Hello! Thank you for choosing to help contribute to one of the Twilio SendGrid open source libraries. There are many ways you can contribute and help is always welcome. We simply ask that you follow the following contribution policies.

All third party contributors acknowledge that any contributions they provide will be made under the same open source license that the open source project is provided under.

- [Roadmap & Milestones](#roadmap)

- [Feature Request](#feature-request)

- [Submit a Bug Report](#submit-a-bug-report)

- [Improvements to the Codebase](#improvements-to-the-codebase)

- [Understanding the Code Base](#understanding-the-codebase)

- [Testing](#testing)

- [Style Guidelines & Naming Conventions](#style-guidelines-and-naming-conventions)

- [Creating a Pull Request](#creating-a-pull-request)

- [Code Reviews](#code-reviews)

<a name="roadmap"></a>

We use [GitHub Projects](https://github.com/sendgrid/sendgrid-csharp/projects) to help define current roadmaps, please feel free to grab an issue from our [GitHub Issues](https://github.com/sendgrid/sendgrid-csharp/issues). Please indicate that you have begun work on it to avoid collisions. Once a PR is made, community review, comments, suggestions and additional PRs are welcomed and encouraged.

There are a few ways to contribute, which we'll enumerate below:

<a name="feature-request"></a>

## Feature Request

If you'd like to make a feature request, please read this section.

The GitHub issue tracker is the preferred channel for library feature requests, but please respect the following restrictions:

- Please \*\*search for existing issues\*\* in order to ensure we don't have duplicate bugs/feature requests.

- Please be respectful and considerate of others when commenting on issues

<a name="submit-a-bug-report"></a>

## Submit a Bug Report

Note: DO NOT include your credentials in ANY code examples, descriptions, or media you make public.

A software bug is a demonstrable issue in the code base. In order for us to diagnose the issue and respond as quickly as possible, please add as much detail as possible into your bug report.

Before you decide to create a new issue, please try the following:

1. Check the Github issues tab if the identified issue has already been reported, if so, please add a +1 to the existing post.

2. Update to the latest version of this code and check if issue has already been fixed

3. Copy and fill in the Bug Report Template we have provided below

### Please use our Bug Report Template

In order to make the process easier, we've included a [sample bug report template](ISSUE\_TEMPLATE.md).

<a name="improvements-to-the-codebase"></a>

## Improvements to the Codebase

We welcome direct contributions to the sendgrid-csharp code base. Thank you!

Please note that we utilize the [Gitflow Workflow](https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow) for Git to help keep project development organized and consistent.

### Development Environment ###

#### Install and Run Locally ####

##### Prerequisites #####

- .NET 4.5.2+

- [Visual Studio Community 2017](https://www.visualstudio.com/downloads/)

##### Initial setup: #####

```bash

git clone https://github.com/sendgrid/sendgrid-csharp.git

```

- Open `sendgrid-csharp/SendGrid.sln`

### Environment Variables

First, get your free Twilio SendGrid account [here](https://sendgrid.com/free?source=sendgrid-csharp).

Next, update your Environment with your [SENDGRID\_APIKEY](https://app.sendgrid.com/settings/api\_keys).

##### Execute: #####

- Check out the documentation for [Web API v3 endpoints](https://sendgrid.com/docs/API\_Reference/Web\_API\_v3/index.html).

- Review the corresponding [examples](https://github.com/sendgrid/sendgrid-csharp/blob/master/examples).

- You can add your test code to our [Example Project](https://github.com/sendgrid/sendgrid-csharp/blob/master/ExampleCoreProject/Example.cs).

<a name="understanding-the-codebase"></a>

## Understanding the Code Base

- \*\*[/examples](https://github.com/sendgrid/sendgrid-csharp/blob/master/examples)\*\*

- Examples that demonstrate usage.

- \*\*[/ExampleCoreProject/Example.cs](https://github.com/sendgrid/sendgrid-csharp/blob/master/ExampleCoreProject/Example.cs)\*\*

- A working .NET Core example project for testing.

- \*\*[/ExampleNet45Project/Example.cs](https://github.com/sendgrid/sendgrid-csharp/blob/master/ExampleNet45Project/Example.cs)\*\*

- A working .NET 4.5.2 example project for testing.

- \*\*[/src/SendGrid/SendGridClient.cs](https://github.com/sendgrid/sendgrid-csharp/blob/master/src/SendGrid/SendGridClient.cs)\*\*

- The main interface to the Twilio SendGrid API is the class `SendGridClient`.

- \*\*[/tests/SendGrid.Tests/Integration.cs](https://github.com/sendgrid/sendgrid-csharp/blob/master/tests/SendGrid.Tests/Integration.cs)\*\*

- Integration tests

<a name="testing"></a>

## Testing

All PRs require passing tests before the PR will be reviewed. All test files are in the [`SendGrid.Tests`](https://github.com/sendgrid/sendgrid-csharp/blob/master/tests/SendGrid.Tests) directory. For the purposes of contributing to this repo, please update the [`Integration.cs`](https://github.com/sendgrid/sendgrid-csharp/blob/master/tests/SendGrid.Tests/Integration.cs) file with unit tests as you modify the code.

The integration tests require a Twilio SendGrid mock API in order to execute. We've simplified setting this up using Docker to run the tests. You will just need [Docker Desktop](https://docs.docker.com/get-docker/) and `make`.

Once these are available, simply execute the Docker test target to run all tests: `make test-docker`. This command can also be used to open an interactive shell into the container where this library is installed. To start a \*bash\* shell for example, use this command: `command=bash make test-docker`.

<a name="style-guidelines-and-naming-conventions"></a>

## Style Guidelines & Naming Conventions

Generally, we follow the style guidelines as suggested by the official language. However, we ask that you conform to the styles that already exist in the library. If you wish to deviate, please explain your reasoning. In this case, we generally follow the [C# Naming Conventions](https://msdn.microsoft.com/library/ms229045(v=vs.100).aspx), the suggestions provided by the Visual Studio IDE and StyleCop (see the [stylecop.json](https://github.com/sendgrid/sendgrid-csharp/blob/master/src/SendGrid/stylecop.json) and [SendGrid.ruleset](https://github.com/sendgrid/sendgrid-csharp/blob/master/src/SendGrid/SendGrid.ruleset) configuration files.

<a name="creating-a-pull-request"></a>

## Creating a Pull Request

1. [Fork](https://help.github.com/fork-a-repo/) the project, clone your fork,

and configure the remotes:

```bash

# Clone your fork of the repo into the current directory

git clone https://github.com/sendgrid/sendgrid-csharp

# Navigate to the newly cloned directory

cd sendgrid-csharp

# Assign the original repo to a remote called "upstream"

git remote add upstream https://github.com/sendgrid/sendgrid-csharp

```

2. If you cloned a while ago, get the latest changes from upstream:

```bash

git checkout <dev-branch>

git pull upstream <dev-branch>

```

3. Create a new topic branch off the `development` branch to

contain your feature, change, or fix:

```bash

git checkout -b <topic-branch-name> development

```

4. Commit your changes in logical chunks. Please adhere to these [git commit

message guidelines](http://tbaggery.com/2008/04/19/a-note-about-git-commit-messages.html)

or your code is unlikely to be merged into the main project. Use Git's

[interactive rebase](https://help.github.com/articles/interactive-rebase)

feature to tidy up your commits before making them public.

4a. Create tests.

4b. Create or update the example code that demonstrates the functionality of this change to the code.

5. Locally merge (or rebase) the upstream `development` branch into your topic branch:

```bash

git pull [--rebase] upstream development

```

6. Push your topic branch up to your fork:

```bash

git push origin <topic-branch-name>

```

7. [Open a Pull Request](https://help.github.com/articles/using-pull-requests/)

with a clear title and description against the `development` branch. All tests must be passing before we will review the PR.

<a name="code-reviews"></a>

## Code Reviews

If you can, please look at open PRs and review them. Give feedback and help us merge these PRs much faster! If you don't know how, Github has some great information on how to review a Pull Request.