[Python 103: Discord Bot for Personal Functions]
Tuesday, February 18th | 12:30 PM - 1:30 PM | HUB Games Arena
Details:

Review of what we should have covered in tutorial #2:

Tasks & Loops, Errors & Administrative server functions

Tasks & loops:

Overview: Tasks are able to have loops that run scheduled based off of what time increment you set within it. You will be able to start and stop the command through different functions. For a post_time method, you these are the primary commands for controlling it: post_time.stop(), post_time.start().

Importing: You will need to import tasks from discord.ext

- Depending on what you want to send, you may want to consider import random for a random generator

Example: Reference the post_time method and the header.

Errors:

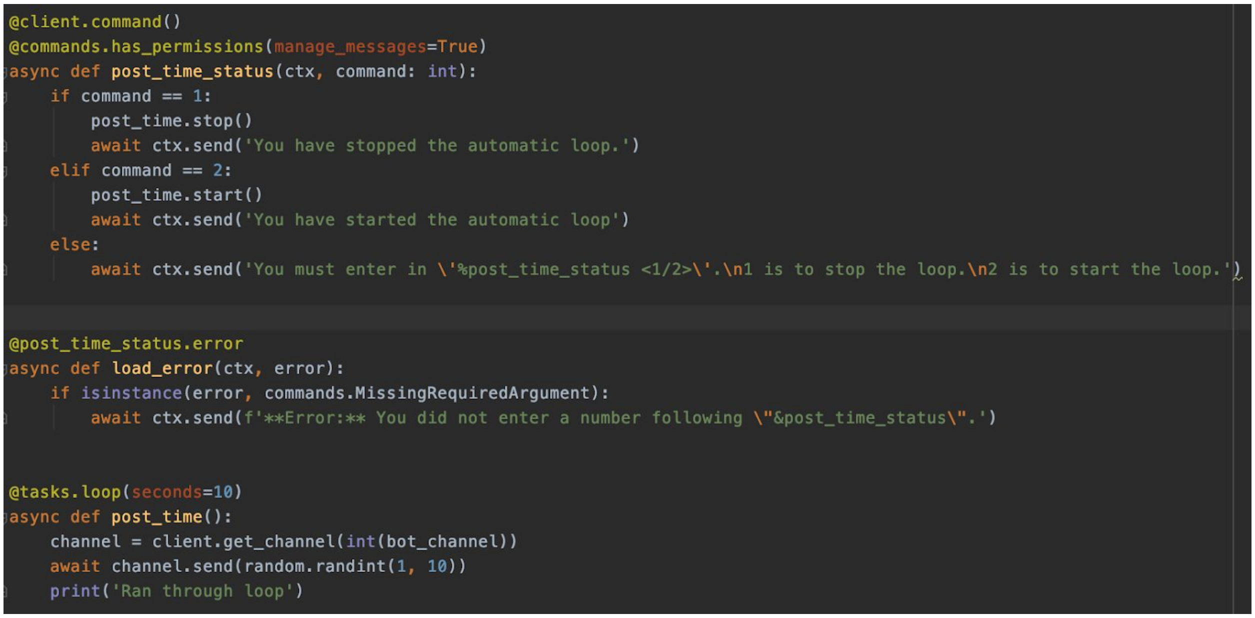
Overview: User errors such as inputting the wrong number of inputs, having the wrong input type, or any other errors can be used using *method_name*.error. This function will run only when an error has been given where you can then use this to prompt the user.

Example: Reference the load_error method below

Administrative Server Functions:

Overview: Administrative server functions focuses around having permissions. Every user has a different permission within discord and checking these will section off which functions are applicable to which user.

Example: Reference the post_time_status method header

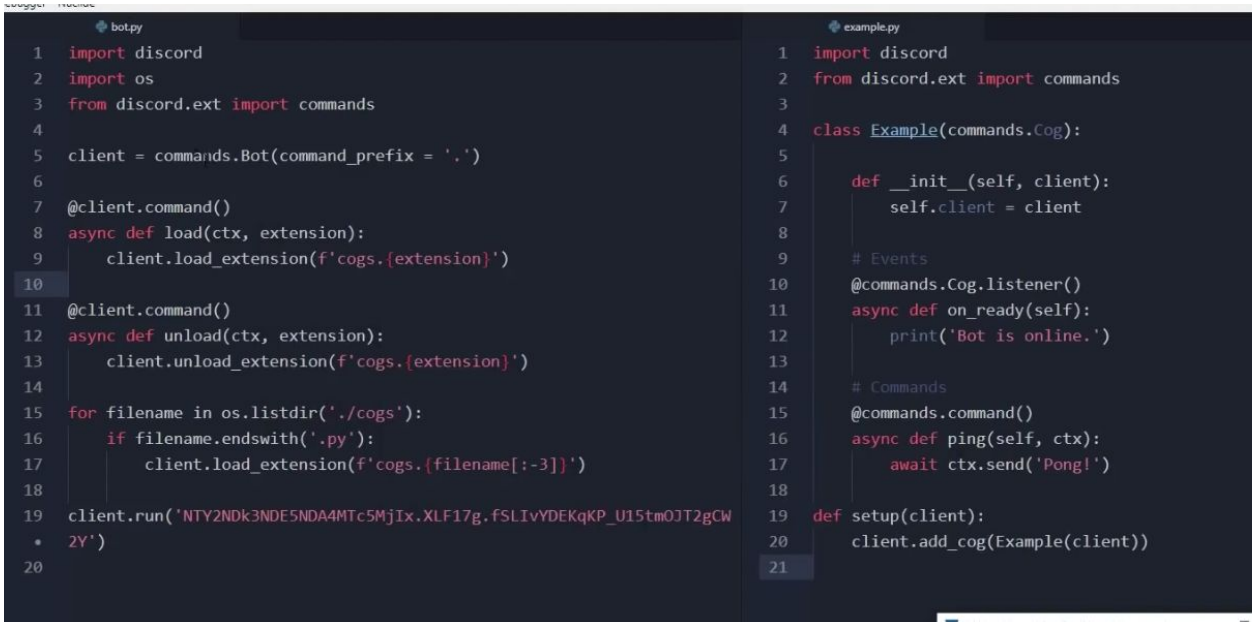


```
@client.command()
@commands.has_permissions(manage_messages=True)
async def post_time_status(ctx, command: int):
    if command == 1:
        post_time.stop()
        await ctx.send('You have stopped the automatic loop.')
    elif command == 2:
        post_time.start()
        await ctx.send('You have started the automatic loop')
    else:
        await ctx.send('You must enter in \'%post_time_status <1/2>\'.\n1 is to stop the loop.\n2 is to start the loop.')


@post_time_status.error
async def load_error(ctx, error):
    if isinstance(error, commands.MissingRequiredArgument):
        await ctx.send(f'**Error:** You did not enter a number following \"&post_time_status\".')


@tasks.loop(seconds=10)
async def post_time():
    channel = client.get_channel(int(bot_channel))
    await channel.send(random.randint(1, 10))
    print('Ran through loop')
```

Walkthrough - cogs:



```
import discord
import os
from discord.ext import commands

client = commands.Bot(command_prefix = '.')

@client.command()
async def load(ctx, extension):
    client.load_extension(f'cogs.{extension}')

@client.command()
async def unload(ctx, extension):
    client.unload_extension(f'cogs.{extension}')

for filename in os.listdir('./cogs'):
    if filename.endswith('.py'):
        client.load_extension(f'cogs.{filename[:-3]}')

client.run('NTY2NDk3NDE5NDA4MTc5MjIx.XLF17g.fSLIvYDEKqKP_U15tmOJT2gCW2Y')
```

```
import discord
from discord.ext import commands

class Example(commands.Cog):

    def __init__(self, client):
        self.client = client

    # Events
    @commands.Cog.listener()
    async def on_ready(self):
        print('Bot is online.')

    # Commands
    @commands.command()
    async def ping(self, ctx):
        await ctx.send('Pong!')

def setup(client):
    client.add_cog(Example(client))
```

Walk through of my PCM discord bot:
What are some skills used for 14X in this

- Using text files - things aren’t constant and saving progress needs to be somewhere
- String concatenation
- Classes and inheritance

What can you build next?

Next week - we will be going back a little bit and covering - what is Python and how it is such a big phenomenon in today’s world. It will be less coding based but more informational with a few just do simple things afterwards. Next quarter, there will be a 5 or 9-week one hour class on learning more about Python, based on your feedback now, we will gauage it from there.