


The `sendgrid` Package

Prerequisites: [Environment Variables](#), [The `dotenv` Package](#)

The `sendgrid` package provides some useful emailing capabilities via the [SendGrid Email Delivery Service](#). 



Reference:

- [Source](#)
- [Package](#)
- [Example Usage](#)
- [Heroku's SendGrid Guide](#)
- [SendGrid Account Types, including Free Tier](#)
- [SendGrid Account Signup](#)
- [Obtaining API Keys](#)

Installation

From within a virtual environment, install `sendgrid`, if necessary:

```
pip install sendgrid==6.0.5
```

NOTE: previous versions of these instructions were applicable for `sendgrid` package version 5.6.0, but the examples below apply to a more current version (6.0.5)

Setup

First, [sign up for a free account](#), then click the link in a confirmation email to verify your account. Then [create an API Key](#) with "full access" permissions.

To setup the usage examples below, store the API Key value in an environment variable called `SENDGRID_API_KEY`. Also set an environment variable called `MY_EMAIL_ADDRESS` to be the email address you just associated with your SendGrid account (e.g. "abc123@gmail.com").

Usage

Send yourself an email:

```
import os
from dotenv import load_dotenv
```

```

from sendgrid import SendGridAPIClient
from sendgrid.helpers.mail import Mail

load_dotenv()

SENDGRID_API_KEY = os.environ.get("SENDGRID_API_KEY", "OOPS, please set env
var called 'SENDGRID_API_KEY'")
MY_ADDRESS = os.environ.get("MY_EMAIL_ADDRESS", "OOPS, please set env var
called 'MY_EMAIL_ADDRESS'")

client = SendGridAPIClient(SENDGRID_API_KEY) #> <class
'sendgrid.sendgrid.SendGridAPIClient>
print("CLIENT:", type(client))

subject = "Your Receipt from the Green Grocery Store"

html_content = "Hello World"
#
# or maybe ...
#html_content = "Hello <strong>World</strong>"
#
# or maybe ...
#html_list_items = "<li>You ordered: Product 1</li>"
#html_list_items += "<li>You ordered: Product 2</li>"
#html_list_items += "<li>You ordered: Product 3</li>"
#html_content = f"""
#<h3>Hello this is your receipt</h3>
#<p>Date: _____</p>
#<ol>
#     {html_list_items}
#</ol>
#"""
print("HTML:", html_content)

message = Mail(from_email=MY_ADDRESS, to_emails=MY_ADDRESS, subject=subject,
html_content=html_content)

try:
    response = client.send(message)

    print("RESPONSE:", type(response)) #> <class
'python_http_client.client.Response'>
    print(response.status_code) #> 202 indicates SUCCESS
    print(response.body)
    print(response.headers)

except Exception as e:
    print("OOPS", e.message)

```

NOTE: the message was successfully sent if you see a 202 status code

NOTE: this message might take a minute to send, and it might be in your spam folder to start

Email Templates

Thanks to @mgallea for surfacing these email template capabilities

If you'd like further control over the content displayed in the email's body, you can use Sendgrid's email templates.

Reference:

- <https://sendgrid.com/docs/ui/sending-email/how-to-send-an-email-with-dynamic-transactional-templates/>

Let's try sending a simple receipt:

Navigate to https://sendgrid.com/dynamic_templates and press the "Create Template" button on the top right. Give it a name like "example-receipt", and click "Save". At this time, you should see your template's unique identifier (e.g. "d-b902ae61c68f40dbbd1103187a9736f0"). Copy this value and store it in an environment variable called `SENDGRID_TEMPLATE_ID`.

Back in the SendGrid platform, click "Add Version" to create a new version of this template and select the "Code Editor" as your desired editing mechanism.

At this point you should be able to paste the following HTML into the "Code" tab, and the corresponding example data in the "Test Data" tab:

```


<h3>Hello this is your receipt</h3>

<p>Date: {{human_friendly_timestamp}}</p>

<p>Total: {{total_price_usd}}</p>

<ul>
{{#each products}}
  <li>You ordered: {{this.name}}</li>
```

```
{{/each}}
</ul>
```

NOTE: the "handlebars" syntax above is like HTML, but allows us to construct HTML dynamically based on some data like the example below

```
{
  "total_price_usd": "$14.95",
  "human_friendly_timestamp": "July 4th, 2019 10:00 AM",
  "products": [
    {"id": 1, "name": "Product 1"},
    {"id": 2, "name": "Product 2"},
    {"id": 3, "name": "Product 3"},
    {"id": 2, "name": "Product 2"},
    {"id": 1, "name": "Product 1"}
  ]
}
```

Finally, configure the template's subject by clicking on "Settings" in the left sidebar. Choose an email subject like "Your Receipt from the Green Grocery Store". Then click "Save Template".

After configuring and saving the email template, we should be able to send an email using the template:

```
import os
from dotenv import load_dotenv
from sendgrid import SendGridAPIClient
from sendgrid.helpers.mail import Mail

load_dotenv()

SENDGRID_API_KEY = os.environ.get("SENDGRID_API_KEY", "OOPS, please set env var called 'SENDGRID_API_KEY'")
SENDGRID_TEMPLATE_ID = os.environ.get("SENDGRID_TEMPLATE_ID", "OOPS, please set env var called 'SENDGRID_TEMPLATE_ID'")
MY_ADDRESS = os.environ.get("MY_EMAIL_ADDRESS", "OOPS, please set env var called 'MY_EMAIL_ADDRESS'")

#print("API KEY:", SENDGRID_API_KEY)
#print("TEMPLATE ID:", SENDGRID_TEMPLATE_ID)
#print("EMAIL ADDRESS:", MY_ADDRESS)

template_data = {
  "total_price_usd": "$14.95",
  "human_friendly_timestamp": "June 1st, 2019 10:00 AM",
```

```

    "products":[
        {"id":1, "name": "Product 1"},
        {"id":2, "name": "Product 2"},
        {"id":3, "name": "Product 3"},
        {"id":2, "name": "Product 2"},
        {"id":1, "name": "Product 1"}
    ]
} # or construct this dictionary dynamically based on the results of some
other process :-D

client = SendGridAPIClient(SENDGRID_API_KEY)
print("CLIENT:", type(client))

message = Mail(from_email=MY_ADDRESS, to_emails=MY_ADDRESS)
print("MESSAGE:", type(message))

message.template_id = SENDGRID_TEMPLATE_ID

message.dynamic_template_data = template_data

try:
    response = client.send(message)
    print("RESPONSE:", type(response))
    print(response.status_code)
    print(response.body)
    print(response.headers)

except Exception as e:
    print("OOPS", e)

```