コーディング規約

制作者：槌田駿

制作日：2019/11/21

最終更新日：2020/02/03

* 目的
* プロジェクトの構成
* 命名規則
* コーディングスタイル
* 禁止事項
* 制限事項
* 推奨事項

目的

この文書は、セーリングリポジトリ内のコーディングにおいて留意すべき事項にまとめたものである。

ほかに書くことがないので、コーディング規約のメリット・デメリットを書き出しておきます。

意味を知っていたほうがやる意味を見出せるので。

メリット

・ソースコードの可読性・バグの防止

人によって変わる変数のつけ方などの癖をなくし、統一かすることで、ほかの人も読みやすく、また、癖がなくなるため、変数のつけ方によるバグなどを減らせます。

デメリット

・ソースコードに個性がなくなる

・生産性が落ちる（制限のかけ方による）

書き方の制限を行うため、プログラマーの個性（変数のつけ方など）が消えます

プロジェクトの構成

Project

ーAssets

｜ーAnimation

｜ーEdit

｜ーFont

｜ーImage

｜ーMaterials

｜ーModels

｜ーPhoton

｜ーPrefabs

｜ーResources

｜ ｜ーPrefabs

｜ ｜ーScriptable

｜ ｜ーSounds

｜ーScenes

｜ーScripts

｜ーStoreAssets

ーDocument

ーKey

ーその他（.gitignoreで省かれています）

.mataファイルも忘れずにコミットすること。

（→[Unityの.metaファイルとは何なのか？](https://qiita.com/4_mio_11/items/91ebcdd30398af607373)）

ソースコード｜の先頭につけるコメント

\* @date日付（開始日？）

\* @author書いた人

\* @brief簡単な説明

\* 変更履歴

\* ：日付

\* ：内容

\*/

命名規則

|  |  |
| --- | --- |
| **種類** | **Case** |
| クラス | Pascal *（例：ShipObject）* |
| インタフェース | "I" + Pascal *（例：IList）* |
| パッケージ / ネームスペース | Pascal *（例：Sailing.Ship）* |
| メソッド | Pascal *（例：ToString()）* |
| ローカル変数 | Camel *（例：refStr）* |
| メンバ変数 | Camel *（例：Calendar.isGoal）* |
| 定数 | Snake *（例：Integer.MAX\_VALUE）* |
| 列挙定数 | Snake *（例：State.POOR）* |

CamelCase

・最初の単語はすべて小文字  
・それ以降の単語は頭だけ大文字、頭以外は小文字

・例：fooBar

PascalCase

・単語の頭は大文字、頭以外は小文字  
・例： FooBar

SnakeCase

・単語はすべて大文字

・単語と単語の間にアンダースコアを挿入  
・例：FOO\_BAR

参考　<https://yujisoftware.hatenablog.com/entry/20110529/1306676847>

コーディングスタイル

特に指定なし。

禁止事項

アノテーションコメント

|  |  |
| --- | --- |
| **TODO:** | あとで追加予定、修正するべき部分 |
| **HACK:** | 無理やり実装されている。リファクタリングが必要 |
| **WARNING:** | 他人のソースを理解せず使用している |
| **REVIEW:** | 動作未確認 |
| **MEMO:** | ソースの意図や使い方、意味を記述や不安事項 |
| **OPTIMIZE:** | もう少しスマートに書けそう |

参考文書がある場合は必ず記載すること

[http://www.aerith.net/design/commentーj.html](http://www.aerith.net/design/comment-j.html)

https://qiita.com/tadnakam/items/5d1280559eb75b29847c