読みやすいソースコードの書き方

izumi.nakamura

読みにくいソースって？

◆ どこに何が書いてあるか分からない

◆ メソッドがやたら長いので、自分の関心のある変数がどこで変更されているか分からない

◆ 分岐や繰り返しが多用されすぎている

◆ 書いた人の意図が不明（コメントが少ない）

◆ etc…

どうすれば読みやすくなる？

- 人間は理解に時間がかかるものを難しいと感じる
- 理解に時間がかかる原因は、**推測と整理**である
- 推測と整理にかかる時間を減らすことで読みやすくなる

よって特に以下の２点を心がけると飛躍的に読みやすくなる

① 意図の分かる変数名と関数名をつける
- 推測にかかる時間を削減

② 関数の見通しをよくする
- 整理にかかる時間を削減

意図の分かる変数名と関数名をつける

■長くなってもいいので分かりやすさを優先する

- IDE（開発環境）の発達によりタイプミスや生産性のデメリットはなくなった。分かりやすさこそ正義。
- 小さいforループなどのインデックス用変数は i でいい。
- 一貫性も大事。プロジェクト間で表記が揺れると混乱の元になる。コーディングルールからは外れてはいけない。（例：物件→bk, bukken, building…）

■クラスは名詞、メソッドは動詞＋目的語

- 「○○を取得する」はget~だが、より具体化した方が分かりやすい。（例：データベースから取得する場合→fetch~、インターネットから取得する場合→download~ など。）
- ネーミングに悩んだら https://codic.jp/

■何でもかんでも英語にしない

- 例えば “FixedTermLease” 何のことか分かりますか？

関数の見通しをよくする

■関数の中身を短くする

- 最重要事項。目安は35行。（関数が1画面内に全て表示可能であることが望ましい）
- 同一粒度で論理的に分割すると非常に見通しがよくなる。（実践で説明する）

■制御のネスト（入れ子構造）を浅くする

- 目安は2段まで。3段以上のネストは普通の人にとっては“難しい”。

■１つの関数で行うのは１つのことだけ

- 関数名以外のことは行わない。（推測の妨げになる）

■無駄なコメントを書かない

- 読み手の立場に立って書く。何をしているか？ではなく、何故そうしたのか？が大事。

■コメントアウトしたソースは消す

- Grepの妨げ、可読性の悪化につながる。履歴はソース管理で確認。

実践してみよう

https://github.com/mshade0314/readable