PROJECT-SCOPES DESIGN CONVENTIONS

by MicroScopes

Contents

1 Comments rules

General rule that refers to all cases is that any line cannot be longer than 80 characters.

1.1 Doxygen comments

In order to make the project well documented doxygent tool is used. The following rules must be followed to keep the code clear and readable.

- Doxygen comments are used to document class-files, classes, public methods and public attributes.
- First line of the doxygen comment starts with `/*!` and then is followed by new line. All other lines, until the penultimate, contain `*` preceded with one whitespace. Last line closes the comment with `*/` in front of which one whitespace occurs.

```
/*!
 * ... text ...
 */
```

- All doxygen commands are allowed to be used and must start with `@`. For more information please refer to `https://www.stack.nl/~dimitri/doxygen/manual/commands.html`.
- The minimum number of commands that must be used depends on the comment type and is as follows:
  - in case of class-file: `@file`, `@brief`, `@author`,
  - in case of class definition: `@brief`, `@details`,
  - in case of public attributes: `@brief`,
  - in case of public methods: `@brief`, `@details`, `@param` (if applicable), `@return` (if applicable).
- Each doxygen command is followed by one whitespace and then comment message.
- Between `@brief` and `@details` commands there is always one comment line break.

```
/*!
 * @brief ...
 *
 * @details ...
 */
```

1.2 Non-Doxygen comments

For all the comments that are not included in the category of doxygen comments single line `C#` comment style is used. This rule applies to multiline comments as well. Each comment line starts with `//` followed by one whitespace.

```
// ... text ...
// ... text ...
```

This kind of comment is used to document new section of code, protected and private methods and protected and private attributes.

1.3 Files

Each file that contains a class definition should begin with a doxygen comment with the following information:

- name of the file in which it is located (the same as the class name),
- brief information of what it contains,
- name of the author (e.g. team name) that created the file.

```
/*!
 * @file ...
 * @brief ...
 * @author ...
 */
```

1.4 General information

General commets are apllied to indicate that a new section of code begins. Each general type comment consists of 3 lines. First and last line are identical and starts with `//` followed by 50 `=`. Middle line contains information of what type of section it opens. The information shall be written in capital letters only separated by one whitespace. It is also centered relative to first and last line. There are 3 sections which can be considered as separate parts of code:

- directives,

```
//================================================
//              D I R E C T I V E S
//================================================
```

- namespaces,

```
//================================================
//               N A M E S P A C E
//================================================
```

- classes.

```
//================================================
//                   C L A S S
//================================================
```

1.5 Classes

Each class must be commented with doxygen comment type that contains two mandatory commands: `@brief` and `@details`.

```
/*!
 * @brief ...
 *
 * @details ...
 */
```

1.5.1 Methods

Public methods must be documented with doxygen comment type that contains two mandatory commands: `@bried` and `@details`. If the return type of the method is different than `void` there must be `@return` command used to explain the importance of the return value. Additionaly, if the method takes a parameter, it must be documented with `@param` command.

```
/*!
 * @brief ...
 *
 * @details ...
 * @param ...
 * @param ...
 * @return ...
 */
```

Protected and private methods are commented with non-doxygent comment style, but parameters and return value must be taken into account.

1.5.2 Attributes

For all public attributes class rules are taken into consideration. In case of protected and privated attributes non-doxygen comment style is used.

2 Naming conventions

The overall style used in the project is consistent with the `C#` language style conventions. For detailed information please refer to `https://msdn.microsoft.com/en-us/library/ms229043.aspx`. There are some additional rules on files and classes that are described below.

2.1 Files

The `C#` file name and the name of the class contained has to be matched. Each file must contain only one class definition (does not apply to private classes).

2.2 Classes

Each class name should start with the project name abbrieviation, i.e. PS for Project-Scopes. It must be then followed by proper class name (without any separator between).

```
class PSClassName { ... }
```

3 Style guideline

3.1 Braces

To improve comprehention of the code, all braces are located in new line. It applies to both single and multilnie instructions.

```
if (...)
{
        ...
}
else
{
        ...
}
```

```
for (...)
{
        ...
        ...
        ...
}
```