コーディングルール

2024/5/13

チーム制作応用におけるコーディングルールについて、大事そうな部分+独自のルールをまとめる。基本は https://qiita.com/Ted-HM/items/67eddbe36b88bf2d441d と https://hackmd.io/@afami/SJBoxCyf_を参照。

命名規則

・Pascal / Camel 形式を基本とする。
大文字 / 小文字の違いのみによる区別は避ける。

Pascal 形式: PascalCaseName, IoException, XmlTransfer
Camel 形式: camelCaseName, ioException, xmlTransfer

・以下の識別名は Camel 形式とし、それ以外は Pascal 形式とする。
パラメータ、private なフィールド、ローカル変数
(それ以外： public 変数、メソッド名、unity 上のオブジェクト名 など)

・コメント以外は半角英数のみで記述し、英単語を使用すること。
日本語名が必要な場合は、コメントに含めること。ローマ字綴りは禁止。

・スコープの短い局所変数であれば省略形でも可、それ以外では単語は省略せずに記述すること。
例えば、表記揺れで jpeg, jpg, message, msg, window, win, windows などが混ざることは避ける。

・コレクションリストの変数名は複数形とする。

単数形: File file
複数形: List<File> files, byte[] bytes

・bool 型について、否定系の名前は避ける。
例： bool disableSsl = false; ×
bool useSsl = false; ○

・時間やバイト数など計測できるものであれば、変数名にアンダースコアで区切り単位を入れる
例：start_ms
　　delay_secs
また、識別のために必要な情報をつける時もアンダースコアで区切り記述する
例：isInputTrigger_L

メソッド(≒関数、サブルーチン、ファンクション)

Pascal 形式とで、動詞とする。
インスタンスを生成するメソッドは Create-や New-を使う。
論理値を返すメソッドには Is-, Can-, Has-を使う。
特別な場合を除き Is+<形容詞>、Can+<動詞>、Has+<過去分詞>とする。
型を変換するメソッドは例外的に To-を使う。

インスタンスを生成するメソッド: CreateInstance, NewInstance
論理値を返すメソッド: IsNull, IsNullable, IsEmpty, CanRead, HasChanged
型を変換するメソッド: ToString, ToInt32

二文字の名前

ID, OK, IP, DB など二文字からなる短い名前について、
Camel 形式では小文字とする。
Pascal 形式では、IP(Internet Protocol), DB(Data Base)など頭字語はすべて大文字とし、ID(Identifier)と OK(Okay)など頭字語ではない省略形は通常の Pascal 形式と同様とし、区別する。

Pascal 形式(MSDN): IdManager, OkButton, IPAddress, DBTable, IOReader, XmlReader
Camel 形式: idManager, okButton, ipAddress, dbTable, ioReader, xmlReader

その他

・if 文と for 文について、{ }(中括弧)は基本省略しないこと。

・メソッドの早い段階での `return` は積極的に行うこと。（ガード節）
・比較演算子は左を主とし、`<, <=` のみ使用すること。
・var は使わない。
・メンバ変数には `m_`をつける。
・アクセス修飾子はちゃんと書く。（private, public 等）
・他オブジェクトをインスタンス化する場合はアンダースコアを最初に付ける。
例：PlayerController　_PlayerController;
・この規約に準拠して可読性が下がる場合は柔軟に対応。（例：コントローラーの LR は小文字だと可読性が低い）