOBJECTTYPE
   [-i SOBJECTID]
   [-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: string

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 | --sobjecttype SOBJECTTYPE
Required

The sObject type of the record you're updating.

Type: string

-i | --sobjectid SOBJECTID
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

Help for `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
```

`data:soql:query`

Executes a SOQL query.

Command Syntax

`sfdx force:data:soql:query`
[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

-q QUERY

[-t]

[-r RESULTFORMAT]

[--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: string

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

SOQL query to execute.

Type: string

-t | --usetoolingapi
Optional

Execute the query using Tooling API.

Type: boolean

-r | --resultformat RESULTFORMAT
Optional

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

Type: string

Permissible values are: human, csv, json

Default value: human

--perflog
Optional

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

Type: boolean

Help for `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
```

data:tree:export

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

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: string

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

Help for `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_sobject_tree.htm

`data:tree:import`

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

Command Syntax

sfdx force:data:tree:import
[--json]

```
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
[-f SOBJECTTREEFILES]
[-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: string

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 | --sobjecttreefiles SOBJECTTREEFILES
Optional

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

Type: array

-p | --plan PLAN
Optional

Path to plan to insert multiple data files that have master-detail relationships. Either --sobjecttreefiles 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

Help for `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 SObject 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_sobject_tree.htm

`doc` Commands

Use the doc commands to display descriptions and help for commands in the force namespace.

`doc:commands:display`

Displays help for commands in the force namespace.

Command Syntax

`sfdx force:doc:commands:display`
[--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: string

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

Default value: warn

Help for `doc:commands:display`

Displays --help output for commands in the force namespace.

To display more details about the commands' parameters, include the --json flag.

`doc:commands:list`

Lists the commands in the force namespace.

Command Syntax

sfdx force:doc:commands:list

[--json]

[--loglevel LOGLEVEL]

[-u]

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: string

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

Default value: warn

-u | --usage

Optional

Lists docopt usage strings instead of command names and descriptions.

Type: boolean

Help for `doc:commands:list`

Displays a list of commands in the force namespace and their descriptions.

`lightning` Commands

Use the lightning commands to create Aura components and Lightning web components and to test Aura 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.

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.

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.

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.

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.

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.

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.

lightning:test:install

Installs a Lightning Testing Service (LTS) unmanaged package into your org.

lightning:test:run

Runs Aura component tests. The Lightning Testing Service (LTS) unmanaged package must be installed in your org. For details, see the LTS documentation.

`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.

Command Syntax

```
sfdx force:lightning:app:create
   [--json]
   [--loglevel LOGLEVEL]
```

```
-n APPNAME
[-t TEMPLATE]
[-d OUTPUTDIR]
[-a APIVERSION]
```

Parameters

--json
Optional

Formats output as JSON.

Type: boolean

--loglevel LOGLEVEL
Optional

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

Type: string

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

-a | --apiversion APIVERSION
Optional

The API version of the created source.

Type: string

Permissible values are: 46.0, 45.0

Default value: 46.0

Help for `lightning:app:create`

`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.

Command Syntax

```
sfdx force:lightning:component:create
   [--json]
   [--loglevel LOGLEVEL]
   -n COMPONENTNAME
   [-t TEMPLATE]
   [-d OUTPUTDIR]
   [-a APIVERSION]
   [--type TYPE]
```

Parameters

`--json`
Optional

Formats output as JSON.

Type: boolean

`--loglevel LOGLEVEL`
Optional

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

Type: string

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

Permissible values are: DefaultLightningCmp

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

-a | --apiversion APIVERSION
Optional

The API version of the created source.

Type: string

Permissible values are: 46.0, 45.0

Default value: 46.0

--type TYPE
Optional

The type of the new Lightning component.

Type: string

Permissible values are: aura, lwc

Default value: aura

Help for `lightning:component:create`

`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.

Command Syntax

sfdx force:lightning:event:create
[--json]

[--loglevel LOGLEVEL]

-n EVENTNAME

[-t TEMPLATE]

[-d OUTPUTDIR]

[-a APIVERSION]

Parameters

--json
Optional

Formats output as JSON.

Type: boolean

--loglevel LOGLEVEL
Optional

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

Type: string

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

-a | --apiversion APIVERSION
Optional

The API version of the created source.

Type: string

Permissible values are: 46.0, 45.0

Default value: 46.0

Help for `lightning:event:create`

`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.

Command Syntax

sfdx force:lightning:interface:create
[--json]

[--loglevel LOGLEVEL]

-n INTERFACENAME

```
[-t TEMPLATE]
[-d OUTPUTDIR]
[-a APIVERSION]
```

Parameters

--json
Optional

Formats output as JSON.

Type: boolean

--loglevel LOGLEVEL
Optional

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

Type: string

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

-a | --apiversion APIVERSION
Optional

The API version of the created source.

Type: string

Permissible values are: 46.0, 45.0

Default value: 46.0

Help for `lightning:interface:create`

`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.

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: string

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

Help for `lightning:lint`

`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.

Command Syntax

sfdx force:lightning:test:create
[--json]

[--loglevel LOGLEVEL]

-n TESTNAME

[-t TEMPLATE]

[-d OUTPUTDIR]

Parameters

--json
Optional

Formats output as JSON.

Type: boolean

--loglevel LOGLEVEL
Optional

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

Type: string

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

Help for `lightning:test:create`

`lightning:test:install`

Installs a Lightning Testing Service (LTS) unmanaged package into your org.

Command Syntax

sfdx force:lightning:test:install
[--json]

[--loglevel LOGLEVEL]

[-u TARGETUSERNAME]

[--apiversion APIVERSION]

[-w WAIT]

[-r RELEASEVERSION]

[-t PACKAGETYPE]

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: string

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.

Type: minutes

Default value: 2

-r | --releaseversion RELEASEVERSION
Optional

The release version of LTS unmanaged package you want to install.

Type: string

Default value: latest

-t | --packagetype PACKAGETYPE
Optional

Type of LTS unmanaged package to install. 'jasmine' and 'mocha' contains the essentials for development, 'full' contains both, and adds sample components and tests in the package. 'full' is best for 'kicking the tires' of LTS.

Type: string

Permissible values are: jasmine, mocha, full

Default value: full

Help for `lightning:test:install`

Examples:

```
$ sfdx force:lightning:test:install
```

```
$ sfdx force:lightning:test:install -w 0 -r v1.0
```

```
$ sfdx force:lightning:test:install -t jasmine
```

`lightning:test:run`

Runs Aura component tests. The Lightning Testing Service (LTS) unmanaged package must be installed in your org. For details, see the LTS documentation.

Command Syntax

```
sfdx force:lightning:test:run
   [--json]
   [--loglevel LOGLEVEL]
   [-u TARGETUSERNAME]
   [--apiversion APIVERSION]
   [-a APPNAME]
   [-d OUTPUTDIR]
   [-f CONFIGFILE]
   [-o]
   [-t TIMEOUT]
   [-r RESULTFORMAT]
```

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: string

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 | --appname APPNAME
Optional

Name of your Lightning test application. The name is case insensitive, and ".app" is optional, so "Test" and "test.app" are equivalent.

Default value: Test.app

Type: string

-d | --outputdir OUTPUTDIR
Optional

Directory path to store test run artifacts: log files, test results, and so on.

Type: directory

-f | --configfile CONFIGFILE
Optional

Path to a test configuration file to configure WebDriver and other settings. For details, see the LTS documentation.

Type: filepath

-o | --leavebrowseropen
Optional

Leaves browser open after the test finishes so that you can view the test suite results.

Type: boolean

-t | --timeout TIMEOUT
Optional

Time, in milliseconds, to wait for the results element to be present in the DOM, before failing and moving on to the next test.

Type: number

Default value: 60000

-r | --resultformat RESULTFORMAT
Optional

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

Type: string

Permissible values are: human, tap, junit, json

Default value: human

Help for `lightning:test:run`

Examples:

```
$ sfdx force:lightning:test:run
```

```
$ sfdx force:lightning:test:run -a tests -r human
```

```
$ sfdx force:lightning:test:run -f config/myConfigFile.json -d testResultFolder
```

`limits` Commands

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

limits:api:display
Displays remaining and maximum calls and events for your org.

`limits:api:display`

Displays remaining and maximum calls and events for your 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: string

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

Help for `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
```

`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.

`mdapi:convert`

Converts metadata retrieved via Metadata API into the source format used in Salesforce DX projects.

Command Syntax

```
sfdx force:mdapi:convert
  [--json]
  [--loglevel LOGLEVEL]
  -r ROOTDIR
  [-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: string

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

Help for `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
```

`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.

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]
```

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: string

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: string

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

Help for `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 ...".

mdapi:deploy:cancel

Cancels an asynchronous metadata deployment.

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: string

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

-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

Help for `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
```

`mdapi:deploy:report`

Checks the current status of an asynchronous metadata deployment.

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: string

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

Help for `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.

`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.

Command Syntax

```
sfdx force:mdapi:describemetadata
   [--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: string

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 (46.0).

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

Help for `mdapi:describemetadata`

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:describemetadata -a 43.0
```

```
$ sfdx force:mdapi:describemetadata -u me@example.com
```

```
$ sfdx force:mdapi:describemetadata -f /path/to/outputfilename.txt
```

```
$ sfdx force:mdapi:describemetadata -u me@example.com -f /path/to/outputfilename.txt
```

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.

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: string

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 (46.0).

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

Help for `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
```

`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.

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: string

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 | --unpackaged 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

Specifies whether only a single package is being retrieved (true) or more than one package (false).

Type: boolean

Help for `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.

`mdapi:retrieve:report`

Check the status of an asynchronous metadata retrieval.

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: string

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 | --retrievetargetdir 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 (asyncId) 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 --retrievetargetdir. Use with --wait to resume waiting.

Type: id

Help for `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.

`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.

`org:clone` (Beta)

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.

Note: This release contains a beta version of the `org:clone` command, which means it's a high-quality feature with known limitations. The the `org:clone` 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 `org:clone` command in the Salesforce DX group in the Trailblazer Community.

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: string

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: string

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

Help for `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
```

`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. This release contains a beta version of the sandbox CLI operations, which means it's a high-quality feature with known limitations.

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: string

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. The creation of sandbox orgs is available as a beta release.

Type: string

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

-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

Help for `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
```

`org:delete`

Marks a scratch or sandbox org for deletion.

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: string

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

Help for `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
```

`org:display`

Gets the description for the current or target org.

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: string

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

Help for `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
```

`org:list`

Lists all orgs that the Salesforce CLI has created or authenticated to.

Command Syntax

sfdx force:org:list
[--json]
[--loglevel LOGLEVEL]
[--verbose]
[--all]
[--clean]
[-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: string

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

Help for `org:list`

Examples:

```
$ sfdx force:org:list
```

```
$ sfdx force:org:list --verbose --json
```

```
$ sfdx force:org:list --verbose --json > tmp/MyOrgList.json
```

`org:open`

Opens an org in your browser.

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: string

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

Help for `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
```

`org:shape:create` (Pilot)

Creates a snapshot of org edition, features, and licenses to use for scratch org creation, allowing your scratch org to look like another org for testing.

Note: We provide the `org:shape: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 `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 `org:shape:create` command in the W19 Pilot: Org Shape for Scratch Orgs group in the Trailblazer Community.

Command Syntax

sfdx force:org:shape:create
[--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: string

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

Help for `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
```

`org:shape:delete` (Pilot)

Deletes all org shapes that you've created for an org using the Salesforce CLI.

Note: We provide the `org:shape: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 `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 `org:shape:delete` command in the W19 Pilot: Org Shape for Scratch Orgs group in the Trailblazer Community.

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: string

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

Help for `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
```

`org:shape:list` (Pilot)

Lists all org shapes that you've created using the Salesforce CLI.

Note: We provide the `org:shape: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 `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 `org:shape:list` command in the W19 Pilot: Org Shape for Scratch Orgs group in the Trailblazer Community.

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: string
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

Help for `org:shape:list`

Examples:

```
$ sfdx force:org:shape:list
```

```
$ sfdx force:org:shape:list --json
```

```
$ sfdx force:org:shape:list --json > tmp/MyOrgShapeList.json
```

`org:snapshot:create` (Pilot)

Creates a snapshot of a scratch org.

Note: We provide the `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 `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 `org:snapshot:create` command in the W19 Pilot: Scratch Org Snapshots group in the Trailblazer Community.

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: string

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

Help for `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 00Dxx0000000000 --snapshotname Dependencies
 --description "Contains PackageA v1.1.0"
```

```
  $ sfdx force:org:snapshot:create -o myuser@myorg -n NightlyBranch -d "Contains PkgA
v2.1.0 and PkgB 3.3.0"
```

`org:snapshot:delete` (Pilot)

Deletes a scratch org snapshot.

Note: We provide the `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 `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 `org:snapshot:delete` command in the W19 Pilot: Scratch Org Snapshots group in the Trailblazer Community.

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: string

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 0Oo) of the snapshot to delete.

Type: string

Help for `org:snapshot:delete`

Examples:

```
$ sfdx force:org:snapshot:delete --snapshot 0Oo...
```

```
$ sfdx force:org:snapshot:delete -s BaseSnapshot
```

`org:snapshot:get` (Pilot)

Retrieves details about a scratch org snapshot.

Note: We provide the `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 `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 `org:snapshot:get` command in the W19 Pilot: Scratch Org Snapshots group in the Trailblazer Community.

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: string

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 0Oo) of the snapshot to retrieve.

Type: string

Help for `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 0Oo...
```

```
$ sfdx force:org:snapshot:get -s Dependencies
```

`org:snapshot:list` (Pilot)

Lists scratch org snapshots for your Dev Hub.

Note: We provide the `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 `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 `org:snapshot:list` command in the W19 Pilot: Scratch Org Snapshots group in the Trailblazer Community.

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: string

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

Help for `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
```

`org:status` (Beta)

Reports sandbox org creation status. If the org is ready, authenticates to the org.

Note: This release contains a beta version of the `org:status` command, which means it's a high-quality feature with known limitations. The the `org:status` 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 `org:status` command in the Salesforce DX group in the Trailblazer Community.

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: string

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

Help for `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
```

`package` Commands

Use the package commands to develop and install packages.

package:create
Creates a package.

package:hammertest:list (Pilot)
Lists the statuses of running and completed ISV Hammer tests.

package:hammertest:report (Pilot)
Returns the status of a running ISV Hammer test or the results of a completed hammer test.

package:hammertest:run (Pilot)
Runs ISV Hammer for the specified package version and subscriber orgs.

package:install
Installs a package in the target org.

package:install:report
Retrieves the status of a package installation request.

package:installed:list
Lists all packages installed in the target org.

package:list
Lists all packages in the Dev Hub org.

package:uninstall
Uninstalls a second-generation package from the target org. To uninstall a first-generation package, use the Salesforce user interface.

package:uninstall:report
Retrieves the status of a package uninstall request.

package:update
Updates details about a package. Does not create a package version.

package:version:create
Creates a package version in the Dev Hub org.

package:version:create:list
Lists all requests to create second-generation package versions in the Dev Hub org.

package:version:create:report
Retrieves details about a package version creation request in the Dev Hub org.

package:version:list
Lists all package versions in the Dev Hub org.

package:version:promote
Promotes a package version to released status.

package:version:report
Retrieves details about a package version in the Dev Hub org.

package:version:update
Updates a second-generation package version in the Dev Hub org.

`package:create`

Creates a package.

Command Syntax

```
sfdx force:package:create
   [--json]
   [--loglevel LOGLEVEL]
   [-v TARGETDEVHUBUSERNAME]
   [--apiversion APIVERSION]
```

```
-n NAME
-t PACKAGETYPE
[-d DESCRIPTION]
[-e]
-r PATH
```

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: string

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). Managed packages are currently in beta.

These options determine upgrade and editability rules.

Type: string

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

Help for `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.

`package:hammertest:list` (Pilot)

Lists the statuses of running and completed ISV Hammer tests.

Note: We provide the `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 `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 `package:hammertest:list` command in the ISV Hammer - Pilot group in the Trailblazer Community.

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: string

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

Help for `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
```

`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 `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 `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 `package:hammertest:report` command in the ISV Hammer - Pilot group in the Trailblazer Community.

Command Syntax

`sfdx force:package:hammertest: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: string

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 0TW).

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

Help for `package:hammertest:report`

Example:

```
$ sfdx force:package:hammertest:report -u you@yourpackagingorg.com -i 0TW...
```

`package:hammertest:run` (Pilot)

Runs ISV Hammer for the specified package version and subscriber orgs.

Note: We provide the `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 `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 `package:hammertest:run` command in the ISV Hammer - Pilot group in the Trailblazer Community.

Command Syntax

`sfdx force:package:hammertest:run`

```
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-i PACKAGEVERSIONIDS
[-s SUBSCRIBERORGS]
[-f SUBSCRIBERFILE]
[-d SCHEDULEDRUNDATETIME]
[-p]
[-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: string

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 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

Help for `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
```

`package:install`

Installs a package in the target org.

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: string

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

For unlocked packages only, specifies whether to compile all Apex in the org and package, or only the Apex in the package.

Type: string

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: string

Permissible values are: AllUsers, AdminsOnly

Default value: AllUsers

-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).

Type: string

Permissible values are: DeprecateOnly, Mixed, Delete

Default value: Mixed

Help for `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
```

package:install:report

Retrieves the status of a package installation request.

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: string

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

Help for `package:install:report`

Examples:

```
$ sfdx force:package:install:report -i 0Hf...
```

```
$ sfdx force:package:install:report -i 0Hf... -u me@example.com
```

`package:installed:list`

Lists all packages installed in the target org.

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: string

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

Help for `package:installed:list`

Examples:

```
$ sfdx force:package:installed:list
```

```
$ sfdx force:package:installed:list -u me@example.com
```

`package:list`

Lists all packages in the Dev Hub org.

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: string
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

Help for `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
```

`package:uninstall`

Uninstalls a second-generation package from the target org. To uninstall a first-generation package, use the Salesforce user interface.

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: string

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

Help for `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.

`package:uninstall:report`

Retrieves the status of a package uninstall request.

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: string

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

Help for `package:uninstall:report`

Examples:

```
$ sfdx force:package:uninstall:report -i 06y...
```

```
$ sfdx force:package:uninstall:report -i 06y... -u me@example.com
```

`package:update`

Updates details about a package. Does not create a package version.

Command Syntax

sfdx force:package:update
[--json]

[--loglevel LOGLEVEL]

[-v TARGETDEVHUBUSERNAME]

[--apiversion APIVERSION]

-p PACKAGE

[-n NAME]

[-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: string

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

Help for `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.

package:version:create

Creates a package version in the Dev Hub org.

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]

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: string

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

-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

Help for `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".

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
```

`package:version:create:list`

Lists all requests to create second-generation package versions in the Dev Hub org.

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: string

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: string

Permissible values are: Queued, InProgress, Success, Error

Help for `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
```

`package:version:create:report`

Retrieves details about a package version creation request in the Dev Hub org.

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: string

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

Help for `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".

package:version:list

Lists all package versions in the Dev Hub org.

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: string

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 0Ho 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

Help for `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
```

package:version:promote

Promotes a package version to released status.

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: string

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

Help for `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"
```

`package:version:report`

Retrieves details about a package version in the Dev Hub org.

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: string

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

Help for `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".

`package:version:update`

Updates a second-generation package version in the Dev Hub org.

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: string

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

Help for `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".

`package1` Commands

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

`package1:version:create`

Creates a first-generation package version in the release org.

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: string

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

Release notes URL.

Type: url

-p | --postinstallurl POSTINSTALLURL
Optional

Post install URL.

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

Help for `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.

`package1:version:create:get`

Retrieves the status of a package version creation request.

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: string

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

Help for `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
```

`package1:version:display`

Displays detailed information about an individual first-generation package version.

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: string

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

Help for `package1:version:display`

Display detailed information about an individual package version, including metadata package ID, name, the release state, and build number.

`package1:version:list`

Lists the versions for the specified package or all first-generation packages in the org.

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: string

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

Help for `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.

`project` Commands

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

`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.

Command Syntax

```
sfdx force:project:create
   [--json]
   [--loglevel LOGLEVEL]
   -n PROJECTNAME
   [-t TEMPLATE]
   [-d OUTPUTDIR]
   [-s NAMESPACE]
   [-p DEFAULTPACKAGEDIR]
   [-x]
```

Parameters

`--json`

Optional

Formats output as JSON.

Type: boolean

`--loglevel LOGLEVEL`

Optional

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

Type: string

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

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

-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 web components, Aura components, and static resources.

Type: boolean

Help for `project:create`

`project:upgrade`

Updates project configuration and definition files to the latest format.

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: string
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

Help for `project:upgrade`

Examples:

$ sfdx force:project:upgrade

$ sfdx force:project:upgrade -f

`schema` Commands

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

`schema:sobject:describe`

Displays the metadata for a standard or custom object.

Command Syntax

sfdx force:schema:sobject:describe
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-s SOBJECTTYPE
[-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: string

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 | --sobjecttype SOBJECTTYPE
Required

The API name of the object to describe.

Type: string

-t | --usetoolingapi
Optional

Execute using Tooling API.

Type: boolean

Help for `schema:sobject:describe`

Examples:

```
$ sfdx force:schema:sobject:describe -s Account
```

```
$ sfdx force:schema:sobject:describe -s MyObject__c
```

```
$ sfdx force:schema:sobject:describe -s ApexClass -t
```

`schema:sobject:list`

Lists all objects of a specified sObject category.

Command Syntax

`sfdx force:schema:sobject:list`
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
-c SOBJECTTYPECATEGORY

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: string

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 | --sobjecttypecategory SOBJECTTYPECATEGORY
Required

The type of objects to list: all, custom, or standard.

Type: string

Help for `schema:sobject:list`

Lists all objects, custom objects, or standard objects in the org.

Examples:

```
$ sfdx force:schema:sobject:list -c all
```

```
$ sfdx force:schema:sobject:list -c custom
```

```
$ sfdx force:schema:sobject:list -c standard
```

`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.

source:convert
Converts source-formatted files into metadata that you can deploy using Metadata API.

source:delete
Deletes source files from your project and from a non-source-tracked org, such as a sandbox.

source:deploy
Deploys metadata in source format to an org.

source:deploy:cancel
Cancels an asynchronous source deployment.

source:deploy:report
Checks the current status of an asynchronous metadata deployment.

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.

source:pull
Pulls changed source from the scratch org to your project to keep them in sync.

source:push
Pushes changed source from your project to a scratch org to keep them in sync.

source:retrieve
Retrieves metadata in source format from an org to your local Salesforce DX project.

source:status
Lists changes that have been made locally, in a scratch org, or both.

`source:convert`

Converts source-formatted files into metadata that you can deploy using Metadata API.

Command Syntax

`sfdx force:source:convert`
[`--json`]
[`--loglevel LOGLEVEL`]
[`-r ROOTDIR`]
[`-d OUTPUTDIR`]
[`-n PACKAGENAME`]

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: string

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

Help for `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'
```

source:delete

Deletes source files from your project and from a non-source-tracked org, such as a sandbox.

Command Syntax

sfdx force:source:delete
[--json]
[--loglevel LOGLEVEL]
[-u TARGETUSERNAME]
[--apiversion APIVERSION]
[-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: string

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 | --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

-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

Help for `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>
```

`source:deploy`

Deploys metadata in source format to an org.

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: string

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

-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: string

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.

If you specify this parameter, don't specify --metadata or --sourcepath.

Type: filepath

Help for `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.

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
```

source:deploy:cancel

Cancels an asynchronous source deployment.

Command Syntax

sfdx force:source: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: string

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

-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

Help for `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
```

`source:deploy:report`

Checks the current status of an asynchronous metadata deployment.

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: string

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

-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

Help for `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
```

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.

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: string

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

Help for `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
```

`source:pull`

Pulls changed source from the scratch org to your project to keep them in sync.

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: string

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

-f | --forceoverwrite
Optional

Runs the pull command even if conflicts exist. Changes in the scratch org overwrite changes in the project.

Type: boolean

Help for `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.

`source:push`

Pushes changed source from your project to a scratch org to keep them in sync.

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: string

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

Help for `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.

source:retrieve

Retrieves metadata in source format from an org to your local Salesforce DX project.

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: string

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

-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

Help for `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.

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
```

source:status

Lists changes that have been made locally, in a scratch org, or both.

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: string

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

Help for `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
```

`user` Commands

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

`user:create`

Creates a user for a scratch org.

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: string

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

Help for `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
```

user:display

Displays information about a user of a scratch org that the Salesforce CLI has created or authenticated.

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: string
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

Help for `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
```

`user:list`

Lists all users of a scratch org that the Salesforce CLI has created or authenticated.

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: string

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

Help for `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
```

`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.

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: string

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

Help for `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
```

`user:permset:assign`

Assigns a named permission set to one or more users of an org.

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: string

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

Help for `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
```

`visualforce` Commands

Use the visualforce commands to create Visualforce pages and components.

`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.

Command Syntax

`sfdx force:visualforce:component:create`

```
[--json]
[--loglevel LOGLEVEL]
[-t TEMPLATE]
[-d OUTPUTDIR]
-n COMPONENTNAME
[-a APIVERSION]
-l LABEL
```

Parameters

`--json`

Optional

Formats output as JSON.

Type: boolean

`--loglevel LOGLEVEL`

Optional

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

Type: string

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

-n | --componentname COMPONENTNAME
Required

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

Type: string

-a | --apiversion APIVERSION
Optional

The API version of the created source.

Type: string

Permissible values are: 46.0, 45.0

Default value: 46.0

-l | --label LABEL
Required

The label saved in the metadata for the Visualforce component.

Type: string

Help for `visualforce:component:create`

`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.

Command Syntax

```
sfdx force:visualforce:page:create
   [--json]
   [--loglevel LOGLEVEL]
   [-t TEMPLATE]
```

```
[-d OUTPUTDIR]
-n PAGENAME
[-a APIVERSION]
-l LABEL
```

Parameters

--json
Optional

Formats output as JSON.

Type: boolean

--loglevel LOGLEVEL
Optional

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

Type: string

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: DefaultVFPage

Default value: DefaultVFPage

-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

-n | --pagename PAGENAME
Required

The Visualforce page name. The name can be up to 40 characters and must start with a letter.

Type: string

-a | --apiversion APIVERSION
Optional

The API version of the created source.

Type: string

Permissible values are: 46.0, 45.0

Default value: 46.0

-l | --label LABEL
Required

The label saved in the metadata for the Visualforce page.

Type: string

Help for `visualforce:page:create`

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.