

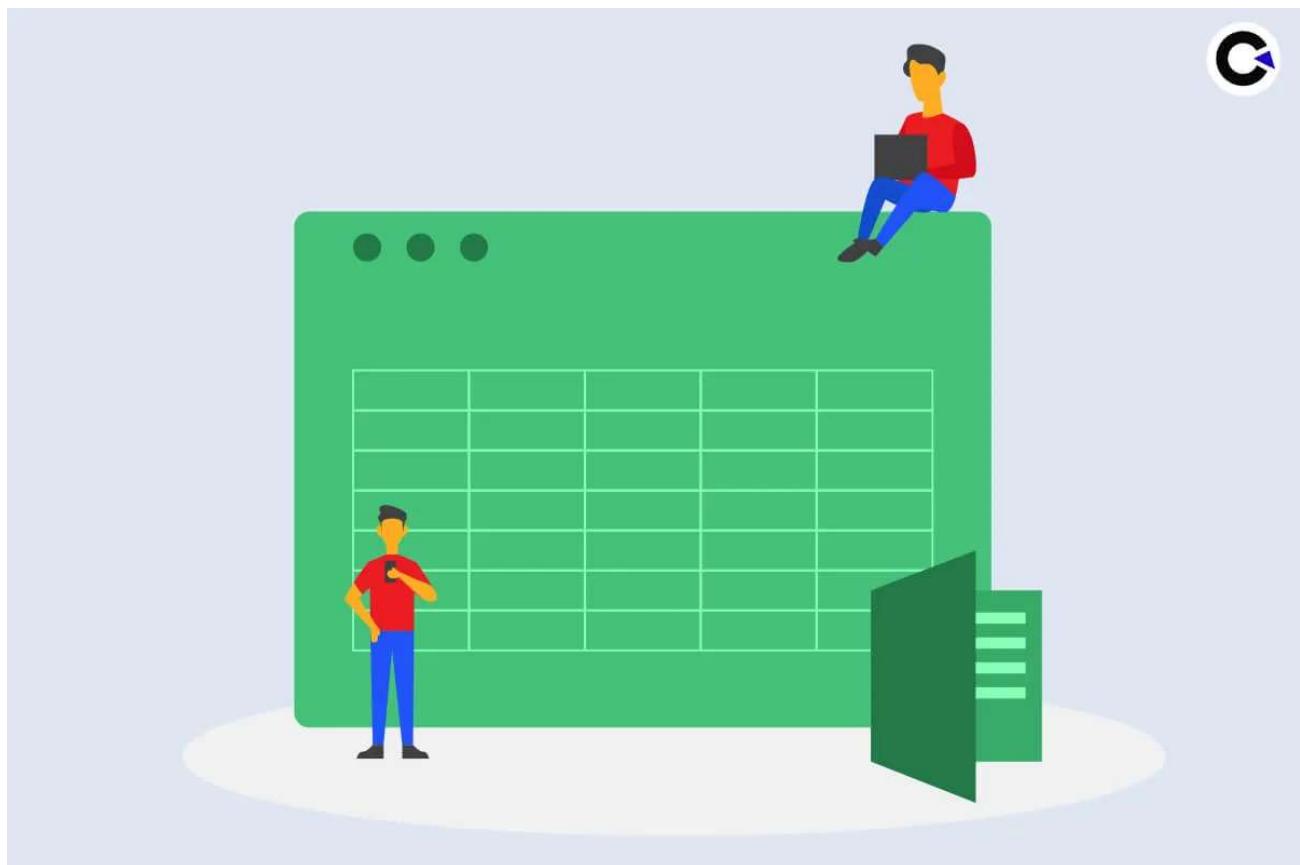
gspread Python tutorial for Google Sheets Automation

Learn the full setup process & find out how to open a spreadsheet, read & insert data into it and much more from our gspread Python Tutorial



Hannah Rivera

Posted On 23/02/2023



Listen to this blog

0:00 / 0:00

On average, more than 2 Billion people use Google Sheets on a monthly basis. It is a powerful tool for managing and analyzing data. It is a popular choice among individuals as it is very user-friendly in nature. Additionally, it is also widely used by professionals and businesses. With continuous & extensive usage, few tasks could end up becoming repetitive and time-consuming. But with the help of automation, you will be able to overcome this major issue and streamline your workflow and perform tasks more efficiently. Being an **automation testing company**, we have written this gspread Python tutorial to help you automate google sheets using Python. So let's get started.

There are actually several libraries such as pygsheets, gdata, and gspread that one can use to automate Google Sheets using Python. And we have picked gspread for this particular tutorial. We'll explore the basic setup & configuration and then dive into how we can use it.

We use cookies to ensure your best experience. For more information read our

[Privacy Policy](#)

DENY ACCEPT

Google Sheet Automation Set up

Before we explore how to automate Google Sheets using Python in our gspread Python Tutorial, we'll



[Install a Python Library](#)

[Create a New Project](#)

[Enable API & Services](#)

[Create Credentials](#)

[Create a Google Sheet](#)

Install a Python library

We have chosen gspread for this tutorial as it is one of the most popular and widely used libraries. So the first step in our gspread Python Tutorial is to install the gspread library that is needed to achieve Google Sheets automation using Python.

You can install gspread using the below command,

Command:

```
1 | pip install gspread
```

Output:

```
pip install gspread
```

```
Collecting gspread
  Downloading gspread-5.7.2-py3-none-any.whl (40 kB)
----- 40.5/40.5 kB 644.2 kB/s eta 0:00:00
Requirement already satisfied: google-auth-oauthlib>=0.4.1 in c:\users\admin\appdata\local\program
ckages (from gspread) (0.8.0)
Requirement already satisfied: google-auth>=1.12.0 in c:\users\admin\appdata\local\programs\python
```

Once you have installed the library, you'll need to set up authentication. And to do that, you'll have to create a "service account" in Google API Console and grant access to the Google Sheet you want to automate. Let's see what steps have to be followed to do it.

Create a New Project

Navigate to [Google Developers Console](#) and click on 'Create Project'.

For explanation purposes, we have created a project called 'Gsheet Reader' for our gspread Python Tutorial. You can 'Create' button.

We use cookies to ensure your best experience. For more information read our [Privacy Policy](#)

New Project



[MANAGE QUOTAS](#)

! Make sure all fields are correct to continue

Project name *

Gsheet Reader



Project ID: gsheet-reader-376808. It cannot be changed later. [EDIT](#)

Organization *

codoid.com



Select an organization to attach it to a project. This selection can't be changed later.

Location *

codoid.com

[BROWSE](#)

Parent organization or folder

[CREATE](#)

[CANCEL](#)

Enable API & Services

As the project has been created, you will see a new option to 'Enable APIs and Services' as shown below. You'll have to click on it to add the Google Sheet API.

The screenshot shows the Google Cloud API & Services interface. On the left, there's a sidebar with 'APIs & Services' selected. In the main area, there's a search bar at the top right. Below it, a button labeled '+ ENABLE APIs AND SERVICES' is highlighted with a red box. To its left, there are two tabs: 'Enabled APIs & services' (which is selected) and 'Library'. At the bottom right, there are some filter and search options.

There will be a list of APIs and you can enter an appropriate keyword such as 'sheet' in the search bar and select the 'Google Sheets API'.

The screenshot shows the search results for 'sheet' in the Google Cloud API & Services search bar. A single result, 'Google Sheets API', is listed. Its icon is a green document with a grid. Below the icon, the text 'Google Sheets API' and 'Google Enterprise API' is shown, with 'Google Enterprise API' having a question mark icon. A red box highlights this entire row. At the bottom of the screen, there's a cookie consent message.

You can then click

[Product details](#)


The Sheets API gives you full control over the content and appearance of your spreadsheet data.

[ENABLE](#) [TRY THIS API](#)

OVERVIEW

DOCUMENTATION

SUPPORT

Create Credentials:

Google Sheets API will now appear in your Enabled APIs & Services tab. You will see 3 sections namely Metrics, Quotas, and Credentials under it. So the next step in our gspread Python Tutorial would be to see how to create the required credentials to access the Google Sheet API.

So click the 'Create Credentials' button and follow the below steps.

The screenshot shows the Google Cloud Platform interface. In the top left, there's a navigation bar with 'Google Cloud' and a dropdown menu. A search bar is at the top right. On the left, a sidebar lists 'Enabled APIs & services' with options like 'Library', 'Credentials', 'OAuth consent screen', and 'Page usage agreements'. The main content area shows 'API/Service Details' for the 'Google Sheets API'. It has a brief description: 'Reads and writes Google Sheets.' and 'By Google Enterprise API'. Below this, it shows 'Service name: sheets.googleapis.com', 'Type: Public API', and 'Status: Enabled'. At the bottom of this card are three tabs: 'METRICS', 'QUOTAS', and 'CREDENTIALS'. To the right of the card are 'LEARN MORE' and 'MAINTENANCE & SUPPORT' links. In the top right corner of the main content area, there's a 'CREATE CREDENTIALS' button, which is highlighted with a red box.

Credential Type

Select 'Google Sheets API' in the 'Select an API' drop-down

Select the 'Application Data' radio button, and

Choose the 'No, I'm not using them' option in the last question.

Click 'NEXT'.

The screenshot shows the 'Create credentials' wizard. Step 1: Credential Type. It asks 'Which API are you using?' with a note about different APIs having different auth platforms. It shows a dropdown menu where 'Google Sheets API' is selected. Below it, it asks 'What data will you be accessing?' with a note about different credentials being required for different types of data. It shows two radio button options: 'User data' (which is not selected) and 'Application data' (which is selected). The 'Application data' option is described as belonging to your own application like Cloud Firestore. At the bottom, it asks 'Are you planning to use this API with Compute Engine, Kubernetes Engine, App Engine, or Cloud Functions?' with a note about applications running on GCE, GKE, GAE, and Cloud Functions not requiring credentials. It shows two radio button options: 'Yes, I'm using one or more' (not selected) and 'No, I'm not using them' (selected). A red arrow points to the 'NEXT' button at the bottom of the form.

Service Account Details

After which, you'll have to enter the Service account name & Service account ID in the respective fields



1 Service account details

Service account name: sheetreader
Display name for this service account:

Service account ID *: sheetreader

Email address: sheetreader@gsheet-reader-376808.iam.gserviceaccount.com

Service account description:
Describe what this service account will do

2 Grant this service account access to project (optional)

3 Grant users access to this service account (optional)

Grant Access

Now, you'll have to specify the level of access you are going to provide for the service account.

Click on 'Select a Role' and select 'Editor' under the 'Basic' section. Based on your needs, you can choose whichever role will be the aptest.

Press 'CONTINUE' and leave the other optional fields and click DONE.

Service account details

2 Grant this service account access to project (optional)

Grant this service account access to Gsheet Reader so that it has permission to complete specific actions on the resources in your project. [Learn more](#)

Role

Quick access	Roles
Currently used	Browser
Basic	Editor
By product or service	Owner
Access Approval	Viewer

3 Access Context Manager

You will now be able to see the newly created Service account under the Credentials section.

We use cookies to ensure your best experience. For more information read our [Privacy Policy](#)

Service name sheets.googleapis.com	Type Public API	Status Enabled	LEARN MORE	MAINTENANCE & SUPPORT
METRICS	QUOTAS	CREDENTIALS		



To view all credentials visit [Credentials in APIs & Services](#)

⚠ Remember to configure the OAuth consent screen with information about your application.

[CONFIGURE CONSENT SCREEN](#)

OAuth 2.0 Client IDs

<input type="checkbox"/> Name	Creation date ↓	Type	Client ID	Actions
No OAuth clients to display				

Service Accounts

[Manage service accounts](#)

<input type="checkbox"/> Email	Name ↑	Actions
<input type="checkbox"/> sheetreader@gsheet-reader-376808.iam.gserviceaccount.com	sheetreader	

Under the Keys Section, you can click on 'ADD KEY' and choose 'Create new key' to create a credential secret key.

DETAILS PERMISSIONS KEYS METRICS LOGS

Keys

⚠ Service account keys could pose a security risk if compromised. We recommend you avoid downloading service account keys and instead use the [Workload Identity Federation](#). You can learn more about the best way to authenticate service accounts on Google Cloud [here](#).

Add a new key pair or upload a public key certificate from an existing key pair.

Block service account key creation using [organization policies](#).
[Learn more about setting organization policies for service accounts](#)

ADD KEY

Create new key Key creation date Key expiration date

Upload existing key

There are two key types, and as suggested by Google, it is better to choose the JSON key type. After making the selection, click the 'CREATE' button.

Create private key for "sheetreader"

Downloads a file that contains the private key. Store the file securely because this key can't be recovered if lost.

Key type

JSON
Recommended

P12
For backward compatibility with code using the P12 format

CANCEL

CREATE

Once you click the button, the key will be downloaded to your computer. Please make sure to not share this file with anyone else and keep it safe.

We use cookies to ensure your best experience. For more information read our [Privacy Policy](#)

Private key saved to your computer

gsheet-reader-376808-2b0922978326.json allows access to your cloud resources, so store it securely. [Learn more best practices](#)

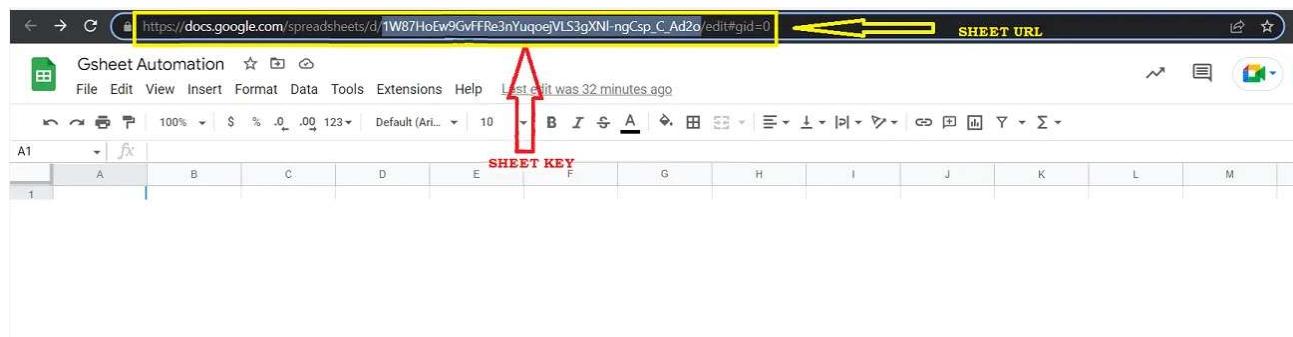
CLOSE

Tip: It will be helpful if you rename the file to credentials.json as it will be easy for you to remember the file name.

Create a Google Sheet

So the last step in our setup process is to create a Google Sheet that we can use to store our data. Once we are clear with that, we can go ahead and see the different **Python code commands** you can use to perform Google Sheet Automation.

If you already have a Google sheet, you can just copy the sheet url or sheet id as shown in the image as this copied string will be needed to access Google Sheets.



gspread Python Tutorial

Everything you'll need to achieve Google Sheet Automation is now ready. So you can start automating the listed actions by typing the Python code commands we have mentioned in a code editor.

[Opening a Spreadsheet](#)

[Get a Worksheet's Name](#)

[Read Data from a Sheet](#)

[Insert Data into the Sheet](#)

[Update Multiple Ranges](#)

We use cookies to ensure your best experience. For more information read our

[Privacy Policy](#)

Opening a Sheet

First up in our gspread Python Tutorial, we'll be seeing how to open a spreadsheet using Python code commands. Remember the URL we had copied after creating a new Google Sheet? We will now use it to



Code:

```

1 import gspread
2
3 Sheet_credential = gspread.service_account("credential.json")
4
5 # Open Spreadsheet by URL
6 # spreadsheet = Sheet_credential.open_by_url('paste your sheet url')
7
8 # Open Spreadsheet by key
9 spreadsheet = Sheet_credential.open_by_key('paste your sheet key')
10 print(spreadsheet.title)

```

Output:

You can see that the name of the Google Sheet we created (Gsheet Automation) has been fetched here.

Get a Worksheet's Name

A Spreadsheet can have more than one worksheet and if you wish to print the name and ID of a worksheet, you can use the below Python code commands.

Code:

```

1 # to print worksheet name using sheet id
2 worksheet = spreadsheet.get_worksheet(0)
3
4 # to print worksheet name using sheet name
5 worksheet = spreadsheet.worksheet('Sample Sheet')
6 print(worksheet)

```

Output:

Read Data from a Sheet

Next up in our gspread Python tutorial, we're going to see the command you'll have to use to read data from a Google Sheet and printing it.

We use cookies to ensure your best experience. For more information read our Privacy Policy

Code:

```

1 all_values = worksheet.get_all_records()
2 for value in all_values:
3     print(value)

```

Output:

Insert Data into the Sheet

Reading the data alone isn't enough. You'll have to insert data into a sheet as well. So, let's look at the Python Code command you'll have to use to achieve that. In our example, we will be updating multiple values after a defined row and read & print the updated values.

Code:

```

1 # Update multiple values after A6 row
2 worksheet.update('A7', [[ "106", "Robert", "J", "Pass" ], [ "107", "Robert", "G", "Fail"]])
3
4 # read data after update
5 update_data = worksheet.get_all_values()
6 for valu in update_data:
7     print(valu)

```

Output:

Update Multiple Ranges

Finally in our gspread Python tutorial, we are going to see the commands you can use to update multiple ranges of data at the same time.

Code:

We use cookies to ensure your best experience. For more information read our
[Privacy Policy](#)

```
1 # Update multipl
```

```

2 worksheet.batch_update([
3     'range': 'E1:F2',
6     'range': 'G1:H2',
7     'values': [[['A', 'AB'], ['C', 'CD']]],
8 ])
9 # To Read data after the update
10 update_data = worksheet.get_all_values()
11 for valu in update_data:
12     print(valu)

```

Output:**Related Blogs**

- Desktop App Automation Testing using Python
- Selenium WebDriver with Python Cheat Sheet
- Java Excel Libraries

CONCLUSION

So in our gspread Python tutorial, we have laid the foundations you'll have to know to achieve Google Sheets automation using Python. By following these simple steps, you can easily automate tasks such as data entry, data analysis, and report generation. We hope this will help you significantly improve your workflow and save you time. Being a leading big data analytics testing company, we have used Python libraries and found gspread to be the best of the lot. Hope you find it useful too.

[Share Via -](#)

Comments(0)

Similar Blogs



We use cookies to ensure your best experience. For more information read our
[Privacy Policy](#)



Automation Testing

Automation Testing

Automation Testing

Automation

A Conclusive Allure Report Tutorial with Sample Report**Allure report vs Extent report. Which is Better?****Cypress Limitations you should be aware of****Best Cypress Must Know**

Search..



SUBSCRIBE



Sign up for our newsletter to never miss out on our latest blogs.

username@mail.com



Talk to our Experts

Full Name*

Business Email*

Phone Number

Message*

SUBMIT

We use cookies to ensure your best experience. For more information read our
[Privacy Policy](#)



Amazing clients who
trust us



SERVICES	TEST AUTOMATION	RESOURCES	EMAIL US	CALL US
Automation		Blogs	For Business	India (Toll Free):
Testing	Selenium Testing	Case Studies	Enquiries:	1800 (212) 6988
Agile Testing	Visual Automation	Memes	sales@codoid.com	United States:
Performance	Testing	Comics	For Marketing	+1 833-685-6161
Testing	Desktop App	Videos	Enquiries:	Whatsapp :
Data Analytics	Automation		marketing@codoi	+91 9600062660
Testing	Testing		d.com	
Regression Testing	Mobile App		For Job	
	Automation		Enquiries:	
	Testing		recruitment@codoi	
	API Automation		id.com	
	Testing			

Copyright © Codoid. All Rights Reserved.

We use cookies to ensure your best experience. For more information read our
[Privacy Policy](#)