

```
import os
```

```
import json
```

```
from sendgrid import SendGridAPIClient
```

```
from sendgrid.helpers.mail import *
```

```
# NOTE: you will need move this file to the root
```

```
# directory of this project to execute properly.
```

```
def build_hello_email():
```

```
    ## Send a Single Email to a Single Recipient
```

```
    message = Mail(from_email=From('from@example.com', 'Example From  
Name'),
```

```
        to_emails=To('to@example.com', 'Example To Name'),
```

```
        subject=Subject('Sending with SendGrid is Fun'),
```

```
        plain_text_content=PlainTextContent('and easy to do anywhere, even  
with Python'),
```

```
        html_content=HtmlContent('<strong>and easy to do anywhere, even  
with Python</strong>'))
```

```
    try:
```

```
        print(json.dumps(message.get(), sort_keys=True, indent=4))
```

```
return message.get()
```

```
except SendGridException as e:
```

```
    print(e.message)
```

```
mock_personalization = Personalization()
```

```
personalization_dict = get_mock_personalization_dict()
```

```
for cc_addr in personalization_dict['cc_list']:
```

```
    mock_personalization.add_to(cc_addr)
```

```
for bcc_addr in personalization_dict['bcc_list']:
```

```
    mock_personalization.add_bcc(bcc_addr)
```

```
for header in personalization_dict['headers']:
```

```
    mock_personalization.add_header(header)
```

```
for substitution in personalization_dict['substitutions']:
```

```
    mock_personalization.add_substitution(substitution)
```

```
for arg in personalization_dict['custom_args']:
```

```
    mock_personalization.add_custom_arg(arg)
```

```
mock_personalization.subject = personalization_dict['subject']
```

```
mock_personalization.send_at = personalization_dict['send_at']
```

```
message.add_personalization(mock_personalization)
```

```
return message
```

```
def get_mock_personalization_dict():
```

```
    """Get a dict of personalization mock."""
```

```
    mock_pers = dict()
```

```
    mock_pers['to_list'] = [To("test1@example.com",
```

```
        "Example User"),
```

```
        To("test2@example.com",
```

```
        "Example User")]
```

```
    mock_pers['cc_list'] = [To("test3@example.com",
```

```
        "Example User"),
```

```
        To("test4@example.com",
```

```
        "Example User")]
```

```
    mock_pers['bcc_list'] = [To("test5@example.com"),
```

```
        To("test6@example.com")]
```

```
    mock_pers['subject'] = ("Hello World from the Personalized "
```

"SendGrid Python Library")

```
mock_pers['headers'] = [Header("X-Test", "test"),  
                        Header("X-Mock", "true")]
```

```
mock_pers['substitutions'] = [Substitution("%name%", "Example User"),  
                             Substitution("%city%", "Denver")]
```

```
mock_pers['custom_args'] = [CustomArg("user_id", "343"),  
                           CustomArg("type", "marketing")]
```

```
mock_pers['send_at'] = 1443636843
```

```
return mock_pers
```

```
def build_multiple_emails_personalized():
```

```
    # Note that the domain for all From email addresses must match
```

```
    message = Mail(from_email=From('from@example.com', 'Example From  
Name'),
```

```
                  subject=Subject('Sending with SendGrid is Fun'),
```

```
                  plain_text_content=PlainTextContent('and easy to do anywhere, even  
with Python'),
```

```
                  html_content=HtmlContent('<strong>and easy to do anywhere, even  
with Python</strong>'))
```

```

mock_personalization = Personalization()

mock_personalization.add_to(To('test@example.com', 'Example User 1'))

mock_personalization.add_cc(Cc('test1@example.com', 'Example User 2'))

message.add_personalization(mock_personalization)


mock_personalization_2 = Personalization()

mock_personalization_2.add_to(To('test2@example.com', 'Example User 3'))

mock_personalization_2.set_from(From('from@example.com', 'Example
From Name 2'))

mock_personalization_2.add_bcc(Bcc('test3@example.com', 'Example User
4'))

message.add_personalization(mock_personalization_2)


try:

    print(json.dumps(message.get(), sort_keys=True, indent=4))

    return message.get()


except SendGridException as e:

    print(e.message)


return message


def build_attachment1():

    """Build attachment mock. Make sure your content is base64 encoded before
    passing into attachment.content.
```

Another example: [https://github.com/sendgrid/sendgrid-python/blob/HEAD/use\\_cases/attachment.md](https://github.com/sendgrid/sendgrid-python/blob/HEAD/use_cases/attachment.md)"""

```
attachment = Attachment()

attachment.file_content =
("TG9yZW0gaXBzdW0gZG9sb3Igc2l0IGFtZXQsIGNvb3N1I"
"Y3RldHVyIGFkaXBpc2NpbmcgZWxpdC4gQ3JhcyBwdW12")

attachment.file_type = "application/pdf"
attachment.file_name = "balance_001.pdf"
attachment.disposition = "attachment"
attachment.content_id = "Balance Sheet"

return attachment
```

```
def build_attachment2():

    """Build attachment mock."""

    attachment = Attachment()

    attachment.file_content = "BwdW"

    attachment.file_type = "image/png"

    attachment.file_name = "banner.png"

    attachment.disposition = "inline"

    attachment.content_id = "Banner"

    return attachment
```

```

def build_kitchen_sink():
    """All settings set"""

    from sendgrid.helpers.mail import (
        Mail, From, To, Cc, Bcc, Subject, PlainTextContent,
        HtmlContent, SendGridException, Substitution,
        Header, CustomArg, SendAt, Content, MimeType, Attachment,
        FileName, FileContent, FileType, Disposition, ContentId,
        TemplateId, Section, ReplyTo, Category, BatchId, Asm,
        GroupId, GroupsToDisplay, IpPoolName, MailSettings,
        BccSettings, BccSettingsEmail, BypassListManagement,
        FooterSettings, FooterText, FooterHtml, SandBoxMode,
        SpamCheck, SpamThreshold, SpamUrl, TrackingSettings,
        ClickTracking, SubscriptionTracking, SubscriptionText,
        SubscriptionHtml, SubscriptionSubstitutionTag,
        OpenTracking, OpenTrackingSubstitutionTag, Ganalytics,
        UtmSource, UtmMedium, UtmTerm, UtmContent, UtmCampaign)

    import time

    import datetime

    message = Mail()

    # Define Personalizations

    message.to = To('test1@sendgrid.com', 'Example User1', p=0)

```

```
message.to = [  
    To('test2@sendgrid.com', 'Example User2', p=0),  
    To('test3@sendgrid.com', 'Example User3', p=0)  
]
```

```
message.cc = Cc('test4@example.com', 'Example User4', p=0)  
message.cc = [  
    Cc('test5@example.com', 'Example User5', p=0),  
    Cc('test6@example.com', 'Example User6', p=0)  
]
```

```
message.bcc = Bcc('test7@example.com', 'Example User7', p=0)  
message.bcc = [  
    Bcc('test8@example.com', 'Example User8', p=0),  
    Bcc('test9@example.com', 'Example User9', p=0)  
]
```

```
message.subject = Subject('Sending with SendGrid is Fun 0', p=0)
```

```
message.header = Header('X-Test1', 'Test1', p=0)  
message.header = Header('X-Test2', 'Test2', p=0)  
message.header = [  
    Header('X-Test3', 'Test3', p=0),  
    Header('X-Test4', 'Test4', p=0)
```



]

message.substitution = Substitution('%name1%', 'Example Name 1', p=0)

message.substitution = Substitution('%city1%', 'Example City 1', p=0)

message.substitution = [

    Substitution('%name2%', 'Example Name 2', p=0),

    Substitution('%city2%', 'Example City 2', p=0)

]

message.custom\_arg = CustomArg('marketing1', 'true', p=0)

message.custom\_arg = CustomArg('transactional1', 'false', p=0)

message.custom\_arg = [

    CustomArg('marketing2', 'false', p=0),

    CustomArg('transactional2', 'true', p=0)

]

message.send\_at = SendAt(1461775051, p=0)

message.to = To('test10@example.com', 'Example User10', p=1)

message.to = [

    To('test11@example.com', 'Example User11', p=1),

    To('test12@example.com', 'Example User12', p=1)

]

message.cc = Cc('test13@example.com', 'Example User13', p=1)

message.cc = [

Cc('test14@example.com', 'Example User14', p=1),

Cc('test15@example.com', 'Example User15', p=1)

]

message.bcc = Bcc('test16@example.com', 'Example User16', p=1)

message.bcc = [

Bcc('test17@example.com', 'Example User17', p=1),

Bcc('test18@example.com', 'Example User18', p=1)

]

message.header = Header('X-Test5', 'Test5', p=1)

message.header = Header('X-Test6', 'Test6', p=1)

message.header = [

Header('X-Test7', 'Test7', p=1),

Header('X-Test8', 'Test8', p=1)

]

message.substitution = Substitution('%name3%', 'Example Name 3', p=1)

message.substitution = Substitution('%city3%', 'Example City 3', p=1)

message.substitution = [

Substitution('%name4%', 'Example Name 4', p=1),

Substitution('%city4%', 'Example City 4', p=1)

]

```
message.custom_arg = CustomArg('marketing3', 'true', p=1)
```

```
message.custom_arg = CustomArg('transactional3', 'false', p=1)
```

```
message.custom_arg = [
```

```
    CustomArg('marketing4', 'false', p=1),
```

```
    CustomArg('transactional4', 'true', p=1)
```

```
]
```

```
message.send_at = SendAt(1461775052, p=1)
```

```
message.subject = Subject('Sending with SendGrid is Fun 1', p=1)
```

```
# The values below this comment are global to entire message
```

```
message.from_email = From('help@twilio.com', 'Twilio SendGrid')
```

```
message.reply_to = ReplyTo('help_reply@twilio.com', 'Twilio SendGrid  
Reply')
```

```
message.subject = Subject('Sending with SendGrid is Fun 2')
```

```
message.content = Content(MimeType.text, 'and easy to do anywhere, even  
with Python')
```

```
message.content = Content(MimeType.html, '<strong>and easy to do  
anywhere, even with Python</strong>')
```

```
message.content = [  
    Content('text/calendar', 'Party Time!!'),  
    Content('text/custom', 'Party Time 2!!')  
]
```

```
message.attachment = Attachment(FileContent('base64 encoded content 1'),  
    FileName('balance_001.pdf'),  
    FileType('application/pdf'),  
    Disposition('attachment'),  
    ContentId('Content ID 1'))
```

```
message.attachment = [  
    Attachment(FileContent('base64 encoded content 2'),  
        FileName('banner.png'),  
        FileType('image/png'),  
        Disposition('inline'),  
        ContentId('Content ID 2')),  
    Attachment(FileContent('base64 encoded content 3'),  
        FileName('banner2.png'),  
        FileType('image/png'),  
        Disposition('inline'),  
        ContentId('Content ID 3'))  
]
```

```
message.template_id = TemplateId('13b8f94f-bcae-4ec6-b752-70d6cb59f932')
```

```
message.section = Section('%section1%', 'Substitution for Section 1 Tag')
```

```
message.section = [  
    Section('%section2%', 'Substitution for Section 2 Tag'),  
    Section('%section3%', 'Substitution for Section 3 Tag')  
]
```

```
message.header = Header('X-Test9', 'Test9')
```

```
message.header = Header('X-Test10', 'Test10')
```

```
message.header = [  
    Header('X-Test11', 'Test11'),  
    Header('X-Test12', 'Test12')  
]
```

```
message.category = Category('Category 1')
```

```
message.category = Category('Category 2')
```

```
message.category = [  
    Category('Category 1'),  
    Category('Category 2')  
]
```

```
message.custom_arg = CustomArg('marketing5', 'false')
message.custom_arg = CustomArg('transactional5', 'true')
message.custom_arg = [
    CustomArg('marketing6', 'true'),
    CustomArg('transactional6', 'false')
]

message.send_at = SendAt(1461775053)

message.batch_id = BatchId("HkJ5yLYULb7Rj8GKSx7u025ouWVlMgAi")

message.asm = Asm(GroupId(1), GroupsToDisplay([1,2,3,4]))

message.ip_pool_name = IpPoolName("IP Pool Name")

mail_settings = MailSettings()
mail_settings.bcc_settings = BccSettings(False,
BccSettingsTo("bcc@twilio.com"))

mail_settings.bypass_list_management = BypassListManagement(False)

mail_settings.footer_settings = FooterSettings(True, FooterText("w00t"),
FooterHtml("<string>w00t!<strong>"))

mail_settings.sandbox_mode = SandBoxMode(True)

mail_settings.spam_check = SpamCheck(True, SpamThreshold(5),
SpamUrl("https://example.com"))

message.mail_settings = mail_settings
```

```
tracking_settings = TrackingSettings()

tracking_settings.click_tracking = ClickTracking(True, False)

tracking_settings.open_tracking = OpenTracking(True,
OpenTrackingSubstitutionTag("open_tracking"))

tracking_settings.subscription_tracking = SubscriptionTracking(
    True,
    SubscriptionText("Goodbye"),
    SubscriptionHtml("<strong>Goodbye!</strong>"),
    SubscriptionSubstitutionTag("unsubscribe"))

tracking_settings.ganalytics = Ganalytics(
    True,
    UtmSource("utm_source"),
    UtmMedium("utm_medium"),
    UtmTerm("utm_term"),
    UtmContent("utm_content"),
    UtmCampaign("utm_campaign"))

message.tracking_settings = tracking_settings

return message
```

```
def send_multiple_emails_personalized():
```

```
    # Assumes you set your environment variable:
```

```
# https://github.com/sendgrid/sendgrid-  
python/blob/HEAD/TROUBLESHOOTING.md#environment-variables-and-  
your-sendgrid-api-key
```

```
message = build_multiple_emails_personalized()  
  
sendgrid_client =  
SendGridAPIClient(os.environ.get('SENDGRID_API_KEY'))  
  
response = sendgrid_client.send(message=message)  
  
print(response.status_code)  
  
print(response.body)  
  
print(response.headers)
```

```
def send_hello_email():
```

```
    # Assumes you set your environment variable:  
  
    # https://github.com/sendgrid/sendgrid-  
python/blob/HEAD/TROUBLESHOOTING.md#environment-variables-and-  
your-sendgrid-api-key
```

```
    message = build_hello_email()  
  
    sendgrid_client =  
SendGridAPIClient(os.environ.get('SENDGRID_API_KEY'))  
  
    response = sendgrid_client.send(message=message)  
  
    print(response.status_code)  
  
    print(response.body)  
  
    print(response.headers)
```

```
def send_kitchen_sink():
```



```
# Assumes you set your environment variable:

# https://github.com/sendgrid/sendgrid-
python/blob/HEAD/TROUBLESHOOTING.md#environment-variables-and-
your-sendgrid-api-key

message = build_kitchen_sink()

sendgrid_client =
SendGridAPIClient(os.environ.get('SENDGRID_API_KEY'))

response = sendgrid_client.send(message=message)

print(response.status_code)

print(response.body)

print(response.headers)


## this will actually send an email

# send_hello_email()


## this will send multiple emails

# send_multiple_emails_personalized()


## this will only send an email if you set SandBox Mode to False

# send_kitchen_sink()
```