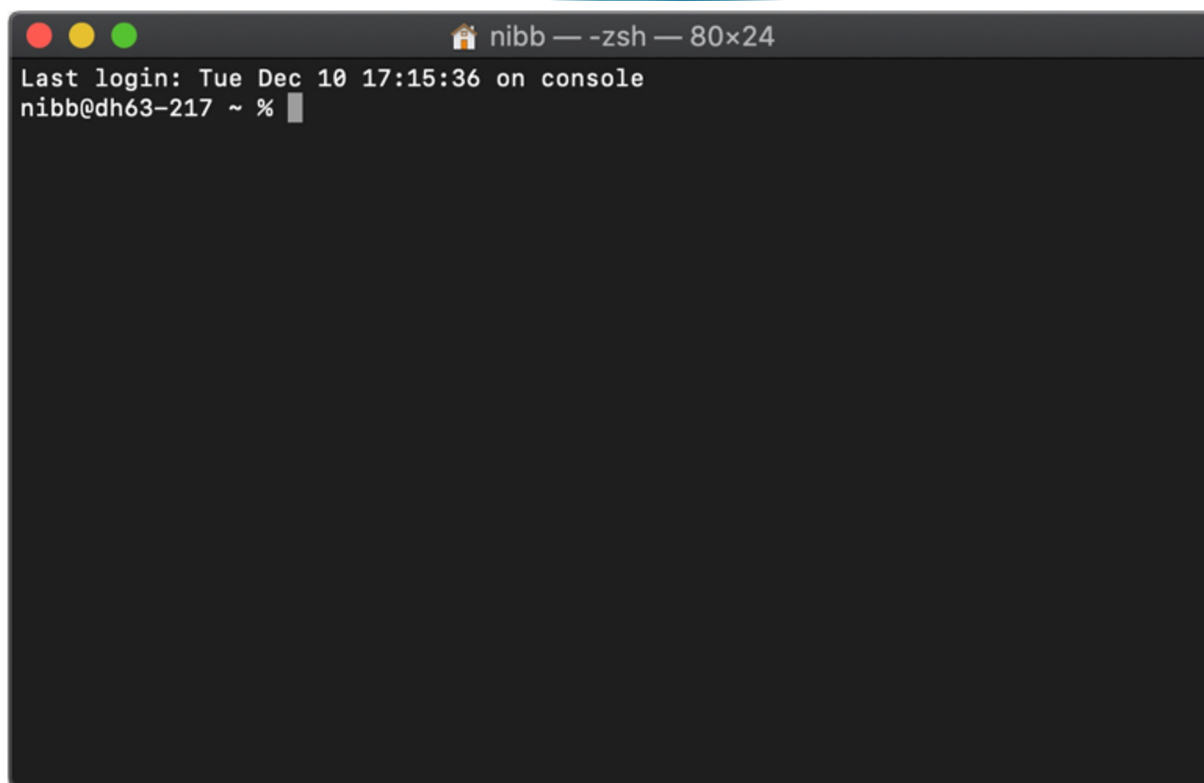


UNIX環境の構築(macOS編)

Mac環境でUnixコマンドを使用するためには

A screenshot of a macOS terminal window. The title bar shows a home icon, the name 'nibb', and the command '-zsh' followed by the window size '80x24'. The terminal content shows a login message: 'Last login: Tue Dec 10 17:15:36 on console', followed by the prompt 'nibb@dh63-217 ~ %' and a cursor. The terminal has a dark background and standard macOS window controls (red, yellow, green buttons) in the top left corner.

```
nibb — -zsh — 80x24
Last login: Tue Dec 10 17:15:36 on console
nibb@dh63-217 ~ %
```

改訂 2025/7

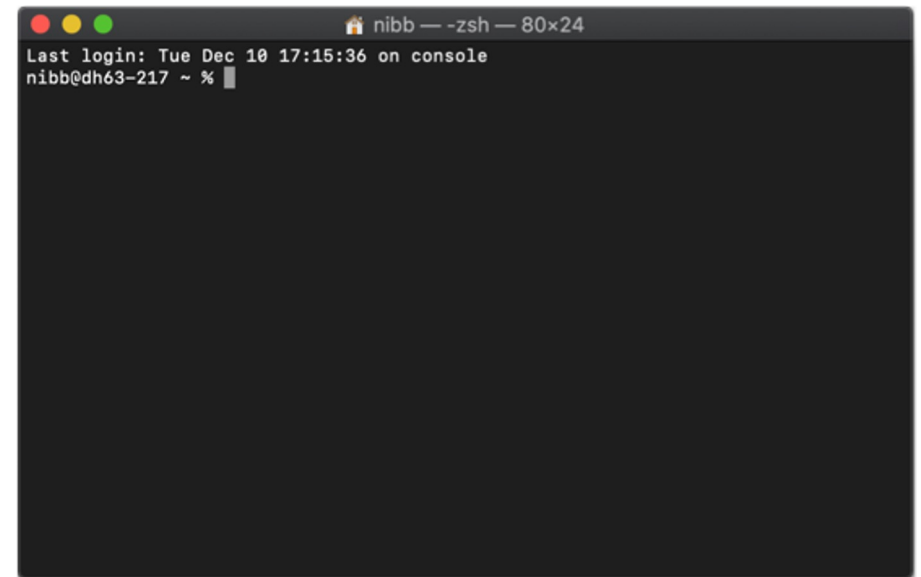
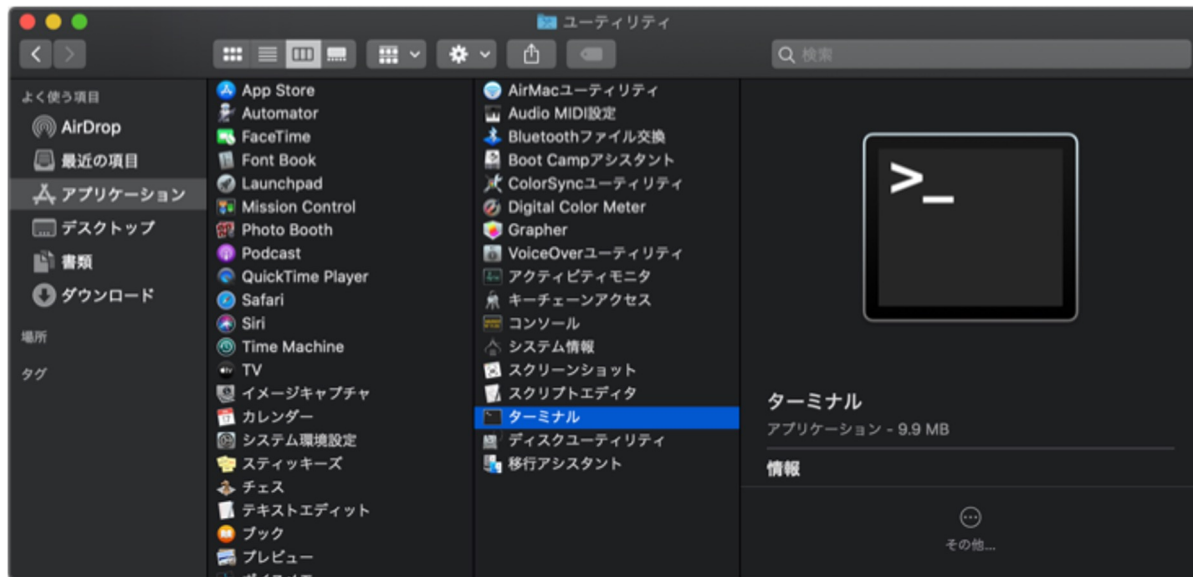
概要

- ▶ macOSは元々Unix系のOSであるDarwin上で動作している。そのため、ターミナル.appから標準的なUnixコマンドを使用することが可能である。
- ▶ しかし、本格的にUnixコマンドを使用した解析操作を行おうとすると、必要なソフトウェアが動作しない・インストールされていないため、つまづくことになる。
- ▶ 本資料はmacOSにおいて必要なソフトウェアをインストールし、解析操作に必要なUnix環境を構築する方法を紹介する。

ターミナル.app



- ▶ [Finder]から[アプリケーション] -> [ユーティリティ]を選択。
- ▶ ターミナル.appを実行することでシェルが起動する。



Xcode

- ▶ Xcode (<https://developer.apple.com/jp/xcode/>)
- ▶ Xcode はソフトウェアを開発するための統合開発環境である。
- ▶ 米Apple社からmacOS用に無償で配布されている。
- ▶ 今回必要なツールはXcodeの一部機能のCommand Line Tools というツールであり、Xcode 全てをインストールする必要はない。
- ▶ (Xcode 全体の容量は7.8GBと巨大である。全てインストールしたい場合はApp Storeから)
- ▶ 今回はCommand Line Tools のみをインストールする。



導入手順の概要

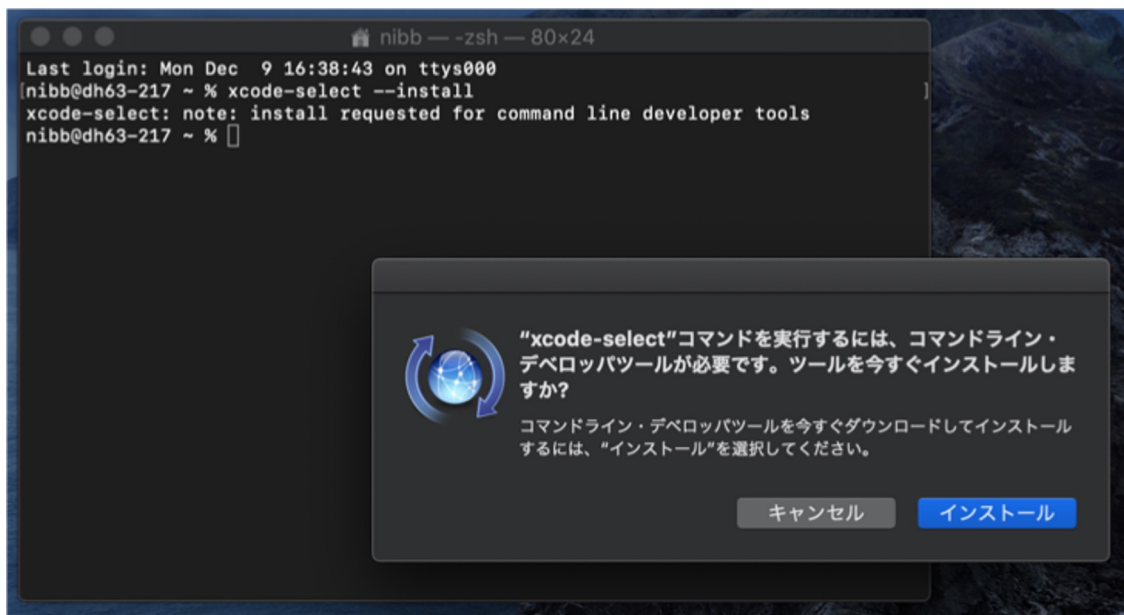
- ▶ 1) Xcode のCommand Line Tools をインストールする。
- ▶ 2) パッケージシステムをインストールする。
(Homebrew, Macports, etc)
- ▶ 3) パッケージシステムを用いて、ソフトウェアをインストールする。

Command Line Tools

- ▶ 前述のXcode は統合開発環境であり、様々なものが同梱されている。
- ▶ その中でソースファイルからコンパイルをするために必要なものは、Xcode に付随するCommand Line Tools である。
- ▶ Command Line Tools のみをインストールしたい場合、ターミナル.appから以下のコマンドを入力することでインストールすることができる。

```
xcodeselect --install
```


Command Line Tools のインストール



- ▶ xcode のCommand Line Toolsのみをインストールする場合、ターミナルで以下のコマンドを入力

```
xcode-select --install
```

- ▶ 左図のようなポップが現れたら「インストール」を選択する。

パッケージシステム

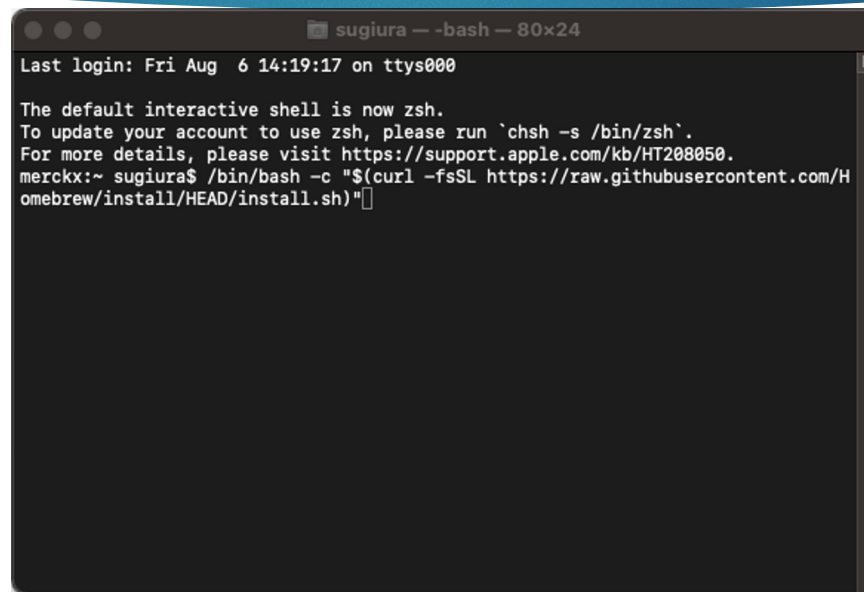
- ▶ Unix環境にソフトウェアをインストールする場合、ソフトウェアに関連づけされているファイル一式をまとめた「パッケージ」を用いることが多い。
- ▶ それらパッケージを用いて、必要なファイルと依存関係も含めてインストールの補助を行ってくれるのが「パッケージシステム」である。
- ▶ macOSで利用できるパッケージシステムは、Homebrew やMacports などが代表例として挙げられる。
- ▶ 本資料ではHomebrew を使用したソフトウェアのインストールを紹介する。

Homebrew



- ▶ Homebrew (<https://brew.sh/ja/>)
- ▶ MacOS上でソフトウェアの導入を単純化するパッケージ管理システムの一つ
- ▶ 管理者権限がない一般ユーザでも使用可能
- ▶ パッケージインストール先は/usr/local
- ▶ インストールにかかる時間が比較的少ない

Homebrew のインストール



```
sugiura ~ -bash — 80x24
Last login: Fri Aug 6 14:19:17 on ttys000

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
merckx:~ sugiura$ /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

- ▶ ターミナル.appを開き、以下のコマンドを入力する。

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

(打ち込むコマンドが長いため、<https://brew.sh/ja/> からコピー&ペーストするとよい)

(2025/7追記 M系チップユーザの方はコマンド実行後次のスライドをご確認ください)

Homebrew のインストール

```
nibb — ruby -e #!/usr/bin/ruby\012# This script installs to /usr/local only. To ins...
/usr/local/share/man/man1/brew.1
/usr/local/share/zsh/site-functions/_brew
/usr/local/etc/bash_completion.d/brew
/usr/local/Homebrew
==> The following new directories will be created:
/usr/local/bin
/usr/local/etc
/usr/local/include
/usr/local/lib
/usr/local/sbin
/usr/local/share
/usr/local/var
/usr/local/opt
/usr/local/share/zsh
/usr/local/share/zsh/site-functions
/usr/local/var/homebrew
/usr/local/var/homebrew/linked
/usr/local/Cellar
/usr/local/Caskroom
/usr/local/Homebrew
/usr/local/Frameworks

Press RETURN to continue or any other key to abort
```

- ▶ リターンキーを押してHomebrewのインストールを続行する。

Homebrew のインストール

```
nibb — sudo • ruby -e #!/usr/bin/ruby\012# This script installs to /usr/local only...
/usr/local/bin
/usr/local/etc
/usr/local/include
/usr/local/lib
/usr/local/sbin
/usr/local/share
/usr/local/var
/usr/local/opt
/usr/local/share/zsh
/usr/local/share/zsh/site-functions
/usr/local/var/homebrew
/usr/local/var/homebrew/linked
/usr/local/Cellar
/usr/local/Caskroom
/usr/local/Homebrew
/usr/local/Frameworks

Press RETURN to continue or any other key to abort
==> /usr/bin/sudo /bin/mkdir -p /usr/local/bin /usr/local/etc /usr/local/include
/usr/local/lib /usr/local/sbin /usr/local/share /usr/local/var /usr/local/opt /
usr/local/share/zsh /usr/local/share/zsh/site-functions /usr/local/var/homebrew
/usr/local/var/homebrew/linked /usr/local/Cellar /usr/local/Caskroom /usr/local/
Homebrew /usr/local/Frameworks

Password: [REDACTED]
```

- ▶ macOS上での管理者権限を持っている場合、パスワードを入力する。
- ▶ 入力しないとHomebrew自体のインストールができない。
- ▶ ターミナル画面でパスワードを入力する際、入力した文字列は表示されない。しかし、ちゃんと入力されているのでタイプミスしないよう慎重に入力を行う。
- ▶ 入力したらエンターキーで決定

Homebrew のインストール

```
nibb — zsh — 80x24
https://github.com/Homebrew/brew#donations
==> Tapping homebrew/core
Cloning into '/usr/local/Homebrew/Library/Taps/homebrew/homebrew-core'...
remote: Enumerating objects: 5101, done.
remote: Counting objects: 100% (5101/5101), done.
remote: Compressing objects: 100% (4897/4897), done.
remote: Total 5101 (delta 50), reused 315 (delta 8), pack-reused 0
Receiving objects: 100% (5101/5101), 4.15 MiB | 4.39 MiB/s, done.
Resolving deltas: 100% (50/50), done.
Tapped 2 commands and 4883 formulae (5,143 files, 12.8MB).
Already up-to-date.
==> Installation successful!

==> Homebrew has enabled anonymous aggregate formulae and cask analytics.
Read the analytics documentation (and how to opt-out) here:
https://docs.brew.sh/Analytics

==> Homebrew is run entirely by unpaid volunteers. Please consider donating:
https://github.com/Homebrew/brew#donations
==> Next steps:
- Run `brew help` to get started
- Further documentation:
https://docs.brew.sh
nibb@nibbnoMacBook-Pro ~ %
```

- ▶ Homebrewのインストールが完了しました。
- ▶ メッセージに従い、`brew help` と入力すると、Homebrewのヘルプが表示されます。

Homebrew のインストール(M系チップの方のみ)

(M系チップの方のみ)

- ▶ インストールコマンドを実行すると、
「下記2つのコマンドを実行してPATHに追加してください」と言われます
- ▶ 下記コマンドを2つ実行してください。

```
(echo; echo 'eval "$(/opt/homebrew/bin/brew shellenv)'" >> /Users/nibb/.zprofile  
eval "$(/opt/homebrew/bin/brew shellenv)"
```

参考: <https://nullnull.dev/blog/how-to-install-homebrew-on-m1-mac/>

Homebrew の基本操作

- ▶ ここではbowtie2というパッケージを例に説明します。
- ▶ `brew search bowtie2(パッケージ名)`
示されるパッケージ(ここではbowtie2)を検索します。
- ▶ `brew install bowtie2(パッケージ名)`
パッケージ(ここではbowtie2)をインストールします。
- ▶ `brew uninstall bowtie2(パッケージ名)`
パッケージ(ここではbowtie2)をアンインストールします。
- ▶ `brew info bowtie2(パッケージ名)`
インストール前にパッケージ(ここではbowtie2)の内容を確認します。

Homebrew の基本操作

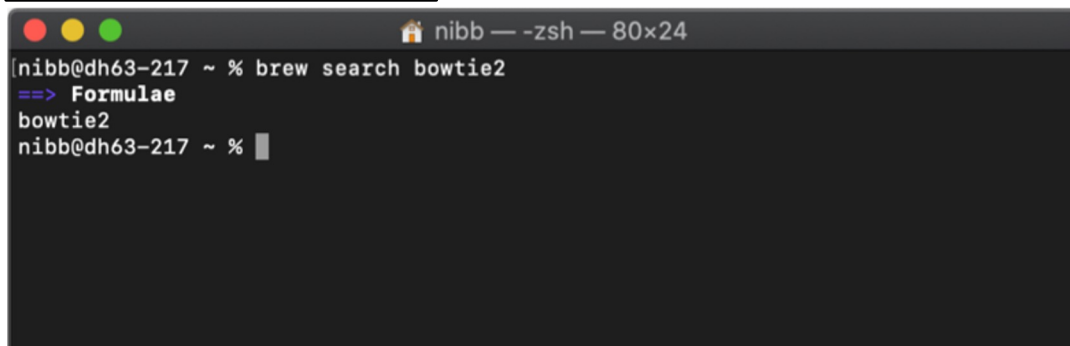
- ▶ ここではbowtie2というパッケージについて
- ▶ `brew list`
homebrewを使ってインストールしたパッケージ一覧を表示します
- ▶ `brew upgrade bowtie2`
bowtie2を最新版に更新します
- ▶ `brew upgrade`
インストール済のソフトウェアを全て更新します
- ▶ `brew cleanup`
古いソフトを自動で削除します

Homebrew の基本操作

- ▶ `brew update`
Homebrew 自体を最新にします。
- ▶ `brew doctor`
Homebrew の環境診断 (コンフリクトの確認)
- ▶ `brew help`
brew に続くコマンドの使い方の確認
- ▶ デフォルトのインストール場所は
/usr/local/Cellar

Homebrew の使用例

- ▶ Homebrewを使ってbowtie2をインストールする例を紹介します。
- ▶ (参照 : <http://bowtie-bio.sourceforge.net/bowtie2/index.shtml>)
- ▶ `brew search bowtie2` でbowtie2を探します。



```
nibb@dh63-217 ~ % brew search bowtie2
==> Formulae
bowtie2
nibb@dh63-217 ~ %
```

- ▶ bowtie2というパッケージが配布されていることを確認できました。

Homebrew の使用例

- ▶ bowtie2 を実際にインストールします。

```
nibb@dh63-217 ~ % brew search bowtie2
=> Formulae
bowtie2
nibb@dh63-217 ~ % brew install bowtie2
```

brew install bowtie2

```
nibb ~ - zsh — 80x24
echo 'export PATH="/usr/local/opt/sqlite/bin:$PATH"' >> ~/.zshrc

For compilers to find sqlite you may need to set:
export LDFLAGS="-L/usr/local/opt/sqlite/lib"
export CPPFLAGS="-I/usr/local/opt/sqlite/include"

=> python
Python has been installed as
/usr/local/bin/python3

Unversioned symlinks 'python', 'python-config', 'pip' etc. pointing to
'python3', 'python3-config', 'pip3' etc., respectively, have been installed into
/usr/local/opt/python/libexec/bin

If you need Homebrew's Python 2.7 run
brew install python@2

You can install Python packages with
pip3 install <package>
They will install into the site-package directory
/usr/local/lib/python3.7/site-packages

See: https://docs.brew.sh/Homebrew-and-Python
nibb@nibbnoMacBook-Pro ~ %
```

- ▶ 特にエラーの表示が出ていなければインストールが完了したことになります。
- ▶ 実際にbowtie2 を実行してみましょう。

bowtie2

Homebrew の使用例

```
nibb@nibbnoMacBook-Pro ~ % bowtie2
No index, query, or output file specified!
Bowtie 2 version 2.3.5.1 by Ben Langmead (langmea@cs.jhu.edu, www.cs.jhu.edu/~langmea)
Usage:
  bowtie2 [options]* -x <bt2-idx> {-1 <m1> -2 <m2> | -U <r> | --interleaved <i> | -b <bam>} [-S <sam>]

<bt2-idx>  Index filename prefix (minus trailing .X.bt2).
           NOTE: Bowtie 1 and Bowtie 2 indexes are not compatible.
<m1>      Files with #1 mates, paired with files in <m2>.
           Could be gzip'ed (extension: .gz) or bzip2'ed (extension: .bz2).
<m2>      Files with #2 mates, paired with files in <m1>.
           Could be gzip'ed (extension: .gz) or bzip2'ed (extension: .bz2).
<r>       Files with unpaired reads.
           Could be gzip'ed (extension: .gz) or bzip2'ed (extension: .bz2).
<i>       Files with interleaved paired-end FASTQ/FASTA reads
           Could be gzip'ed (extension: .gz) or bzip2'ed (extension: .bz2).
<bam>     Files are unaligned BAM sorted by read name.
<sam>     File for SAM output (default: stdout)

<m1>, <m2>, <r> can be comma-separated lists (no whitespace) and can be
specified many times.  E.g. '-U file1.fq,file2.fq -U file3.fq'.
```

- ▶ オプションなしで実行したため、bowtie2についての説明文が出てきました。
- ▶ バージョンは2.3.5.1となっています。
- ▶ これでbowtie2をインストールできたことを確認できました。