Introduction to Development

January 16, 2024

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.

Console

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である。 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、入出力システムともいえる。

「文字だけの黒い画面」というキーワードで一般に想起される画面は、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 ）ともいう。 CUI は、我々が日々利用する GUI（ Graphical User Interface ）と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。 主に情報産業では、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。

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。

ターミナルとシェル (ii)

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。

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。

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

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 (ii)

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。

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
```

コマンド (iii)

ディレクトリの作成 : mkdir

思ったよりお世話になる。 メイクディレクトリと読んでしまうが正解は知らない。 知っておいて損はない。

```
1 mkdir {ディレクトリの名前}
```
Command

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

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。

コマンド (iv)

```
1 less {ファイル名}
```
Command

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を非表示にできるため、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

検索 : grep

ファイルを検索することは多々ある。 よく使うオプションつきで紹介する。

```
1 grep -rn {path} -e ${text}
```
Command

- r : 再帰的にサブディレクトリを含める

コマンド (v)

- n : マッチした行番号を表示
- e : 検索する文字列を指定

改名もしくは移動 : mv

便利だが、個人的にリスクがあると思っているコマンドの１つである。２つめの引数が存在するパスならそこへ１つ目の引数に指定されたファイルを移動させる。 存在しないパスなら、１つ目の引数に指定されたファイルを２つ目の文字列に改名する。

```
1 mv {path} {path} // 改名                    Command
2 mv {path} {rename} // 新しい名前に変更
```

移動先を誤字脱字すると、その文字列が新しい名前となるので注意が必要である。 ただし、-iv オプションを付与すると警告してくれるので必須である。

コマンド (vi)

削除 : rm

mv より危険なコマンドである。 削除をコマンドで実施する場合、取り返しがつかない(=ゴミ箱に移動しない)ため、-iv オプションを付与して警告を有りにする習慣をつけよう。

```
1  rm -iv {対象ファイル}                                    Command
```

正規表現

ワイルドカード

1. `Ctrl+c` キーを用いよ．
2. 別のターミナルを起動し，`ps`，`grep` で `yes` コマンドの PID を調べ，`kill` コマンドを用いよ．
3. `top` コマンドを用いよ．

Coding

コーディング

プログラムを開発する作業全般をプログラミングといい，特にソースコードを書く工程をコーディングという。 コーディングする上でシンタックスとデザインパターンは重要なコアである。

シンタックスとは、プログラミング言語の仕様として定められた構文規則を指す。 デザインパターンについては Wiki[1] を引用する。

> ソフトウェア開発におけるデザインパターンまたは設計パターン（英: design pattern）とは、過去のソフトウェア設計者が発見し編み出した設計ノウハウを蓄積し、名前をつけ、再利用しやすいように特定の規約に従ってカタログ化したものである。
>
> — Wikipedia

[1] https://w.wiki/rvm

Hello, World!

ハローワールドプログラムを観察する。

Rust

```rust
fn main() {
    println!("Hello, world!");
}
```

制御フロー

if

loop

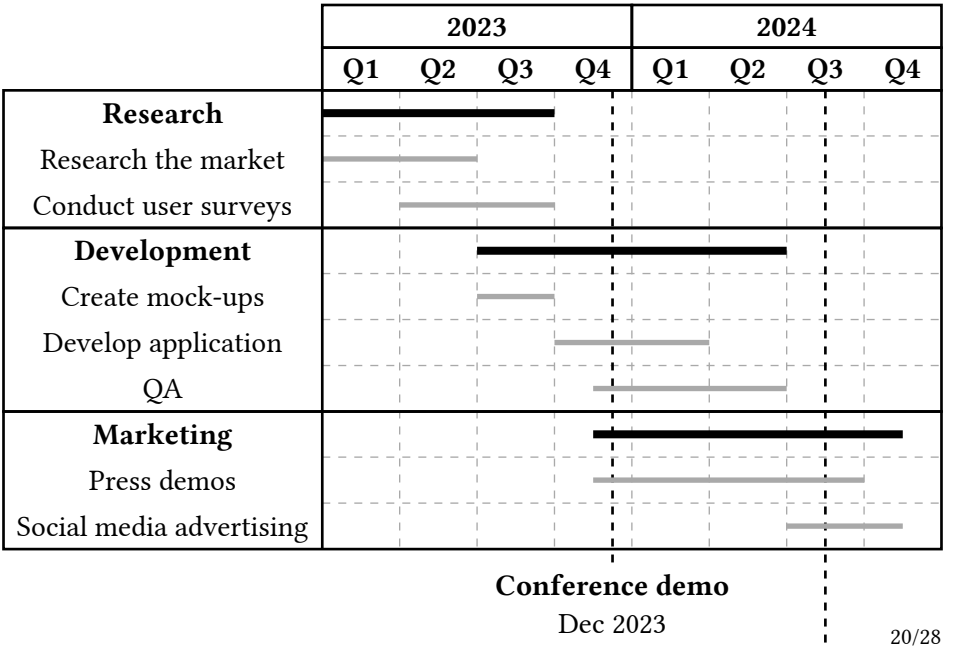
while

for

Algorithm

Quality Control

開発ワークフロー


2023
2024
Q1
Q2
Q3
Q4
Q1
Q2
Q3
Q4
Research
Research the market
Conduct user surveys
Development
Create mock-ups
Develop application
QA
Marketing
Press demos
Social media advertising
Conference demo
Dec 2023
App store launch

Git

分散型バージョン管理システム

Git と GitHub

Git の仕組み

ワーキングディレクトリ

ステージング

リポジトリ

コミット

プッシュ

プル

ブランチ

Entity Component System

Web

沿革

本格的にワールドワイドウェブ（ www ）が普及したのは 1995 年頃だった。 当時は静的な HTML（ HyperText Markup Language ）ページが主流だった。 HTML は情報を構造化し、文書の意味や見出し、段落、リストなどを表現していた。 ページのデザインやスタイルは限定的だったが、これが Web の基盤となり、今日の進化したウェブページの基礎となった。

ユーザはインターネットブラウザを介して Web サーバと通信し、HTML を取得している。 さらに、CSS や JavaScript などがページデザインを制御することにより、インタラクティブな体験を得られている。

Database

Technical Writing