Developer Guidelines (Simple)

The following coding guidelines must be adhered to whenever you write some source code, however much or little it is (even if it's only a small assignment).


```
For assignment 1, don't worry
about exception handling and
testing
```

Coding Style

Do not Hardcode Anything

Do not hardcode anything - no variables, no error messages, nothing.

- Bad: "Please visit www.mr-dlib.org for more details"
- Good: "Please visit" + project_url +"for more details".

- Bad :"The maximum file size is 256 MB".
- Good: "The maximum file size is " + max_file_size.

In particular, never ever hardcode a path such as "C:\windows\temp\" but use system variables instead, e.g. System.getProperty("java.io.tmpdir").

Use a Single Point of Definition

All variables (e.g. URLs, default values, …) must be defined in one place so it's easy to find and modify them. This "one" place can be either a separate config file, the beginning of a class, or in the beginning of a method, but not somewhere in the middle of a method.

- "Global" variables, i.e. variables someone else might want to change, and that hence need to be easily found: store in a config file. Examples include a path to a dataset, the version/build number, the name of the project, or the URL to the project's homepage
- Variables being needed by different classes: store in a config file if possible.
- Variables being needed only within one class (by different methods): store in the beginning of the class
- Variables being needed only within one method: store in the beginning of the method
- Temporary variables such as

```
for(int i=1; i<maxLength; i++) ...
```

  store wherever needed.

Avoid Redundant Source Code

There must be no redundant code (copy&paste) anywhere. If you need the same code twice, write a new method that can deal with both situations.

Choose Meaningful Names of Variables, Methods etc.

Variables, methods, classes etc. must have meaningful names.

- Bad: *button1*
- Good: *buttonAboutDialogClose*.

- Bad: *float p, r, f*
- Good: float *precision, recall, fvalue*

Be Consistent in Naming

Be consistent with naming your variables and method names, i.e. don't name one variable "max_file_size" and another one "minimumSizeOfFile".

Write Short Methods (few LOC)

A method should usually not contain more than 30 or 40 lines of code. More than 100 is a definite no-go.

Use Operating-System Independent File Paths

Never use \ or / when specifying a path, if there is a way to use an OS-independent path separator.

Error handling (and prevention)

Catch (and handle) exceptions that might occur

For instance, if your tool writes some data to the disk: Handle if the disk is full or write protected. If your tool reads files or folders: Handle 0KB, corrupt files, and non-existent files. Check if the file does contain the content that you expect (e.g. BibTeX keys, or list of keywords).

Provide meaningful error messages to the user

- Worst error message: "java.lang.NullPointerException ...".
- Bad error message: "fileNotFoundException...."
- Acceptable error message: "The PDF could not be converted"
- Good error message: "The PDF" + filename + "could not be converted because: " + error_text +"--" + error_code.

Use Standard HTTP Error Codes for Web Development

The HTTP standard is specifying a number of error types and error codes. Please, especially when developing the Mr. DLib Web Service, use the standard HTTP error codes.

Do thorough testing

You are responsible for the quality of your program. Before giving the program to a user or other developers, test it. Ideally on a machine different from the one you developed it on.

Never change the original data

Never ever change the original data. Instead, create temporary data that you work with. For instance, have the original data in /data/original/dataset.csv, and the "working data" in /data/pruned/2017-08-13_dataset_pruned.csv. If there is data that is only temporarily needed, and that can be deleted by anyone anytime, store it e.g. in /data/temp/.

Use Proper Encoding

Ensure that the proper encoding of special characters such as ä ö ü ß © and ensure to escape characters such as / or \ properly if necessary.