Introduction to Development

Yu Tokunaga

The primary objective of this document is to facilitate a conceptual comprehension of “What constitutes system development” for individuals lacking IT skills and industry experience. This endeavor is not aimed at presenting efficient methodologies or challenging widely accepted norms; rather, it seeks to methodically consolidate information for those desiring a fundamental grasp of essential points.

Contents

1 Console 3
1.a コンソール 3
1.b ターミナルとシェル 3
1.c コマンド 3
2 Rust 4
2.a 基本的な文法 4
3 Algorithm 4
4 Quality Control 4
5 Git 4
5.a バージョン管理 4
6 Entity Component System 4
7 Web 4
8 Database 4
8.a データベースとは 4
9 Technical Writing 4
Bibliography 4
APPENDIX A: Foo 4
APPENDIX B: Bar 4

1 Console

1.a コンソール

コンソールとは，OS を搭載したコンピュータに接続されたディスプレイおよびキーボードを指す．つまり，コンピュータとの対話を可能にするための物理ハードウェア，広義の入出力システムともいえる．

「文字だけの黒い画面」の実体は，仮想コンソールである． これは入出力を仮想的な空間に提供するものであり，この環境は CUI（Character-based User Interface）とも呼ばれる． CUI は，我々が日常的に利用する GUI（Graphical User Interface）とは対をなす存在であり，文字に基づいたユーザーインターフェースを提供する．

インターフェース

“Interface"という語はハードウェアでもソフトウェアでも登場する．情報産業では主に，異なる二つのモノ（人間や機器）を接触させるための境界面として用いられる語である．

1.b ターミナルとシェル

仮想コンソール（ CUI ）を提供するアプリケーションは一般にターミナルと呼ばれる．Windows におけるコマンドプロンプトや，MacOS における iTerm2 や Warp などがターミナルにあたる． これらは GUI 上で CUI を操作するための窓口になる．

本題となるコンピュータへの命令はコマンドという． コマンドを解釈し，実行する仕組みをシェルという． シェルの役割はコンピュータとの対話を実際に担うアプリケーションであり，Bash や zsh，fish など種類が存在する．

シェル

エンジニアでない一般ユーザが OS と対話するためのシェルに，Windows のエクスプローラーなどがある．CUI のシェルと GUI のシェル，操作しやすいユーザ層は当然異なる．

要するに，エンジニアがコンピュータへ命令するためには基本的に「ターミナルを起動し（シェルを介して）コマンドを入力」する．また，コマンドを入力する行のことをコマンドラインという． 単語が重要なのではなく，この CUI 構成を理解することは今後役に立つ．

1.c コマンド

ここでは，基本的なコマンドを紹介する．細かいオプションまでは解説をしない．なお，ここで使用するコマンドは Windows コマンドプロンプトでは使用できない．

1.c.a ファイルの一覧化： ls

まずはフォルダの内容を確認したい． そのようなときに打つコマンドである． 何回打っても良い．

```
1 ls
```
Command

1.c.b ディレクトリの移動： cd

今いるフォルダから移動したいときに打つ．

```
1 cd {path} // {移動先のパス}へ移動
2 cd ..     //  1つ上のディレクトリへ移動
3 cd ~      //  ホームへ移動
```
Command

コマンドライン

本書では，コマンドラインへの入力表記において{}を代入記号，//をコメントとする．これらは実際に入力せず，可読性を補助する．

1.c.c 現在地 : pwd

プロンプトに書いてあるディレクトリ名をわざわざ絶対パスで表示する． あまり使わないが，重大な作業をしているときに使った記憶がある．

```
1 pwd
```
Command

1.c.d ディレクトリの作成 : mkdir

思ったよりお世話になる． 知っておいて損はない．

```
1 mkdir {ディレクトリの名前}
```
Command

1.c.e ファイルの閲覧 : cat

本来は複数のファイルを結合（ concat ）して表示するコマンドだが，単一のファイルをのぞき見することで使うことが多い．

```
1 cat {ファイル名}
```
Command

1.c.f ページャ : less

本来のファイル閲覧用コマンドだが，大体は cat コマンドで足りてしまう．less を使用した後は，Q キーを押下すると終了する．

```
1 less {ファイル名}
```
Command

1.c.g 容量の確認 : du

「このフォルダは何 MB あるのか」といった疑問を解決する．ちなみに，-hs オプションを付与すると内訳無しで合計値のみ表示するため，ファイル数が異常に多いディレクトリで有効である．

```
1 du {フォルダ名}                    Command
```

1.c.h 検索 : grep

1.c.i 改名 or 移動 : mv

1.c.j 削除 : rm

2 Rust

2.a 基本的な文法

```
1 pub fn main() {                    Rust
2     println!("Hello, world!");
3 }
```

3 Algorithm

4 Quality Control

5 Git

5.a バージョン管理

6 Entity Component System

7 Web

8 Database

8.a データベースとは

9 Technical Writing

Bibliography

APPENDIX A: Foo

APPENDIX B: Bar