Translation tool

`translation-tool` reads all keys from source language, finds differences to each target file and differences to previous commit of source. Changed translations are then retrieved from `https://www.deepl.com/translator` and stored for each target language(s). The order of existing keys is preserved, but new keys are added at the end of the target file.

Function of the translation tool

The tool is controlled with four parameters. An example call could look like:

```
translation-tool -s en -t de -p "translations" -r .
```

If you want to use the tool with standard values, you can start it without parameters.

1. The parameter `-s` or `--source` specifies the source language. If the parameter is omitted, `en` is used as source language.
2. The parameter `-t` or `--target` specifies the target language. If you omit it, translation will be done for all languages that are in the same parent folder as the source language.
3. The `-p` or `--path` parameter specifies the parent folder. If the parameter is omitted, `translations` is assumed as the default path.
4. The repository is set with parameter `-r` or `--repo`. The parameter is required if the tool is started outside of its main directory. If omitted, the current directory is assumed as the repository root.

There are two more parameters, but they are only needed to display information.

- With the parameter `-h` or `--help` the tool provides a short help on how to use it.
- The parameter `-v` or `--verbose` outputs some information on the command line during the translation process.

Use Cases

- Adding a new language
    1. Create a directory with its country code (e.g. it for itallian) in `translations`
    2. Create the file `main.json` in the directory you just created
    3. The file `main.json` must contain at least `{}`

TODO's

- [ ] Extract DeepL translation key into configuration file
- [x] Update PDF Documentation
- [x] Add Java-Doc for Classes
- [ ] Batch translations in packages of 50
- [ ] Use translations from Google, if language is not supported by DeepL
- [ ] Word count and changes count
- [ ] Commit to Git after running the `translation-tool`
- [ ] Handling of codes like `en-us` or `pt-br` as target language (Mapping? Needs to be defined)

Embed into IntelliJ

1. In IntelliJ in the gradle window (View -> Tools Window -> Gradle) add the translation tools directory. After that, all Java files should be recognized. You can start the tool with a run configuration.
2. Adjust the run configurations for different use cases, e.g. just translate from English to German

Call via command line / pipeline

1. If you want to start the tool from the command line, you have to build a FatJar with all the required libs first. For this purpose there is a task named `shadowJar` in the `translation-tool`. The FatJar will be built with `gradlew shadowJar`. (Gradle must have a version of 7.0+)
2. With the command `./gradlew translationTool` the tool is started with default values.