## <a id="how-to-contribute">How to contribute?</a>

Your contributions to Humanizer are very welcome.

If you find a bug, please raise it as an issue.

Even better fix it and send a pull request.

If you like to help out with existing bugs and feature requests just check out the list of [issues](https://github.com/Humanizr/Humanizer/issues) and grab and fix one.

Some of the issues are labeled as as `jump in`. These issues are generally low hanging fruit so you can start with easier tasks.

This project has adopted the code of conduct defined by the [Contributor Covenant](http://contributor-covenant.org/)

to clarify expected behavior in our community.

For more information see the [.NET Foundation Code of Conduct](http://www.dotnetfoundation.org/code-of-conduct).

### <a id="getting-started">Getting started</a>

This project uses C# 8 language features and SDK-style projects, so you'll need any edition of [Visual Studio 2019](https://www.visualstudio.com/downloads/download-visual-studio-vs) to open and compile the project. The free [Community Edition](https://go.microsoft.com/fwlink/?LinkId=532606&clcid=0x409) will work.

### <a id="contribution-guideline">Contribution guideline</a>

This project uses [GitHub flow](http://scottchacon.com/2011/08/31/github-flow.html) for pull requests.

So if you want to contribute, fork the repo, preferably create a local branch, based off of the `master` branch, to avoid conflicts with other activities, fix an issue, run build.cmd from the root of the project, and send a PR if all is green.

Pull requests are code reviewed. Here is a checklist you should tick through before submitting a pull request:

- Implementation is clean

- Code adheres to the existing coding standards; e.g. no curlies for one-line blocks, no redundant empty lines between methods or code blocks, spaces rather than tabs, etc. There is an `.editorconfig` file that must be respected.

- No ReSharper warnings

- There is proper unit test coverage

- If the code is copied from StackOverflow (or a blog or OSS) full disclosure is included. That includes required license files and/or file headers explaining where the code came from with proper attribution

- There are very few or no comments (because comments shouldn't be needed if you write clean code)

- Xml documentation is added/updated for the addition/change

- Your PR is (re)based on top of the latest commits from the `dev` branch (more info below)

- Link to the issue(s) you're fixing from your PR description. Use `fixes #<the issue number>`

- Readme is updated if you change an existing feature or add a new one

- Run either `build.cmd` or `build.ps1` and ensure there are no test failures

Please rebase your code on top of the latest `dev` branch commits.

Before working on your fork make sure you pull the latest so you work on top of the latest commits to avoid merge conflicts.

Also before sending the pull request please rebase your code as there is a chance there have been new commits pushed after you pulled last.

Please refer to [this guide](https://gist.github.com/jbenet/ee6c9ac48068889b0912#the-workflow) if you're new to git.

### <a id="need-your-help-with-localisation">Need your help with localisation</a>

One area where Humanizer can always use your help is localisation.

Currently Humanizer supports quite a few localisations for `DateTime.Humanize`, `TimeSpan.Humanize`, `ToWords` and `ToOrdinalWords`.

Humanizer could definitely do with more translations.

To add a translation for `DateTime.Humanize` and `TimeSpan.Humanize`,

fork the repository if you haven't done yet, duplicate the [resources.resx](https://github.com/Humanizr/Humanizer/blob/master/src/Humanizer/Properties/Resources.resx) file, add your target [locale code](http://msdn.microsoft.com/en-us/library/hh441729.aspx)

to the end (e.g. resources.ru.resx for Russian), translate the values to your language, register your formatter in [FormatterRegistry.cs](https://github.com/Humanizr/Humanizer/blob/master/src/Humanizer/Configuration/FormatterRegistry.cs), write unit tests for the translation, commit, and send a pull request for it. Thanks.

Some languages have complex rules when it comes to dealing with numbers; for example, in Romanian "5 days" is "5 zile", while "24 days" is "24 de zile" and in Arabic "2 days" is "?????" not "2 ???".

Obviously a normal resource file doesn't cut it in these cases as a more complex mapping is required.

In cases like this, in addition to creating a resource file, you should also subclass [`DefaultFormatter`](https://github.com/Humanizr/Humanizer/blob/master/src/Humanizer/Localisation/Formatters/DefaultFormatter.cs) in a class that represents your language;

e.g. [`RomanianFormatter`](https://github.com/Humanizr/Humanizer/blob/master/src/Humanizer/Localisation/Formatters/RomanianFormatter.cs) and then override the methods that need the complex rules.

We think overriding the `GetResourceKey` method should be enough.

To see how to do that check out `RomanianFormatter` and `RussianFormatter`.

Then you return an instance of your class in the [`Configurator`](https://github.com/Humanizr/Humanizer/blob/master/src/Humanizer/Configuration/Configurator.cs) class in the getter of the [Formatter property](https://github.com/Humanizr/Humanizer/blob/master/src/Humanizer/Configuration/Configurator.cs) based on the current culture.

Translations for `ToWords` and `ToOrdinalWords` methods are currently done in code as there is a huge difference between the way different languages deal with number words.

Check out [Dutch](https://github.com/Humanizr/Humanizer/blob/master/src/Humanizer/Localisation/NumberToWords/DutchNumberToWordsConverter.cs) and

[Russian](https://github.com/Humanizr/Humanizer/blob/master/src/Humanizer/Localisation/NumberToWords/RussianNumberToWordsConverter.cs) localisations for examples of how you can write a Converter for your language.

You should then register your converter in the [ConverterFactory](https://github.com/Humanizr/Humanizer/blob/master/src/Humanizer/NumberToWordsExtension.cs#L13) for it to kick in on your locale.

Don't forget to write tests for your localisations. Check out the existing [DateHumanizeTests](https://github.com/Humanizr/Humanizer/blob/master/src/Humanizer.Tests.Shared/Localisation/ru-RU/DateHumanizeTests.cs), [TimeSpanHumanizeTests](https://github.com/Humanizr/Humanizer/blob/master/src/Humanizer.Tests.Shared/Localisation/ru-RU/TimeSpanHumanizeTests.cs) and [NumberToWordsTests](https://github.com/Humanizr/Humanizer/blob/master/src/Humanizer.Tests.Shared/Localisation/ru-RU/NumberToWordsTests.cs).