ScratchConnect For Beginners

WojtekGame

Sid72020123 (for his words)

Chapters

Getting Started

Let’s talk about **ScratchConnect** itself.

As the authors says:

“Python Library to connect Scratch API and much more.

This library can show the statistics of Users, Projects, Studios, Forums and also connect and set cloud variables of a project!„

and,

“You need basic knowledge of Python. Using this library without the knowledge can be risky.„

What i can say, you need to know **Python** before reading this book.

And ScratchConnect is built by those APIs, so ScratchConnect can take data:

1. Scratch API by the Scratch Team
2. ScratchDB by DatOneLefty on Scratch
3. Scratch Comments API, Simple Forum API by Sid72020123 on Scratch
4. Ocular API by Jeffalo on Scratch
5. Aviate API by NFlex123 on Scratch

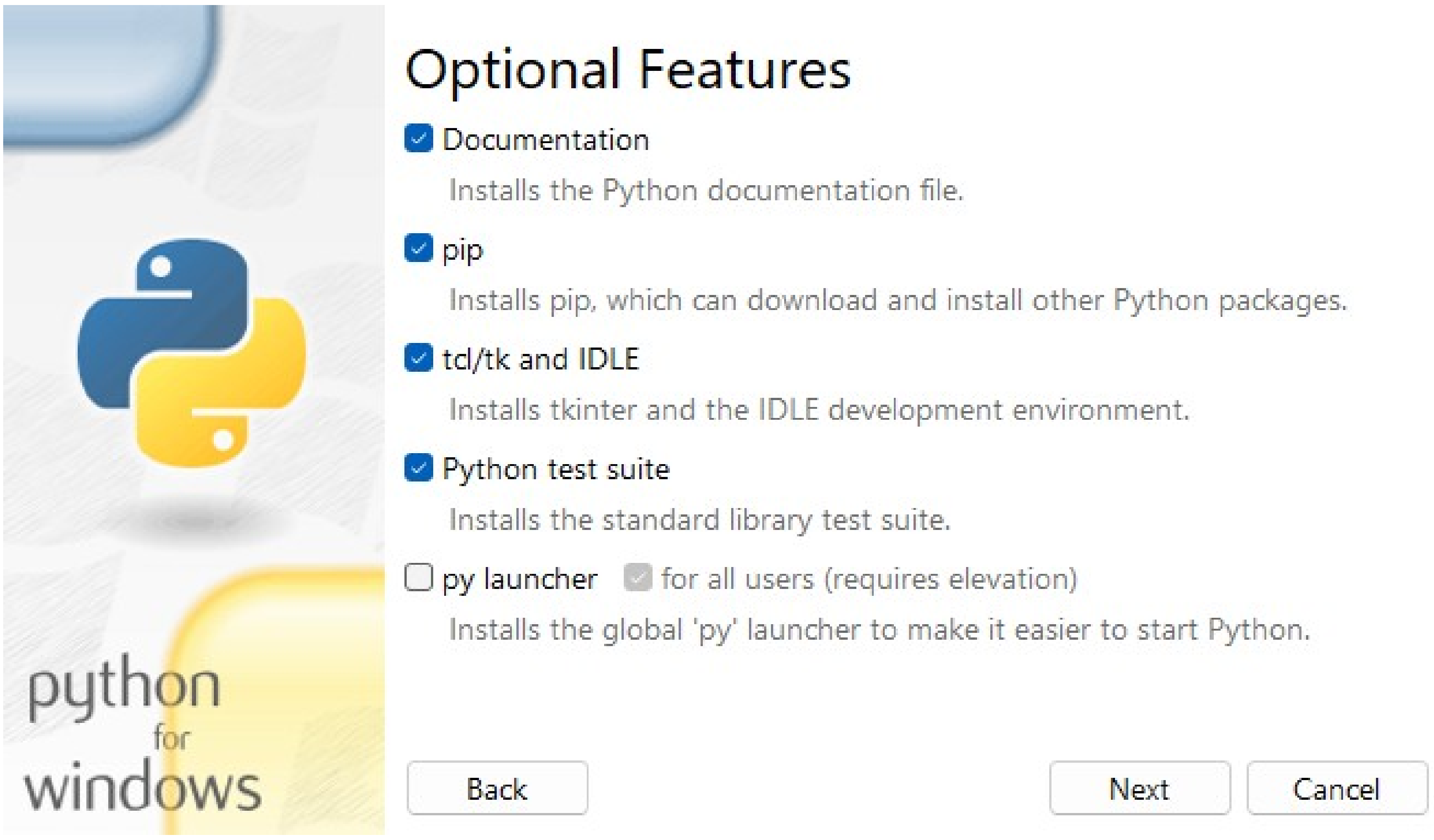
The owner (As of 2022) is Sid72020123

The **source code** is at: https://github.com/Sid72020123/scratchconnect

Setup

You need to download **Python** from https://python.org.

You need to run the installer, and have this screen



Optional Features

- [✓] Documentation
  Installs the Python documentation file.
- [✓] pip
  Installs pip, which can download and install other Python packages.
- [✓] tcl/tk and IDLE
  Installs tkinter and the IDLE development environment.
- [✓] Python test suite
  Installs the standard library test suite.
- [ ] py launcher [✓] for all users (requires elevation)
  Installs the global 'py' launcher to make it easier to start Python.

(checkboxes have to be the same)

To install **ScratchConnect** we need to have `pip` installed.

To install **ScratchConnect** with `pip`, you can use those Methods below.

Method 1, Run this command on your terminal

```
pip install scratchconnect
```

Method 2, Run this Python code on IDLE editor (F5)

```
import os
os.system(„pip install
scratchconnect”)
```
.

If there are problems with , use https://packaging.python.org/en/latest/tutorials/installing-packages/.

Making an connection

Before making an connection, we need to import it on IDLE:
`import scratchconnect`.

Now, we can use the library, let’s connect our Scratch account to **ScratchConnect:**

```
user =
scratchconnect.ScratchConnect("
Username", "Password")
```
.
Please use enviroment variables as the password

If you don’t have an Scratch account, you need to create one on https://scratch.mit.edu.

To have more control with your account here are more uses:

```
user =
scratchconnect.ScratchConnect("
Username", "Password")
user.id()
user.thumbnail_url()
user.messages_count()
user.messages(all=False,
limit=20, offset=0,
filter="all")
user.clear_messages()
user.my_stuff_projects(order="a
ll", page=1, sort_by="")
user.work()
user.bio()
user.status()
user.joined_date()
user.country()
user.featured_data()
user.projects()
user.followers_count()
user.following_count()
user.total_views()
user.total_loves_count()
user.total_favourites_count()
user.following()
user.followers()
user.favourites()
user.toggle_commenting()
user.follow_user(username="Sid7
2020123")
user.unfollow_user(username="Si
d72020123")
user.set_bio(content="Hi!")
user.set_work(content="Hi!")
user.all_data()
user.site_health()
user.site_news()
user.site_front_page_projects()
user.explore_projects(mode="tre
nding", query="*")
user.explore_studios(mode="tren
ding", query="*")
user.search_projects(mode="tren
ding", search="*")
user.search_studios(mode="trend
ing", search="*")
```

```
user.set_featured_project(project_id="1", label='featured_project')
user.user_follower_history()
user.comments(limit=5, page=1)
user.ocular_data()
user.aviate_data(code=False)
user.search_forum(q="Hi!", order="relevance", page=0)
user.update_data().
```

And you can connect an user:

```
login = scratchconnect.ScratchConnect("Username", "Password")
```

```
user = login.connect_user(username="Sid72020123")
```

You can also use without login:

```
user = scratchconnect.ScratchConnect()
```

Connecting an Studio

To connect an Scratch Studio you can use this:

```
user =
scratchconnect.ScratchConnect("Username", "Password")
```

```
studio =
user.connect_studio(studio_id=1).
```

More actions you can do:

```
studio.title()
studio.host_id()
studio.description()
studio.visibility()
studio.is_public()
studio.is_open_to_all()
studio.are_comments_allowed()
studio.history()  # Returns the history of
the studio
studio.stats()
studio.thumbnail_url()
studio.add_project(project_id=1)
studio.remove_project(project_id=1)
studio.open_to_public()
studio.close_to_public()
studio.follow_studio()
studio.unfollow_studio()
studio.toggle_commenting()
studio.post_comment(content="Hi!")
studio.reply_comment(content="Hi!",
comment_id=1)
studio.delete_comment()
studio.report_comment(comment_id=1)
studio.invite_curator(username="Sid72020123")
studio.accept_curator()
studio.promote_curator(username="Sid72020123"
)
studio.set_description(content="Hi!")
studio.set_title(content="Hi!")
studio.projects(all=False, limit=40,
offset=0)
studio.comments(all=False, limit=40,
offset=0)
studio.curators(all=False, limit=40,
offset=0)
studio.managers(all=False, limit=40,
offset=0)
studio.activity(all=False, limit=40,
offset=0)
studio.all_data()
studio.update_data().
```

Connecting an Scratch Project

To connect an Scratch Project you can use this:

```
user = scratchconnect.ScratchConnect("Username", "Password")   project = user.connect_project(project_id=1)
```

More uses:

```
project.author()
```

```
project.title()
```

```
project.notes()
project.instruction()
project.stats()
project.history()
project.remix_data()
project.visibility()
project.is_public()
project.is_published()
project.thumbnail_url()
project.assets_info()
project.scripts()
project.love()
project.unlove()
project.favourite()
project.unfavourite()
project.comments(all=False, limit=40,
offset=0, comment_id=None)
project.remixes(all=False, limit=20,
offset=0)
project.post_comment(content="Hi!")
project.reply_comment(content="Hi!",
comment_id=1)
project.toggle_commenting()
project.turn_on_commenting()
project.turn_off_commenting()
project.report(category="", reason="")
project.unshare()
project.view()
project.set_thumbnail(file="")
project.delete_comment(comment_id=1)
project.report_comment(comment_id=1)
project.reply_comment(comment_id=1,
content="Hi!")
project.set_title()
project.set_description()
project.set_instruction()
project.all_data()
project.update_data()
```
.

You can also access an unshared project:

```
project = user.connect_project(project_id=1,
access_unshared=True)
```

Cloud Variables

To connect variables of an Scratch Project:

```
user =
scratchconnect.ScratchConnect("Username",
"Password")
```

```
project =
user.connect_project(project_id=1)
variables =
project.connect_cloud_variables()
```

Get data of an Cloud Variable:

```
variables.get_variable_data(limit=100,
offset=0)
```

Get value of an Cloud Variable:

```
variables.get_cloud_variable_value(variab
le_name="Name", limit=100)
```

Set value of an Cloud Variable:

```
variables.set_cloud_variable(variable_nam
e="Name", value=123)
```

You can also have evemts of Cloud Variables:

```
event = cloud.create_cloud_event()

@event.on("connect")

def connect():

    print("Connected Cloud!")

@event.on("set")
def set(data):
    print("SET: ", data)
@event.on("create")
def create(data):
    print("CREATE: ", data)
@event.on("delete")
def delete(data):
    print("DELETE: ", data)
@event.on("disconnect")
def disconnect():
    print("Disconnected from Cloud!")
event.start(update_time=1)
```

Turbowarp

You can do in **ScratchConnect** is you can connect from Turbowarp (`turbowarp.org`)

Cloud Variables:

You can connect Turbowarp cloud variables:

```
tw_c = project.connect_turbowarp_cloud(username="Username")
```

You can also set and get an Turbowarp Cloud Variable:

```
tw_c.set_cloud_variable(variable_name="Name", value=0),
```

```
tw_c.get_variable_data()
```

Cloud Errors:

If there are, check your code, otherwise use this URL: `https://turbowarp.org/<project ID>?cloud_host=wss://clouddata.turbowarp.org`, replace `<project ID>` with an valid Project ID.

Cloud Events:

“Use the same method as in Scratch but this time connect the cloud of a project on Turbowarp„ -Sid72020123

Advanced Cloud

You can also use Cloud Storage, that can:

- Create an variable
- Set an variable
- Get an variable
- Delete an variable
- Delete all variables
- Wait for an specified time
- Simple Syntax

“Maximum of 1024 characters can be set as a value to a variable.

You can create any number of variables!„

Before you send to Cloud Variables, you need an sprite from

https://scratch.mit.edu/projects/606881698/

Click **See inside**

And you need to read the instructions in the project

If you have it and created an project you can use now:

```
login =
scratchconnect.ScratchConnect("Use
rname", "Password") project =
login.connect_project(1)
cloud_storage =
project.create_cloud_storage(file_
name="data", rewrite_file=False,
edit_access=['Sid72020123'],all_ac
cess=False)
cloud_storage.start_cloud_loop(upd
ate_time=1,print_requests=True)
```
.

Cookie

If you are using an IDE like Replit, etc.

Scratch blocked Replit and other IDE’s, but **ScratchConnect** has an feature called Cookie Login:

```
scratch_cookie = { "Username": "Your
username", "SessionID": "Your
SessionID",
}
```

```
login =
scratchconnect.ScratchConnect(cookie=
scratch_cookie)  # Login with Cookie
Login.
```

“Note: While running the above code, ScratchConnect will give a warning that some features might not work if the cookie values are wrong. It's not an ERROR, it's a WARNING„ -Sid72020123

Advanced use of Scratch login:

```
scratch_cookie = { "Username": "Your username", "SessionID": "Your SessionID",
}
```

```
login = scratchconnect.ScratchConnect(username="USERNAME", password="PASSWORD", cookie=scratch_cookie, auto_cookie_login=True)
```

```
# Log in with cookie and enable the auto_cookie_login
```

Terminal

Terminal is an feature in **ScratchConnect 3.0+** it can get the data of Scratch User, Studio and Project in the Python console.

“To use this feature, you need to install additional dependencies required, by typing `pip install scratchconnect[terminal]` on your terminal„ -Sid72020123

Here is the code:

```
login = scratchconnect.ScratchConnect("Username", "Password")
terminal = login.create_new_terminal()
terminal.start()
```
.

“You can use many features in it. Just enter `help` to see the list of commands after the terminal starts„ -Sid72020123

Charts

Charts is an feature in **ScratchConnect 3.0+** used in which a user can get the data of Scratch User, Studio and Project in graphical format.

"Note: This feature uses the library `PYHTMLCHART` to create graphs. Any other library can be used in later versions

To use this feature, you need to install additional dependencies required, by typing pip install `scratchconnect[chart]` in the terminal „ - Sid72020123

User comparison Chart

```
login = scratchconnect.ScratchConnect("Username",
"Password") chart = login.create_new_chart()  #
Create a Chart object user_chart =
chart.user_stats_chart( usernames=["griffpatch",
"Will_Wam", "ScratchCat"]) user_table =
chart.user_stats_table( usernames=["griffpatch",
"Will_Wam", "ScratchCat"]) user_chart.open()
user_table.open()
```

To include only some required data in a chart or table, use the include_data parameter of the chart or table function and pass the value as list to get the required data. Example: ['Messages Count', 'Follower Count', 'Following Count']

You can also use any one or more options from the following list:

```
['Username', 'Messages Count', 'Follower
Count', 'Following Count', 'Total Loves',
 'Total Favourites', 'Total Projects
Count']
```

Studio comparison Chart

```
login = scratchconnect.ScratchConnect("Username",
"Password")


chart = login.create_new_chart()  # Create a Chart
object

studio_chart = chart.studio_stats_chart(
    studio_ids=[100, 101, 102])  # Create studio
stats comparison chart

studio_table = chart.studio_stats_table(
    studio_ids=[100, 101, 102])  # Create studio
stats comparison table

studio_chart.open()  # Open Studio chart
studio_table.open()  # Open Studio table
```

To include only some required data in a chart or table, use the include_data parameter of the chart or table function and pass the value as list to get the required data. Example: `['Comments Count', 'Followers Count', 'Managers Count']`

You can also use any one or more options from the following list:

```
['Studio ID', 'Comments Count',
'Followers Count', 'Managers Count',
'Projects Count']
```

Project comparison Chart

```
login = scratchconnect.ScratchConnect("Username",
"Password")


chart = login.create_new_chart()  # Create a Chart
object

project_chart = chart.project_stats_chart(
    project_ids=[104, 105, 106])  # Create project
stats comparison chart

project_table = chart.project_stats_table(
    project_ids=[104, 105, 106])  # Create project
stats comparison table

project_chart.open()  # Open Project chart
project_table.open()  # Open Project table
```

To include only some required data in a chart or table, use the `include_data` parameter of the chart or table function and pass the value as list to get the required data. Example: `['Views', 'Loves', 'Favourites']`

You can also use any one or more options from the following list:

```
['Project ID', 'Views', 'Loves', 'Favourites',
'Remixes', 'Version', 'Costumes', 'Blocks',
 'Variables', 'Assets']
```

Follower History Chart

```
login = scratchconnect.ScratchConnect("Username",
"Password")


chart = login.create_new_chart()  # Create a Chart
object

c =
chart.user_followers_history_chart(username="griffp
atch")  # Followers History Chart

t =
chart.user_followers_history_table(username="griffp
atch")  # Followers History Table

c.open()  # Open chart
t.open()  # Open table
```

Examples

1. Message Counter

```
import os
import scratchconnect
name = “username”
password = os.environ[“password”]
user = scratchconnect.ScratchConnect(name,
password)
project =
user.connect_project(project_id=1)variables =
project.connect_cloud_variables()
variables.set_cloud_variable(variable_name="N
ame", value=user.messages_count)
user.update_data()
```

2. Title following count

```
import os import scratchconnect name =
“username” password = os.environ[“password”]
user = scratchconnect.ScratchConnect(name,
password) project =
user.connect_project(project_id=1)
project.set_title(user.following_count)
user.update_data()
```

ScratchConnect is an library, that can connect Scratch Projects, Studios, Users and Cloud Variables!

WojtekGame