dpy-syntaxer

Release 1.0.0

sevenc-nanashi

Jun 29, 2021

CONTENTS:

This library will help you with syntax analyzing.

CHAPTER

ONE

API REFERENCE

1.1 Syntax class

class discord.ext.syntaxer.**Syntax**(*command: discord.ext.commands.core.Command*, *description: Optional[str] = None*, ***, *default_format: str = '{prefix}{variable_prefix}{command_name}{variable_suffix}{suffix}'*, *kwarg_format: str = '{prefix}{variable_prefix}{command_name}{variable_suffix}'*, *required_formats: Tuple[str, str] = ('<', '>')*, *optional_formats: Tuple[str, str] = ('[', ']')*, *variable_formats: Tuple[str, str] = ('', '...')*)

Analyze command and make Syntax object.

This inherits *SpaceList*, so :func:str will return a formatted syntax.

Parameters

- **command** (*commands.Command*) – Command to analyze syntax.
- **description** (*str, optional*) – Description of the command. If None, use docstring.
- **default_format** (*str, optional*) – Format of regular argument, by default "{prefix}{variable_prefix}{command_name}{variable_suffix}{suffix}"
- **kwarg_format** (*str, optional*) – Format of keyword argument, by default "{prefix}{variable_prefix}{command_name}{variable_suffix}"
- **required_formats** (*tuple, optional*) – Prefix and suffix of required argument, by default ("<", ">")
- **optional_formats** (*tuple, optional*) – Prefix and suffix of optional argument, by default ("[", "]")
- **variable_formats** (*tuple, optional*) – Prefix and suffix of variable argument, by default ("", "...")

property **args**

Returns SpaceList with command arguments.

property **names**

Return SpaceList with command name and parents.

1.2 Syntax elements

class discord.ext.syntaxer.**CommandName**(*name: str*, *parent: Optional[discord.ext.syntaxer.main.ParentName] = None*)

Represents name of a command.

Parameters

- **name** (*str*) – Name of the command.
- **parent** (*ParentName*) – Parent of the command, None if the command doesn't have a parent.

class discord.ext.syntaxer.**CommandArgument**(*name: str*, *required: bool*, *description: str*, *formatted: str*, *flag: discord.ext.syntaxer.main.ArgumentType*, *default: Any*, *param: inspect.Parameter*)

Represents a command argument.

`str` returns formatted value.

Parameters

- **name** (*str*) – Name of the argument.
- **required** (*bool*) – The argument is required or optional.
- **description** (*str*) – Description of the argument.
- **formatted** (*str*) – Formatted value for the argument.
- **flag** (*int*) – Type flag of the argument.
- **default** (*Optional[Any]*) – Default value of the argument if any.
- **param** (*:class:inspect.Parameter*) – Parameter for the argument.

property **optional**

Same as `not self.required`.

1.3 Other classes

class discord.ext.syntaxer.**SpaceList**(*iterable=()*, */*)

A special list. When you pass this list to `str`, content will be converted with :func:str, and joins them with space.

CHAPTER

TWO

EXAMPLE

2.1 Simple

```python
import os
from textwrap import dedent

import discord
from discord.ext import commands, syntaxer

bot = commands.Bot(
    "b ",
    help_command=None,
    allowed_mentions=discord.AllowedMentions(everyone=False, replied_user=False)
)

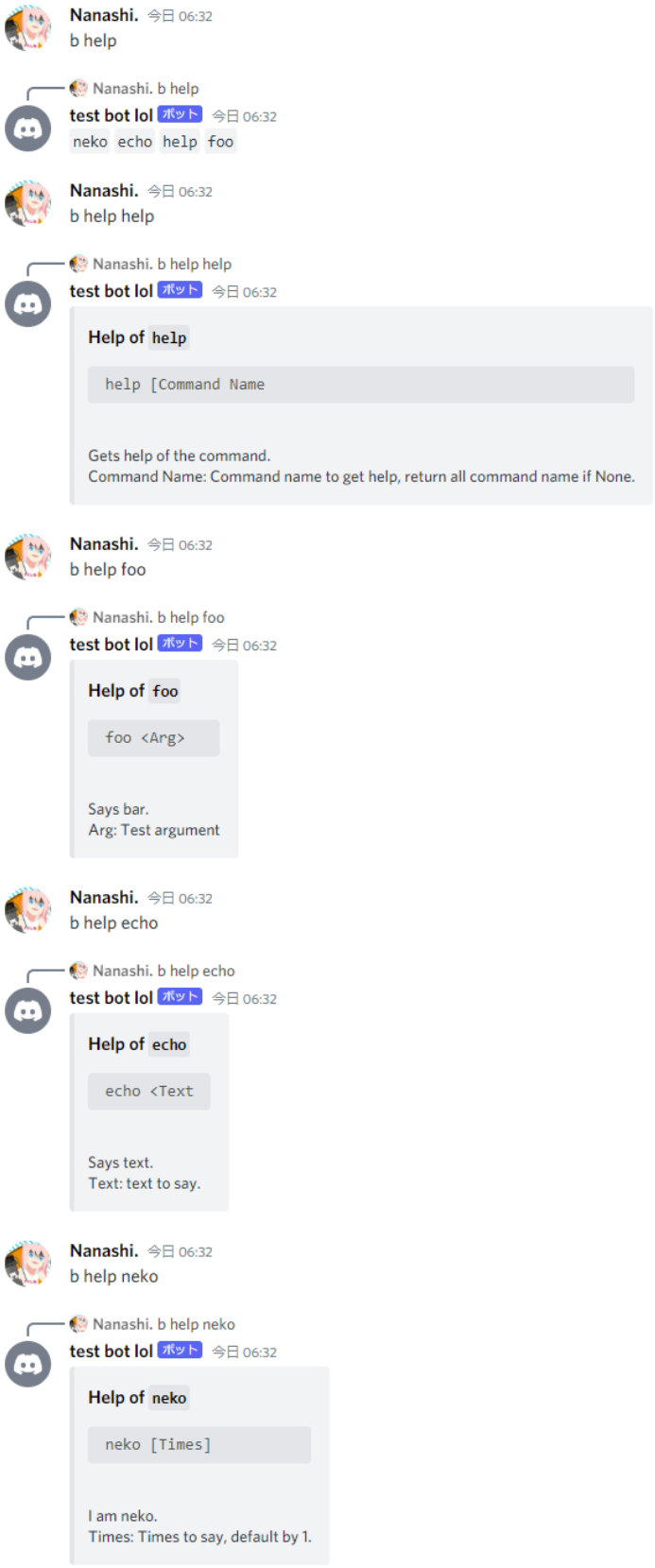

@bot.event
async def on_ready():
    print(f"Logged in as: {bot.user}")


@bot.command("help", aliases=["?"])
async def _help(ctx, *, command_name=None):
    """
    Gets help of the command.
    Command Name: Command name to get help, return all command name if None.
    """
    if command_name is None:
        return await ctx.reply(" ".join([f"`{c}`" for c in bot.commands]))
    command = bot.get_command(command_name)
    if command is None:
        return await ctx.reply("Unknown command.")
    syntax = syntaxer.Syntax(command)
    e = discord.Embed(
        title=f"Help of `{command}`",
        description=dedent(f"""
        ``` {syntax} ```
        {command.callback.__doc__}
        """))
    return await ctx.reply(embed=e)
```

(continues on next page)
```

(continued from previous page)

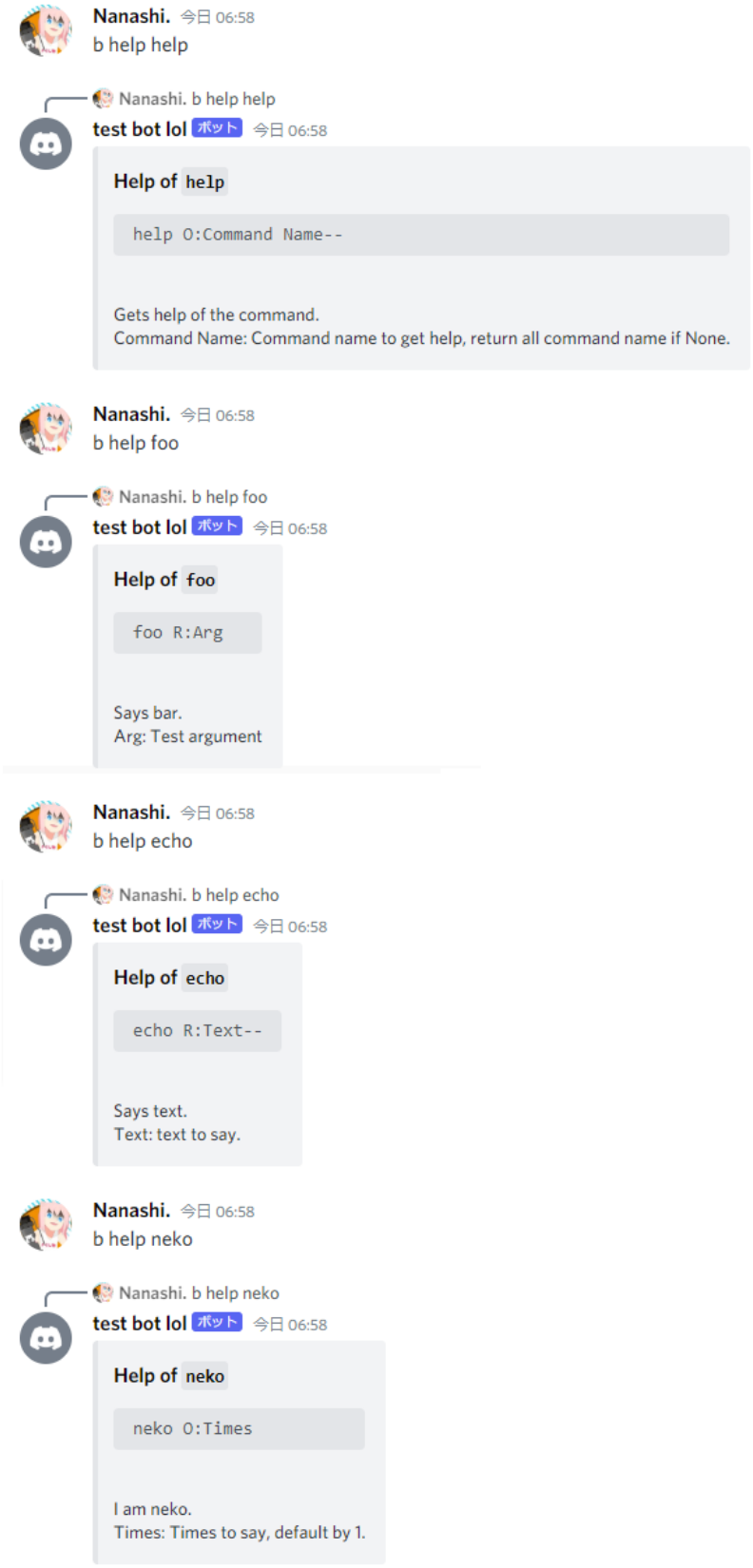
```python


@bot.command()
async def foo(ctx, argument):
    """
    Says bar.
    Arg: Test argument
    """
    await ctx.reply("bar, passed" + argument)


@bot.command()
async def echo(ctx, *, text):
    """
    Says text.
    Text: text to say.
    """
    await ctx.send(text)


@bot.command()
async def neko(ctx, count=1):
    """
    I am neko.
    Times: Times to say, default by 1.
    """
    for _ in range(count):
        await ctx.send("Nyan")

bot.run(os.getenv("token"))
```


Nanashi. 今日 06:32
b help
Nanashi. b help
test bot lol ボット 今日 06:32
neko echo help foo
Nanashi. 今日 06:32
b help help
Nanashi. b help help
test bot lol ボット 今日 06:32
Help of help
help [Command Name
Gets help of the command.
Command Name: Command name to get help, return all command name if None.
Nanashi. 今日 06:32
b help foo
Nanashi. b help foo
test bot lol ボット 今日 06:32
Help of foo
foo <Arg>
Says bar.
Arg: Test argument
Nanashi. 今日 06:32
b help echo
Nanashi. b help echo
test bot lol ボット 今日 06:32
Help of echo
echo <Text
Says text.
Text: text to say.
Nanashi. 今日 06:32
b help neko
Nanashi. b help neko
test bot lol ボット 今日 06:32
Help of neko
neko [Times]
I am neko.
Times: Times to say, default by 1.

2.2 Customize

Note: Code is as same as first, except help.

```python
@bot.command("help", aliases=["?"])
async def _help(ctx, *, command_name=None):
    """
    Gets help of the command.
    Command Name: Command name to get help, return all command name if None.
    """
    if command_name is None:
        return await ctx.reply(" ".join([f"`{c}`" for c in bot.commands]))
    command = bot.get_command(command_name)
    if command is None:
        return await ctx.reply("Unknown command.")
    syntax = syntaxer.Syntax(
        command,
        default_format="{prefix}:{command_name}{variable_suffix}",
        kwarg_format="{prefix}:{command_name}{variable_suffix}--",
        required_formats=("R", ""),
        optional_formats=("O", ""),
        variable_formats=("", "+")
    )
    e = discord.Embed(
        title=f"Help of `{command}`",
        description=dedent(f"""
        ``` {syntax} ```
        {command.callback.__doc__}
        """))
    return await ctx.reply(embed=e)
```
```


Nanashi. 今日 06:58
b help help
Nanashi. b help help
test bot lol ボット 今日 06:58
Help of help
help O:Command Name--
Gets help of the command.
Command Name: Command name to get help, return all command name if None.
Nanashi. 今日 06:58
b help foo
Nanashi. b help foo
test bot lol ボット 今日 06:58
Help of foo
foo R:Arg
Says bar.
Arg: Test argument
Nanashi. 今日 06:58
b help echo
Nanashi. b help echo
test bot lol ボット 今日 06:58
Help of echo
echo R:Text--
Says text.
Text: text to say.
Nanashi. 今日 06:58
b help neko
Nanashi. b help neko
test bot lol ボット 今日 06:58
Help of neko
neko O:Times
I am neko.
Times: Times to say, default by 1.

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search

INDEX