

## 1 目的

情報工学を学ぶ上で、プログラミングや UNIX/Linux/BSD 環境の操作を経験することは避けられません。プログラミングを試してみるだけであれば、Visual Studio<sup>1</sup>や Xcode<sup>2</sup>を導入するのも一つの方法です。しかしこれらのツールは、主に Windows または macOS で作動するアプリケーションを開発することを目的としており、特定のプログラミング言語 (C++, C#, Swift) 以外の開発には適していない場合があります。また、特に研究開発の現場においては、UNIX/Linux/BSD 環境での開発・実行を前提としている場合が多く、このような環境での操作に慣れておく必要もあるでしょう。

macOS は BSD をベースとして開発された OS であり、ターミナル (端末エミュレータ) を立ち上げることで、UNIX/Linux/BSD 環境を体験できます。Windows にはそのような環境が標準で用意されておらず、WSL (Windows Subsystem for Linux, Linux 用 Windows サブシステム) を有効にした上で、UNIX/Linux/BSD 環境を用意する必要があります。

本稿では、Windows 10 を対象とし、WSL を有効にした上で、主要な Linux ディストリビューションの一つである Ubuntu をインストールする方法を説明します。なお、この作業では、多くの通信が発生します。大学の Wi-Fi を利用するなど、通信容量・速度に不安のない環境で行ってください。また、事前に Windows アップデートを実施し、Windows 10 を最新の状態にしてください。Ubuntu のインストールには、5GB 程度の空き容量が必要ですが、Ubuntu に様々なアプリケーションを導入する可能性を踏まえると、10GB 以上の空き容量があることが望ましいでしょう。

## 2 WSL の有効化と Ubuntu のインストール

WSL (Windows Subsystem for Linux, Linux 用 Windows サブシステム) はマイクロソフト社が開発している仮想環境 (のようなもの) です。本稿では、簡略化したインストール手順のみを示すため、公式ドキュメント<sup>3</sup>も読むことをお勧めします。

WSL には、2021 年 3 月現在、WSL 1 と WSL 2 の二つのバージョンがあります。WSL 2 の方が高度な機能を使えるものの、有効化の手順が複雑になるため、本稿では WSL 1 を使います。なお、Windows 10 のプレビュービルド (OS ビルド 20262 以降、開発者向け) では、管理者特権の PowerShell にて、以下のコマンドを実行するだけで WSL 2 + Ubuntu の環境が整う<sup>4</sup>ことから、近い将来、同様に環境構築 (2.1 と 2.2 の作業がコマンド一つで完了) できるようになるでしょう。

Windows PowerShell (管理者)

```
wsl --install
```

### 2.1 WSL 1 の有効化

まずは、WSL を有効化する必要があります。この操作は、GUI または CUI (PowerShell) のいずれかで行えます (どちらか一方のみの作業で十分です)。いずれの作業を行った場合でも、設定を有効にするために Windows の再起動が必要になります。

<sup>1</sup><https://visualstudio.microsoft.com/ja/>

<sup>2</sup><https://developer.apple.com/jp/xcode/>

<sup>3</sup><https://docs.microsoft.com/ja-jp/windows/wsl/>

<sup>4</sup><https://docs.microsoft.com/ja-jp/windows/wsl/install-win10>

GUIであれば、まず、スタート画面から「Windows システムツール」の中にある「コントロールパネル」を左クリックし、「プログラムと機能」を開きます。このページの左側に「Windows の機能の有効化または無効化」のリンクがあるので、これを左クリックします。図1のような画面が表示されるので、「Linux 用 Windows サブシステム」（環境によっては「Windows Subsystem for Linux」と表示されます）にチェックを入れて「OK」を左クリックします。

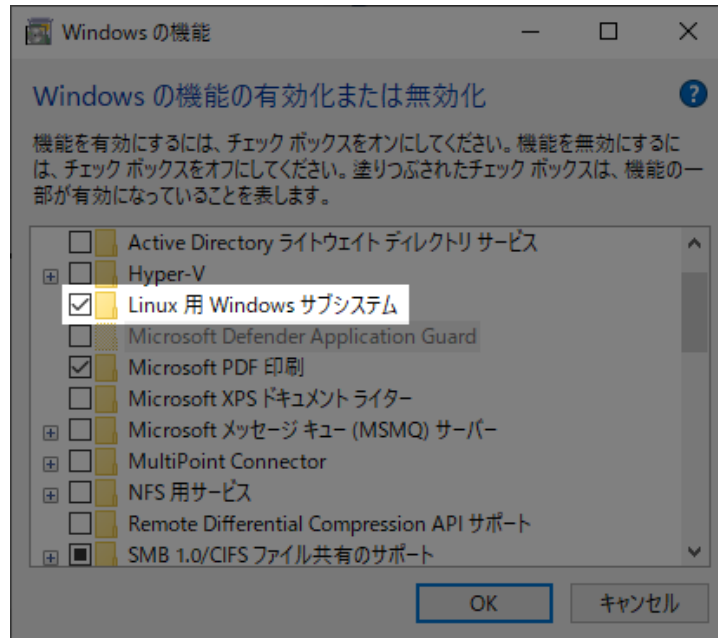


図 1: Windows 機能で「Linux 用 Windows サブシステム」を有効にする

CUIであれば、まず、スタート画面から「Windows PowerShell」の中にある「Windows PowerShell」を右クリックし、「管理者として実行する」を左クリックします。「ユーザアカウント制御」に付いて尋ねられるため、ここで「はい」を左クリックして、Windows PowerShell を管理者権限で立ち上げます。ターミナル上で、以下のように入力し、実行（エンターキーを押す）します（図2）。

Windows PowerShell（管理者）

```
Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux
```

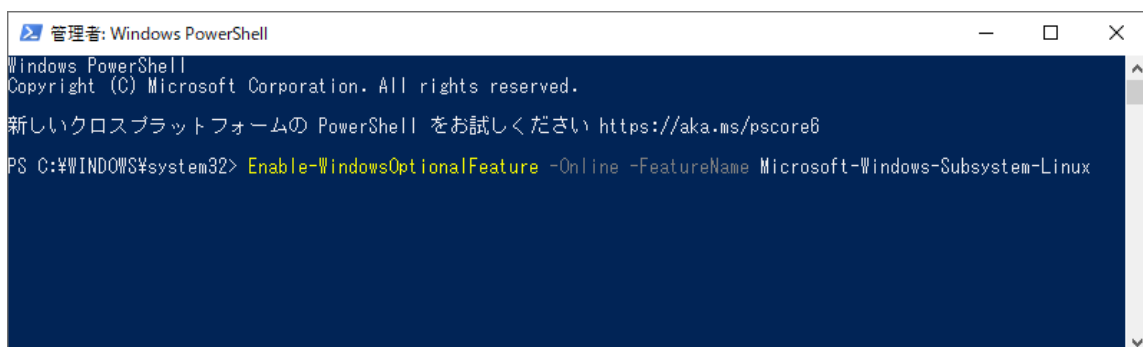


図 2: Windows PowerShell（管理者）によるコマンド実行

GUI または CUI で WSL を有効化した後、設定を反映させるために Windows を再起動します。

## 2.2 Ubuntu のインストール

WSL には様々な Linux ディストリビューションをインストールできますが、本稿では、一般的によく使われている Ubuntu の最新版をインストールします。

まず、スタート画面から「Microsoft Store」を左クリックし、右上の「検索」を左クリックして「Ubuntu」を検索します。図 3 のような画面が表示されるので、「Ubuntu」を左クリックした先のページで「入手」または「インストール」を左クリックします。「サインインする方法」でアカウントを尋ねられた場合、適切なマイクロソフトアカウントでサインイン（ログイン）します。



図 3: Microsoft Store での「Ubuntu」の検索結果

インストールが完了すると、「起動」のボタンが表示されるので、このボタンを左クリックします。スタート画面から「Ubuntu」を左クリックしても、同様に起動できます。Windows アップデートと競合していると、エラーが発生する場合があります。この場合、Windows を再起動して下さい。

## 2.3 Ubuntu の初期設定と作動確認

スタート画面から「Ubuntu」を左クリックし、インストールした Ubuntu を起動します。初回起動時には、図 4 のように、ユーザ名 (username) とパスワードを設定するように促されます。ユーザ名は「user」のように、英数字一単語で設定します。ここで設定するパスワードは、後述する「sudo」を使用する際に入力を求められます。Windows のパスワードと同じにする必要はありません。

ユーザ名やパスワードの設定が終わったら、Ubuntu にインストールされているアプリケーション (パッケージ) を最新にする作業をしましょう。Ubuntu には APT (Advanced Package Tool) と呼ばれるパッケージ管理システムがあり、基本的に、アプリケーションの管理は APT のコマンドを通じて行います。インストールされているパッケージを最新にするには、以下のようにコマンドを入力します (図 5)。「\$」は一般ユーザ権限での実行を表すので、実際には入力不要です (画面上であらかじめ表示されています)。

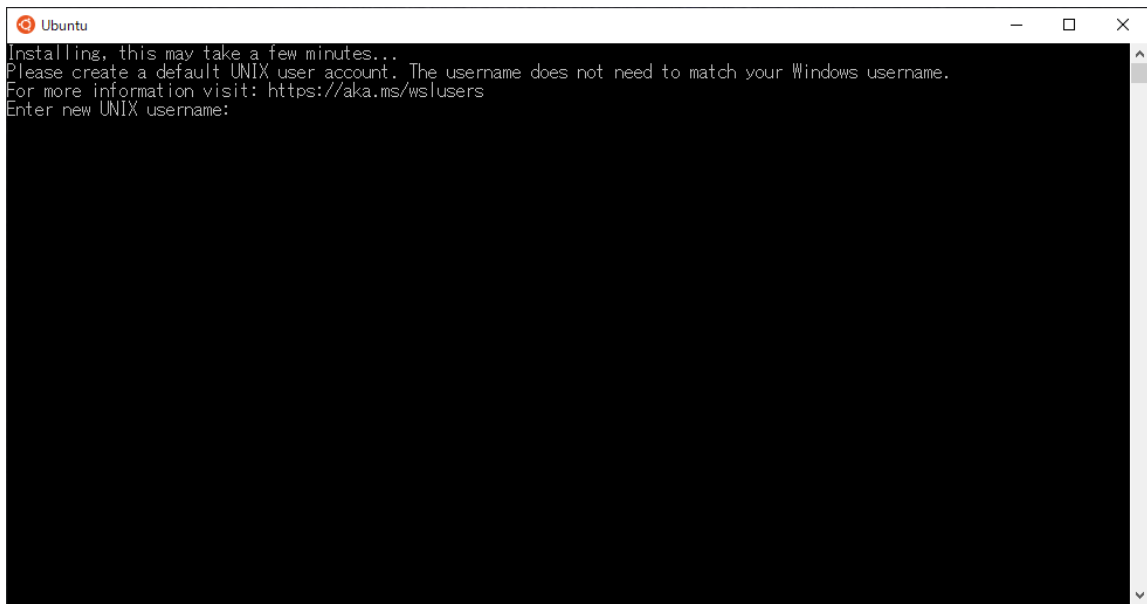


図 4: Ubuntu の初期画面

#### パッケージのアップデート

```
$ sudo apt update  
$ sudo apt upgrade
```

「sudo」は後ろに続くコマンドを管理者権限（root）で実行するためのコマンドです。「sudo」を実行する際には、「[sudo] password for XXX:」とユーザのパスワードが求められる場合があります。この場合、初回起動時に設定したパスワードを入力します。「apt upgrade」を実行すると、「Do you want to continue?」と尋ねられるので、エンターキーを押します（キャンセルする場合は「n」を入力してエンターキーを押します）。

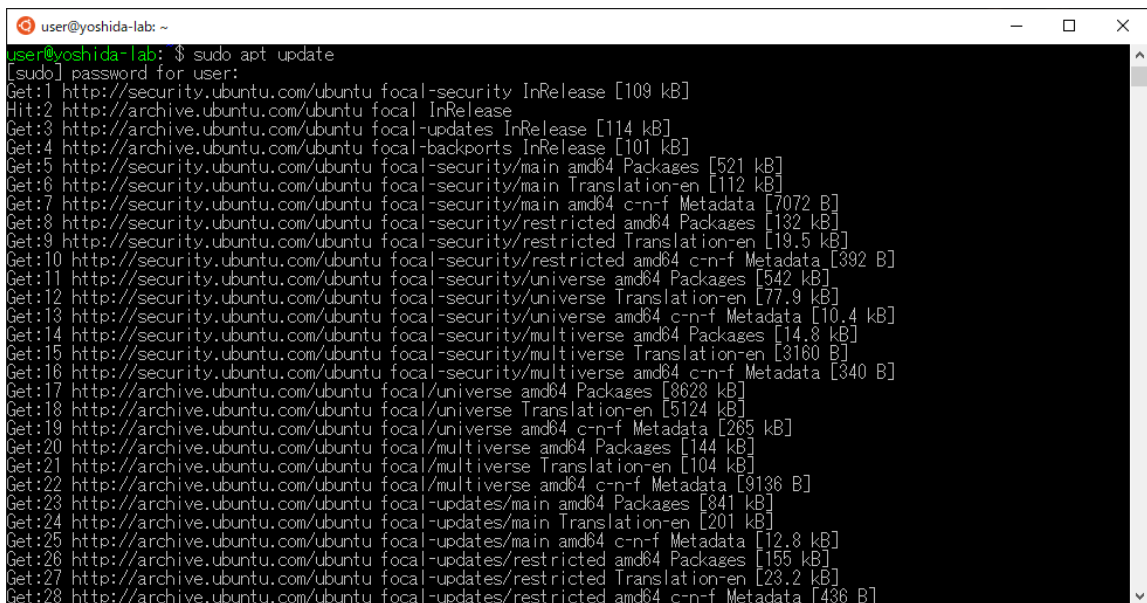


図 5: sudo apt update を実行した例

「apt update」は、インストール可能なパッケージの一覧を更新します。このコマンドだけでは、パッケージのインストールやアップグレードなどは行われません。「apt upgrade」は、インストール済みのパッケージを最新のものに更新します。この処理は、ローカルにあるパッケージの一覧を元に行われるため、パッケージを更新する際は、先に「apt update」を実行しておく必要があります。

## 2.4 Windows から Ubuntu のファイルにアクセスする方法

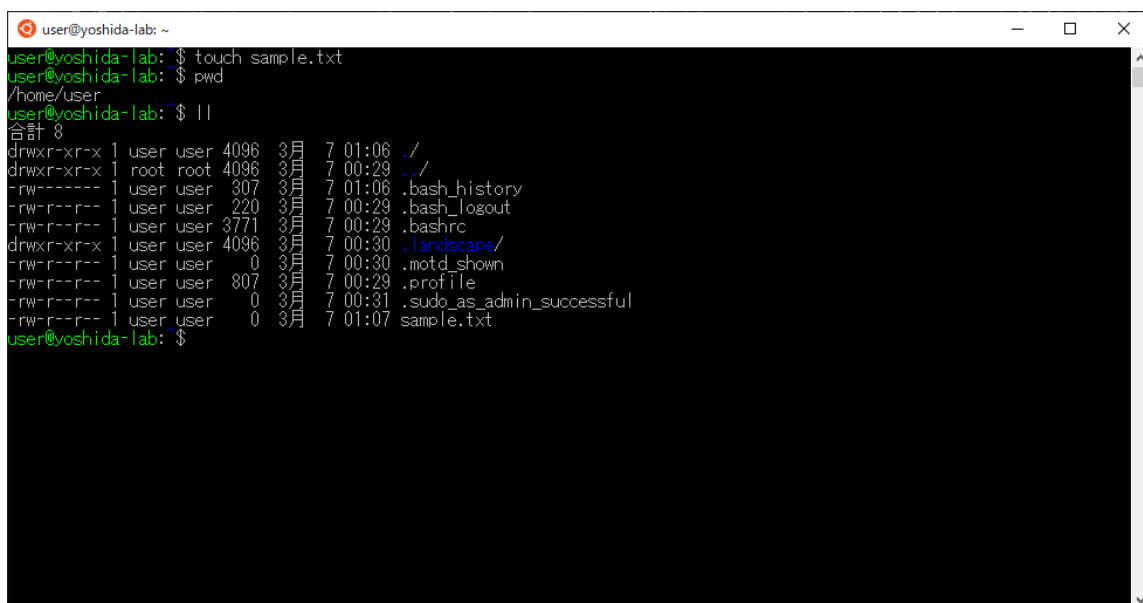
レポートの提出など、Ubuntu で作成したファイルを Windows から参照したい場合があると思います。Windows からネットワークドライブ「\\wsl\$」にアクセスすることにより、WSL 環境上の Ubuntu のファイルにアクセスできます。「\」はバックスラッシュと呼ばれる記号で、Windows の場合は「¥」のように表示される場合があります。また、半角の「¥」を入力すれば、「¥」のように表示されていても、バックスラッシュ（\）として機能します。

Windows からファイルにアクセスできることを確認するために、まずは、以下を実行して空ファイルを作成しましょう（図 6）。

空ファイルの作成と確認

```
$ touch sample.txt
$ pwd
$ ll
```

「touch」は空ファイルを作成するコマンドで、今回は sample.txt という空ファイルを作成しています。「pwd」は現在のディレクトリ（カレントディレクトリ）のパスを表示し、「ll」は現在のディレクトリにあるファイルの一覧を表示します。



```
user@yoshida-lab: ~
user@yoshida-lab: $ touch sample.txt
user@yoshida-lab: $ pwd
/home/user
user@yoshida-lab: $ ll
合計 8
drwxr-xr-x 1 user user 4096 3月 7 01:06 ./
drwxr-xr-x 1 root root 4096 3月 7 00:29 ../
-rw-r--r-- 1 user user 307 3月 7 01:06 .bash_history
-rw-r--r-- 1 user user 220 3月 7 00:29 .bash_logout
-rw-r--r-- 1 user user 3771 3月 7 00:29 .bashrc
drwxr-xr-x 1 user user 4096 3月 7 00:30 .landscape/
-rw-r--r-- 1 user user 0 3月 7 00:30 .motd_shown
-rw-r--r-- 1 user user 807 3月 7 00:29 .profile
-rw-r--r-- 1 user user 0 3月 7 00:31 .sudo_as_admin_successful
-rw-r--r-- 1 user user 0 3月 7 01:07 sample.txt
user@yoshida-lab: $
```

図 6: 空ファイルの作成と確認

次に、Windows エクスプローラーを立ち上げ、図 7 のようにアドレスバーに「\\wsl\$」を入力してエンターキーを押します。エクスプローラーの立ち上げ方が分からない場合、スタート画面から「Windows システムツール」の中にある「エクスプローラー」を左クリックして下さい。

「Ubuntu」というネットワークドライブが見えるので、これをダブルクリックします。その後、Ubuntu の「pwd」のコマンドで表示されたパス（例：「/home/user」）を辿ります。具体的には、「home」をダブ

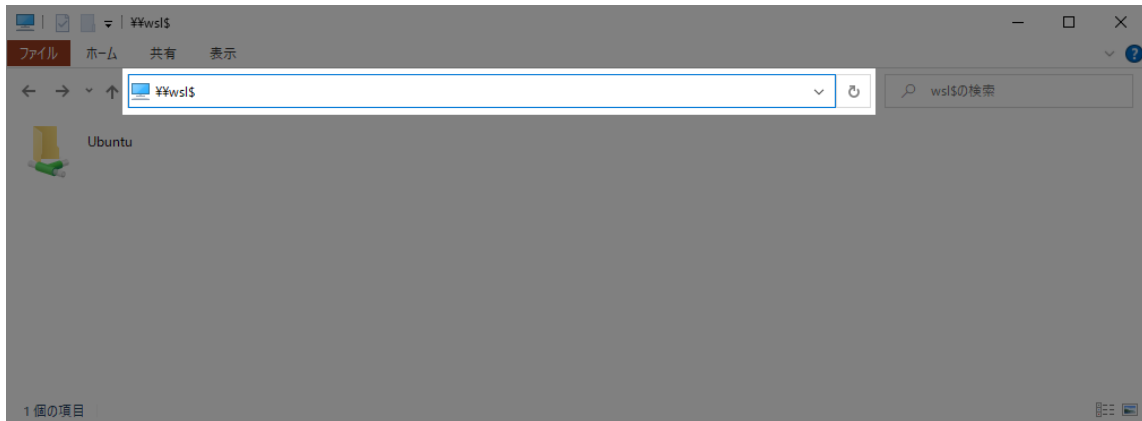


図 7: Windows エクスプローラーのアドレスバーに「\\wsl\$」を入力

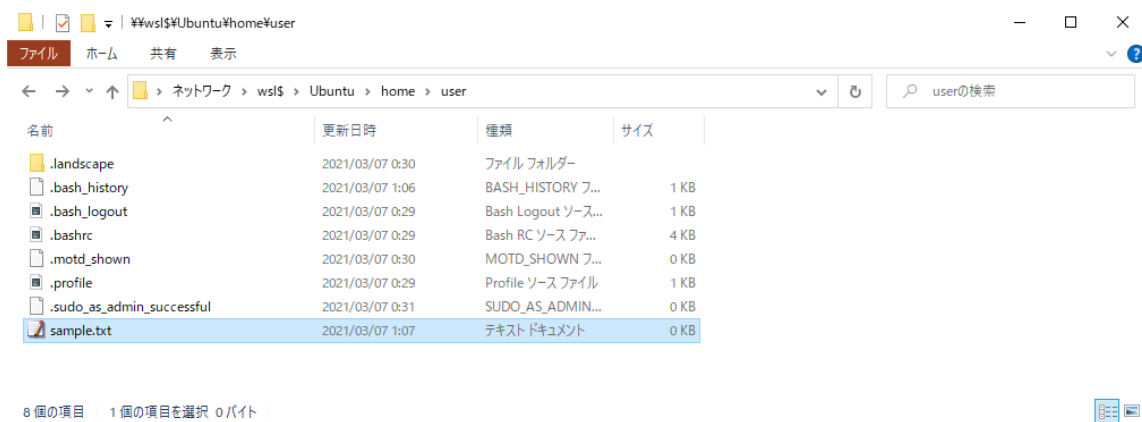


図 8: Ubuntu → home → user とディレクトリを辿った例

ルをクリックし、そして「user」をダブルクリックします。すると、図 8 のように、Ubuntu の「ll」のコマンドで表示されたファイルの一覧が表示されます。Ubuntu で作成・編集した sample.txt は Windows でアクセスできますし、Windows で sample.txt を編集した場合、その結果は Ubuntu にも反映されます。つまり、Windows と Ubuntu でファイルが共有されています。

## 2.5 Ubuntu のアンインストール

Ubuntu のアンインストールは、ほかの Windows アプリケーション同様に、設定の「アプリと機能」で行えます（図 9）。Ubuntu の環境が不要になった場合や環境を作り直す場合には、Ubuntu をアンインストールすると良いでしょう。ただし、Ubuntu の中で作成したファイルも削除されるため、重要なファイルについては、2.4 で説明したように、Windows から Ubuntu のファイルにアクセスし、別のディレクトリにコピーして下さい。

## 3 Ubuntu の設定

初期状態の Ubuntu には、最低限のアプリケーション（パッケージ）しか入っていないため、プログラムを開発する環境としては不十分です。まずは、プログラムを開発するのに必要な最低限のパッケージ（ビルドツール）をインストールしましょう。以下のようにコマンドを入力します。



図 9: Ubuntu のアンインストール

ビルドツールのインストール

```
$ sudo apt update
$ sudo apt install build-essential
```

このように、「apt install XXX」と実行することで、様々なパッケージを自動でインストールできます<sup>5</sup>。このインストール処理は、ローカルにあるパッケージの一覧を元に行われるため、インストールの際は、先に「apt update」を実行しておく必要があります。

以下を実行し、apt コマンドのマニュアルを表示してみましょう。

apt のマニュアルを確認

```
$ man apt
```

Ubuntu のコマンドで分からないことがあれば、まずは man コマンドでマニュアルを読みましょう (q を押すことで終了できます)。インターネット上の解説は、コマンドのバージョンが異なるなどにより、正確でない場合もあります。

初期状態の Ubuntu では、ロケールが「C.UTF8」となっており、表示されるメッセージなどは英語です。このままでも問題はありますが、ロケールを「ja\_JP.UTF8」にし、表示されるメッセージなどを日本語にしたい場合は、以下のようにコマンドを入力します。

<sup>5</sup><https://packages.ubuntu.com/ja/>

日本語言語パックのインストールとロケールの変更

```
$ sudo apt update
$ sudo apt install language-pack-ja
$ sudo update-locale LANG=ja_JP.UTF8
```

Ubuntu の再起動が必要になるため、以下のようにコマンドを入力し、Ubuntu を終了させます。仮想環境上の Ubuntu の再起動（一度終了させてからもう一度起動する）では、Windows を再起動する必要はありません。

Ubuntu の終了

```
$ exit
```

ロケールを「ja\_JP.UTF8」にしたならば、Ubuntu を再起動した後、もう一度 apt コマンドのマニュアルを表示してみましょう。

apt のマニュアルを確認

```
$ man apt
```

先ほどは英語のマニュアルが表示されましたが、今回は日本語のドキュメントが表示されるはずです。また、同じコマンドを入力する場合、矢印キーの上（↑）を押すことで、過去のコマンドを辿ると、手間を省ける場合もあります。

以下では、先のビルドツール（build-essential）に加えて、授業で必要になる可能性のあるアプリケーション（パッケージ）のインストールの方法を説明します。授業中にも指示がありますが、あらかじめ準備しておくといいでしょう。

### 3.1 B3：情報・知能工学実験（データサイエンス）

日本語文を計算機に読み込ませるために、形態素解析器を導入します。形態素解析器は、空白などの区切りのない日本語文を単語単位に分割するアプリケーションです。今回、一般的によく使われる MeCab<sup>6</sup>を利用します。授業では curl も必要になりますが、これはビルドツールのインストールで導入済みです。

MeCab のインストール

```
$ sudo apt update
$ sudo apt install mecab
```

作動確認

```
$ mecab -v
$ echo "今日は晴天なり" | mecab
$ echo "今日は晴天なり" | mecab -Owakati
$ curl --version
```

MeCab を起動した場合、Ctrl を押しながら c を押すことで終了できます。「curl」は外部からファイルをダウンロードするのに利用します。

---

<sup>6</sup><http://taku910.github.io/mecab/>



### 3.2 B3 : ソフトウェア演習 IV

オブジェクト指向プログラミングのために, Java の開発環境 (JDK) を導入します. 授業では git も必要になりますが, これはビルドツールのインストールで導入済みです.

Java のインストール

```
$ sudo apt update  
$ sudo apt install default-jdk
```

作動確認

```
$ java -version  
$ javac -version  
$ git --version
```

「java」は Java のプログラムを実行するために利用し, 「javac」は Java のソースコードをコンパイルする (プログラムに変換する) ために利用します. 「git」はソースコードをバージョン管理するために利用します.

### 3.3 B3 : プログラム言語論

関数型プログラミングのために, CLISP と Emacs を導入します. Emacs は主要なエディタの一つです.

CLISP と Emacs のインストール

```
$ sudo apt update  
$ sudo apt install clisp emacs
```

作動確認

```
$ clisp --version  
$ emacs --version
```

CLISP を起動した場合, 「(exit)」と入力することで終了できます. また, Emacs を起動した場合, Ctrl を押しながら x を押したあと, Ctrl を押しながら c を押すことで終了できます.

### 3.4 M1 : Web システム特論

Web システムの構築演習のために, Node.js を導入します.

Node.js のインストール

```
$ sudo apt update  
$ sudo apt install nodejs
```

作動確認

```
$ nodejs -v
```

Node.js を起動した場合, 「.exit」と入力することで終了できます.