Option

August 25, 2026

Option

Kind: Class

Source: `lib/option.js`

`Option` defines a command-line option, including its flags, default value, environment variable mapping, validation rules, and parsing behavior. It is added to a command and supplies metadata used while parsing arguments and building help output.

Methods

Method	Signature	Returns
`default`	`default(value: undefined, description: undefined)`	`void`
`preset`	`preset(arg: undefined)`	`void`
`conflicts`	`conflicts(names: undefined)`	`void`
`implies`	`implies(impliedOptionValues: undefined)`	`void`
`env`	`env(name: undefined)`	`void`
`argParser`	`argParser(fn: undefined)`	`void`
`makeOptionMandatory`	`makeOptionMandatory(mandatory: undefined)`	`void`
`hideHelp`	`hideHelp(hide: undefined)`	`void`
`_collectValue`	`_collectValue(value: undefined, previous: undefined)`	`void`
`choices`	`choices(values: undefined)`	`void`
`name`	`name()`	`void`
`attributeName`	`attributeName()`	`void`
`helpGroup`	`helpGroup(heading: undefined)`	`void`
`is`	`is(arg: undefined)`	`void`

Method	Signature	Returns
`isBoolean`	`isBoolean()`	`void`

Where it refuses work

- `Option` stops the work with `InvalidArgumentError` when `!this.argChoices.includes(arg)`.
- `Option` stops the work with an early return when `previous === this.defaultValue || !Array.isArray(previous)`.
- `Option` stops the work with an early return when `this.variadic`.
- `Option` stops the work with an early return when `this.long`.
- `Option` stops the work with an early return when `this.negate`.

Diagram

```
graph LR
  Command[Command] -->|addOption| Option[Option]
  Environment[Environment variable] -->|env| Option
  Option -->|flags, defaults, rules| Parser[Argument parser]
  Parser -->|stores parsed value| CommandOptions[command.opts()]
```

Usage

```
const { Command, Option } = require('commander');

const program = new Command();

const formatOption = new Option(
  '-f, --format <type>',
  'select output format'
)
  .choices(['text', 'json'])
  .default('text')
  .env('OUTPUT_FORMAT');

const quietOption = new Option('-q, --quiet', 'suppress output')
  .conflicts('verbose');

program
  .addOption(formatOption)
  .addOption(quietOption);

program.parse();
```

```
const options = program.opts();
console.log(options.format);
```

AI Coding Instructions

- Create options with `new Option(flags, description)` before adding them to a `Command`.
- Use `choices()` for accepted argument values and `argParser()` when values need custom conversion.
- Use `conflicts()` and `implies()` to describe relationships between option names, not flag strings.
- Set `env()` when an option may read its value from an environment variable.
- Use `hideHelp()` only for options that should still parse but should not appear in generated help.

How it works

I'll inspect the option-token parsing branch to document the actual preconditions for required, optional, variadic, boolean, and negated flags.

Relationships

- IMPORTS → `InvalidArgumentError`

Used by

2 references from 2 files. Each is a place in this repository where the symbol is actually used — go read one rather than trusting an example.

Imported by (2)

- `program` — `index.js`:7
- `Command` — `lib/command.js`:14