

# Hail-jp 「The 2nd」 Hands on seminars

第1回 Hail をインストールしてみよう！

**日時**  
2022年 1月 27 日

**会場**  
オンライン開催

mac でも windows でも Linux でも OK  
全3回開催のオンラインセミナーで、インタラクティブでスケーラブルな遺伝統計研究のためのライブラリ Hail をお手元の環境で試せるようになります

第1回 Hail をインストールしてみよう！(1/27 開催)  
第2回 GWAS tutorial を動かしてみよう！(2月下旬開催予定)  
第3回 Hail を使ってポリジェニックスコアを計算してみよう！(3月下旬開催予定)

お問い合わせは [hail-jp-staff@googlegroups.com](mailto:hail-jp-staff@googlegroups.com) まで

会場等協力：  
**GAJ**  
Genomic Analytics Japan  
**NABE International**  
株式会社 ナベインターナショナル



## Hail-jp ハンズオンセミナー

「Hailをインストールしてみよう！」

14:00- m1 mac

14:30- intel mac

15:00- Windows slot#1

15:30- Windows slot#2

16:00- Linux

解説: 内田智博



# 今日の目標

---

自分の環境で動くhailを手に入れる！

できるだけプレーンな状態のマシンにインストールをデモします。

この組み合わせなら「動くはず」と「手順」を手に入れてください。



# 今日の流れ

---

## 1. (DEMO) 実機にインストール

並行してStep by Stepで実行していただきます

(適宜使用するソフトウェアのバージョンやポイントを解説します。)

## 2. Dockerを使ったHailを紹介

## 3. 質疑応答



# 1. (DEMO) 実機にインストール

## 並行してStep by Stepで実行していただきます

注意! まだほとんど使用していないPCへのインストールを前提としています。

使い込んでいる場合はすでにjavaやpythonがあるかもしれません。  
適宜読み替えていただくな、後ほどslackでご相談ください。



## 1. DEMO 実機にインストール

# Hailの動作要件

Java 8

どのLinux??

(最近Java 11がサポートされました。今回は8で行きます)

## Install Hail on GNU/Linux

- Install Java 8 or Java 11.
- Install Python 3.7 or later.
- Install a recent version of the C and C++ standard libraries. GCC 5.0, LLVM version 3.4, or any later versions suffice.
- Install BLAS and LAPACK.
- Install Hail using pip.

Python

Hail

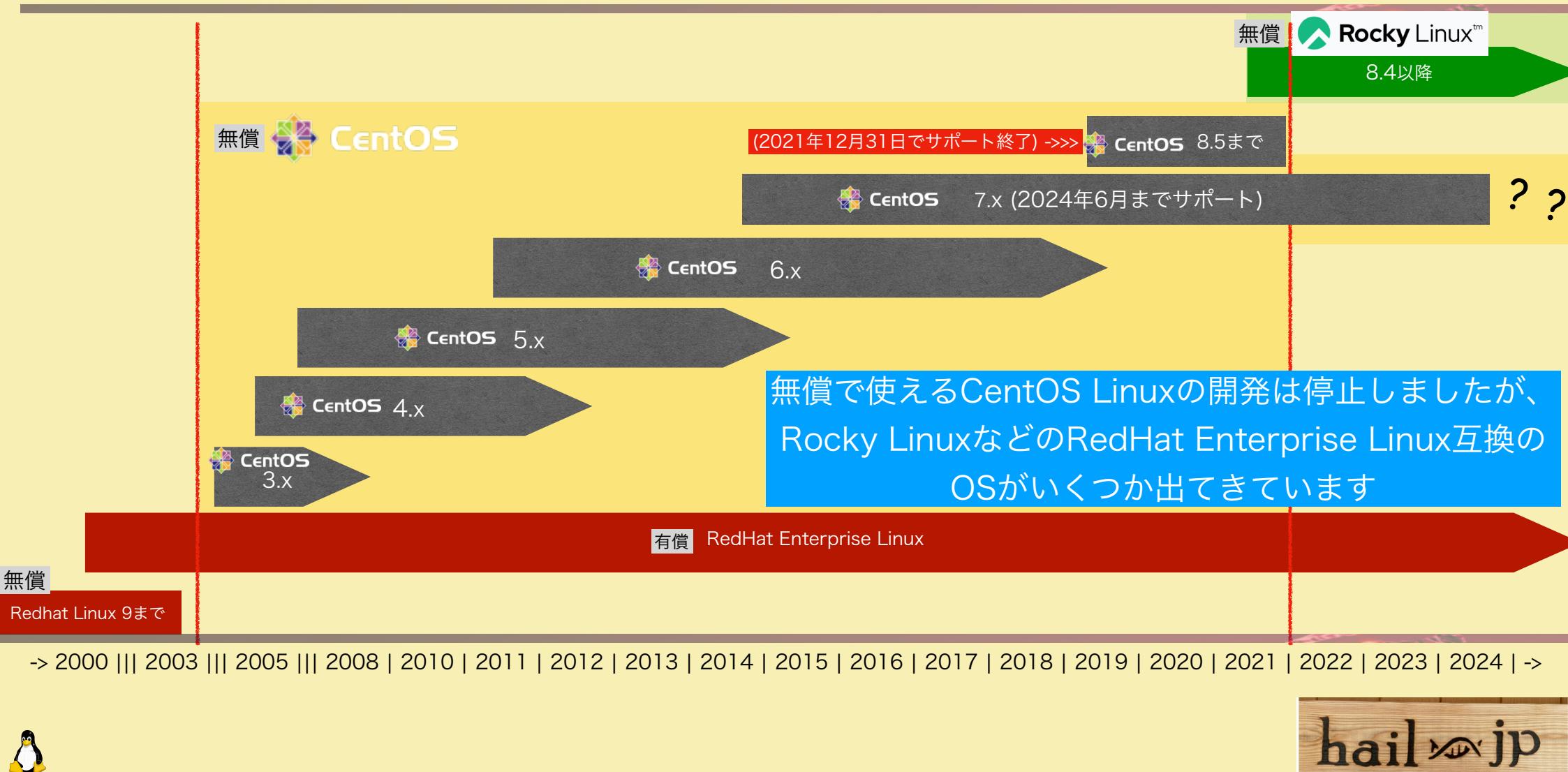
Jupyterlabも

<https://hail.is/docs/0.2/install/linux.html>



## 1. DEMO 実機にインストール

# RedHat系Linuxの変遷



## 1. DEMO 実機にインストール

# RedHat系Linuxの変遷

無償  Rocky Linux™

8.4以降

今日はRedhat Enterprise Linux 8.5互換の  
Rocky Linux 8.5に対して、  
root権限がない場合のインストールをデモします

root権限ありの場合については方法を紹介いたします。

無償

Redhat Linux 9まで

-> 2000 ||| 2003 ||| 2005 ||| 2008 | 2010 | 2011 | 2012 | 2013 | 2014 | 2015 | 2016 | 2017 | 2018 | 2019 | 2020 | 2021 | 2022 | 2023 | 2024 | ->



hail ≈ jp

## 1. DEMO 実機にインストール

# root権限ありの場合と無しの場合

一般ユーザーの  
権限で可能

root権限が必要

/bin , /etc , /usr , /var, /lib など、  
システム上重要なディレクトリ上の  
ファイルへの自由な読み書き

自分のホーム  
ディレクトリ以下  
への読み書き

みんなで共有するコマンドを  
/usr/local以下にインストール



非特権ポートの使用



今回は、一般ユーザー の権限  
のみでインストールできる低侵襲な  
方法でhailを構築します

特権ポートの使用



## 1. DEMO 実機にインストール

# 一般ユーザーが読み書きできる場所

### /tmpディレクトリ以下

```
[ut@rock-10-66 ~]$ ls -lL /
total 252
dr-xr-xr-x. 2 root root 49152 Jan 14 14:38 bin
dr-xr-xr-x. 5 root root 4096 Dec 13 13:16 boot
drwxr-xr-x. 19 root root 2120 Jan 20 15:57 dev
drwxr-xr-x. 142 root root 15:57 etc
drwxr-xr-x. 4 root root 13:35 home
drwxr-xr-x. 37 root root 13:06 lib
dr-xr-xr-x. 127 root root 13:10 lib64
drwxr-xr-x. 2 root root 09:48 media
drwxr-xr-x. 2 root root 6 Oct 11 09:48 mnt
drwxr-xr-x. 2 root root 6 Oct 11 09:48 opt
dr-xr-xr-x. 219 root root 0 Jan 20 15:57 proc
dr-xr-x---. 19 root root 4096 Jan 24 13:33 root
drwxr-xr-x. 41 root root 1240 Jan 20 15:59 run
dr-xr-xr-x. 2 root root 20480 Dec 13 13:08 sbin
drwxr-xr-x. 2 root root 6 Oct 11 09:48 srv
dr-xr-xr-x. 13 root root 0 Jan 20 15:57 sys
drwxrwxrwt. 12 root root 4096 Jan 24 13:34 tmp
drwxr-xr-x. 13 root root 158 Dec 13 13:07 usr
drwxr-xr-x. 21 root root 4096 Dec 13 13:16 var
```

読んで実行できるが  
書き込みできない

### /var/tmpディレクトリ以下

```
[ut@rock-10-66 ~]$ ls -lL /var|grep drwxrwxrw
drwxrwxrwt. 7 root root 4096 Jan 20 15:58 tmp
```

自分のホームディレクトリ以下

この場合はユーザー名が「ut」なので、

ホームディレクトリは「/home/ut」になります

```
[ut@rock-10-66 ~]$ ls -lL /home
total 0
```

```
drwx-----. 3 beowulf beowulf 78 Aug 10 09:29 beowulf
drwx----- 4 ut ut-g 92 Jan 24 13:35 ut
```

ディレクトリを所有  
しているユーザー名

ディレクトリ名

先頭の「d」はディレクトリである証

「rwx-----」は3文字でひと組になっていて、

それぞれ「ディレクトリを所有するユーザー」、「ディレクトリを所有するグループ」、「その他」を表し、rwxでそれぞれのアクセス権を表現しています。

rがread、wはwrite、xはexecutable(実行可能)であるかどうかを示しています。  
この場合はutユーザーのみが読み書き実行できる、ということを示しています。



## 1. DEMO 実機にインストール

# 一般ユーザーが使えるポート

ポートは電話番号のようなもの

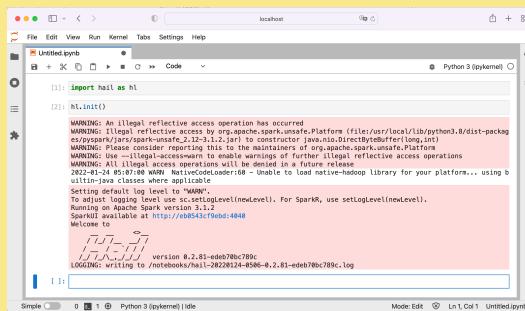
sshなら22番、httpなら80番など、サービスごとにある程度決まっています。  
サーバープログラムが動作して、アクセスされるのを待っています。

また、1024番以下は特権ポートと呼ばれ、rootのみが操作することができます。

hailではjupyterで8888番など  
sparkのUIで4040番などが使われますが、  
いずれも1025番以上なので

一般ユーザーの操作可能な範囲内ということになります。

http://localhost:8888



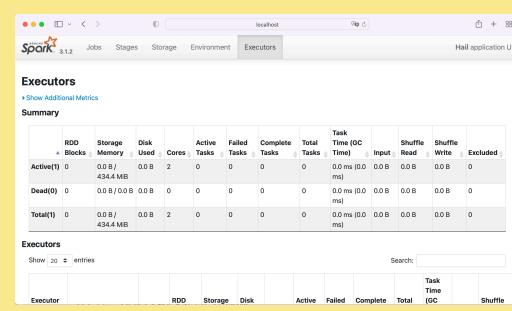
A screenshot of a Jupyter Notebook interface. The code cell contains:

```
import hail as hl
hl.init()
```

The output cell shows a warning message:

```
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.conf.Configuration (file:/usr/local/hadoop/lib/hadoop-3.8/dist-packs/org/spark/jars/spark-unsafe_2.12-3.1.2.jar) to constructor java.nio.DirectByteBuffer.(long,int)
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.conf.Configuration
WARNING: All illegal reflective access operations will be denied in a future release
2023-04-28 14:54:43,708 WARN NativeCodeLoader:108 - Unable to load native-hadoop library for your platform... using built-in Java classes where applicable
Setting default log level to "WARN".
To adjust logging level, set log4j.level.newLevel or log4j.level.newLevel.
Running on Apache Hadoop version 3.1.2
sparkX01 available at http://node043:4440
Hail version 0.2.81-edeb78bc78bc
```

http://localhost:4040



サーバー

webサーバー

http 80番

https 443番

22番

sshサーバー

53番

DNSサーバー

1024以下のポートは「特権ポート」  
rootのみ操作可能

8888番

jupyterlab

4040番

spark UI

hail.jp

## 1. DEMO 実機にインストール

# 一般ユーザーの場合はホームディレクトリの中にすべて押し込む

管理者にrootでインストールしてもらうと

```
ut@rock-10:~$ ls -ll /  
total 252  
dr-xr-xr-x. 2 root root 49152 Jan 14 14:38 bin  
dr-xr-xr-x. 5 root root 4096 Dec 13 13:16 boot  
drwxr-xr-x. 19 root root 3120 Jan 20 15:57 dev  
drwxr-xr-x. 142 root root 8192 Jan 20 15:57 etc  
drwxr-xr-x. 4 root root 31 Jan 24 13:35 home  
dr-xr-xr-x. 37 root root 4096 Dec 13 13:06 lib  
dr-xr-xr-x. 127 root root 81920 Dec 13 13:10 lib64  
drwxr-xr-x. 2 root root 6 Oct 11 09:48 media  
drwxr-xr-x. 2 root root 6 Oct 11 09:48 mnt  
drwxr-xr-x. 2 root root 6 Oct 11 09:48 opt  
dr-xr-xr-x. 219 root root 0 Jan 20 15:57 proc  
dr-xr-x--. 19 root root 4096 Jan 24 13:33 root  
drwxr-xr-x. 41 root root 1240 Jan 20 15:59 run  
dr-xr-xr-x. 2 root root 20480 Dec 13 13:08 sbin  
drwxr-xr-x. 2 root root 6 Oct 11 09:48 srv  
dr-xr-xr-x. 13 root root 0 Jan 20 15:57 sys  
drwxrwxrwt. 12 root root 4096 Jan 24 13:34 tmp  
drwxr-xr-x. 13 root root 158 Dec 13 13:07 usr  
drwxr-xr-x. 21 root root 4096 Dec 13 13:16 var
```

/usr/bin以下や/usr/local/bin以下などに  
コマンドがインストールされますが・・・

そうでない場合は何とかして自分のホームディレクトリに押し込みます

```
[[ut@rock-10-66 ~]$ ls -ll /  
total 252  
dr-xr-xr-x. 2 root root 49152 Jan 14 14:38 bin  
dr-xr-xr-x. 5 root root 4096 Dec 13 13:16 boot  
drwxr-xr-x. 19 root root 3120 Jan 20 15:57 dev  
drwxr-xr-x. 142 root root 8192 Jan 20 15:57 etc  
drwxr-xr-x. 4 root root 31 Jan 24 13:35 home  
dr-xr-xr-x. 37 root root 4096 Dec 13 13:06 lib  
dr-xr-xr-x. 127 root root 81920 Dec 13 13:10 lib64  
drwxr-xr-x. 2 root root 6 Oct 11 09:48 media  
drwxr-xr-x. 2 root root 6 Oct 11 09:48 mnt  
drwxr-xr-x. 2 root root 6 Oct 11 09:48 opt  
dr-xr-xr-x. 219 root root 0 Jan 20 15:57 proc  
dr-xr-x--. 19 root root 4096 Jan 24 13:33 root  
drwxr-xr-x. 41 root root 1240 Jan 20 15:59 run  
dr-xr-xr-x. 2 root root 20480 Dec 13 13:08 sbin  
drwxr-xr-x. 2 root root 6 Oct 11 09:48 srv  
dr-xr-xr-x. 13 root root 0 Jan 20 15:57 sys  
drwxrwxrwt. 12 root root 4096 Jan 24 13:34 tmp  
drwxr-xr-x. 13 root root 158 Dec 13 13:07 usr  
drwxr-xr-x. 21 root root 4096 Dec 13 13:16 var
```

```
[[ut@rock-10-66 ~]$ ls -ll /home  
total 0  
drwx-----. 3 beowulf beowulf 78 Aug 10 09:29 beowulf  
drwx----- 4 ut ut-g 92 Jan 24 13:35 ut
```



## 1. DEMO 実機にインストール

# ホームディレクトリにいろいろなツールを押し込む、便利なminiconda

### Install Hail on Mac OS X

- Install Java 8. We recommend using a [packaged installation from Azul](#) (make sure the OS version and architecture match your system) or using [Homebrew](#):

```
brew cask install adoptopenjdk8
brew install --cask adoptopenjdk8
```

- Install Python 3.6+. We recommend [Miniconda](#).
- Open Terminal.app and execute `pip install hail`.
- Run your first Hail query!

Miniconda

ソフトウェアを簡単にインストールできる、  
パッケージ管理ツールです。  
特に、Python関連の管理が得意です。  
javaもこれでインストールします。

Minicondaというと、右のMinicondaを指しますが、  
今回は「miniforge」という別のものを代わりに  
使います。

Miniconda is a free minimal installer for conda. It is a small, bootstrap version of Anaconda that includes only conda, Python, the packages they depend on, and a small number of other useful packages, including pip, zlib and a few others. Use the `conda install` command to install 720+ additional conda packages from the Anaconda repository.

See if Miniconda is right for you.

#### System requirements

- License: Free use and redistribution under the terms of the [EULA for Miniconda](#).
- Operating system: Windows 8 or newer, 64-bit macOS 10.13+, or Linux, including Ubuntu, RedHat, CentOS 7+, and others.
- If your operating system is older than what is currently supported, you can find older versions of the Miniconda installers in our [archive](#) that might work for you.
- System architecture: Windows- 64-bit x86, 32-bit x86; macOS- 64-bit x86 & Apple M1 (ARM64); Linux- 64-bit x86, 64-bit aarch64 (AWS Graviton2 / ARM64), 64-bit IBM Power8/Power9, s390x (Linux on IBM Z & LinuxONE).
- The `linux-armhf` Miniconda installer requires `libc >=2.26` and thus will not work with CentOS 7, Ubuntu 16.04, or Debian 9 ("stretch").
- Minimum 400 MB disk space to download and install.

On Windows, macOS, and Linux, it is best to install Miniconda for the local user, which does not require administrator permissions and is the most robust type of installation. However, if you need to, you can install Miniconda system wide, which does require administrator permissions.

#### Latest Miniconda Installer Links

Latest - Conda 4.10.3 Python 3.9.5 released July 21, 2021

Platform	Name	SHA256 hash
Windows	Miniconda3 Windows 64-bit	b33797064593ab2229a8135dc69001bea85cb5
	Miniconda3 Windows 32-bit	24f438e5ff2ef1ce1e93858d4e9d13f585895
MacOSX	Miniconda3 Mac OSX 64-bit bash	7864e9721f43e2c7d2883144cd6357fe48234
	Miniconda3 Mac OSX 64-bit pkg	8fa371ae97218c3b805cd5f84b1f40156d1586

<https://docs.conda.io/en/latest/miniconda.html>



# Miniforgeである理由

## 1. DEMO 実機にインストール

標準的にはanaconda/minicondaが使われていましたが、2020年4月30日の発表で、利用が有償になるケースがでてきました。商用利用や200人以上の企業は注意が必要です。

miniforgeは無償で利用できる代替え手段のひとつです。

### Miniforge

 Build miniforge passing

This repository holds a minimal installer for Conda specific to conda-forge. Miniforge allows you to install the conda package manager with the following features pre-configured:

- conda-forge set as the default (and only) channel.
  - Packages in the base environment are obtained from the [conda-forge](#) channel.
- Optional support for PyPy in place of standard Python interpreter (aka "CPython").
- Optional support for [Mamba](#) in place of Conda.
- An emphasis on supporting various CPU architectures (x86\_64, ppc64le, and aarch64 including Apple M1).

It can be compared to the [Miniconda](#) installer.

<https://github.com/conda-forge/miniforge>

Miniforgeは使用するリポジトリが無償で利用できる「conda-forge」にセットされています。



## 1. DEMO 実機にインストール

# Javaのバージョンを整理

Java 8 とは？？

Stable LTSとあるのが安定版

Javaの最新バージョンは18

Javaの最新の安定版は17

今回使うのは8

明示的に8を指定してインストールしないと、17が入ってしまい  
Hailが動かない場合があるので  
注意！

	Supported	Ready for Production	Role	Comment
JDK 8	+	✓	Stable LTS	Available as non-mainline 8u backport. Check with your vendor for availability. See known vendors list below.
JDK 9	!	✗		Discontinued, migrate to 11/17 <b>as soon as possible</b> .
JDK 10	!	✗		Discontinued, migrate to 11/17 <b>as soon as possible</b> .
JDK 11	✓	✓	Stable LTS	In mainline OpenJDK 11u <b>since 11.0.9</b> . Requires opt-in during build time, check with your vendor for availability. See known vendors list below.
JDK 12	!	✗		Discontinued, migrate to 17 <b>as soon as possible</b> .
JDK 13	!	!		Discontinued, migrate to 17
JDK 14	!	!	OpenJDK(オープンソースのJava) のホームページより	
JDK 15	!	!	Documentation, bug reports, etc... https://openjdk.java.net/jeps/	
JDK 16	!	!		Discontinued, migrate to 17.
JDK 17	✓	✓	Stable LTS	In mainline OpenJDK builds.
JDK 18	💡	💡	Dev/Test	In mainline OpenJDK builds.

<https://wiki.openjdk.java.net/display/shenandoah/Main>



# Java 8のバージョン

## 1. DEMO 実機にインストール

Java 8

セキュリティやbugfixのために  
できるだけ最新を使いたい

Java 8 の最新は 8u312

Java 8のrelease 8u312  
を使う!



<https://wiki.openjdk.java.net/display/jdk8u/Main>

# OpenJDK Wiki

Dashboard > JDK 8u > Main

## Main

Created by Iris Clark, last modified by Andrew Hughes on Dec 28, 2021

### Welcome to OpenJDK 8 Updates!

OpenJDK 8 updates are a [separate project](#) of OpenJDK. [Andrew Haley](#) serves as the Project Lead. The list of Reviewers, Committe

#### Maintainers

- Andrew Haley
- Andrew Hughes
- Severin Gehwolf

#### Releases

Latest GA release: 8u312

Latest Generally Available (GA) binary releases of the OpenJDK jdk8u project are available at: [https://adoptopenjdk.net/upstream.h](https://adoptopenjdk.net/upstream.html)

Latest Early Access (EA) binary releases of the OpenJDK jdk8u project are available at: <https://adoptopenjdk.net/upstream.html?va>

#### Most recent and past release details:

- 8u312-b07 (GA), October 19th 2021 [Release] [Tag] [Binaries]
- 8u302-b08 (GA), July 20th 2021 [Release] [Tag] [Binaries]
- 8u292-b10 (GA), April 20th 2021 [Release] [Tag] [Binaries]
- 8u282-b08 (GA), January 19th 2021 [Release] [Tag] [Binaries]
- 8u275-b01 (GA), November 5th 2020 [Release] [Tag] [Binaries]
- 8u272-b10 (GA), October 20th 2020 [Release] [Tag] [Binaries]

## 1. DEMO 実機にインストール

# 利用可能なJavaのビルドの種類と選択

Java 8

複数のベンダーが  
OpenJDKをbuildして提供。  
それをインストールして使用する。

Oracle Javaは利用条件要確認

Linuxは各distributionによって  
公式のjavaがありますが、

今回はroot権限の要らないインストール  
を目指しますので、condaを使用してjava  
をインストールします。

ビルド	LTS	パーミッシブ	TCK	商用サポート
<a href="#">AdoptOpenJDK</a>	Yes	Yes	No	オプション(IBM)
<a href="#">Amazon Corretto</a>	Yes	Yes	Yes	オプション(AWS上)
<a href="#">Azul Zulu Builds of OpenJDK</a>	Yes	Yes	Yes	オプション
<a href="#">BellSoft Liberica JDK</a>	Yes	Yes	Yes	オプション
<a href="#">Eclipse Temurin</a>	Yes	Yes	Yes	オプション(Azul, IBM)
<a href="#">Microsoft Build of OpenJDK</a>	Yes	Yes	Yes	オプション(Azure上)
<a href="#">Oracle Java SE</a>	Yes	(バージョン次第)	Yes	Yes
<a href="#">Oracle OpenJDK</a>	No	Yes	Yes	No
<a href="#">Red Hat OpenJDK</a>	Yes	Yes	Yes	Yes
<a href="#">SapMachine</a>	Yes	Yes	Yes	オプション(SAP製品)

<https://wiki.openjdk.java.net/display/shenandoah/Main>

condaにより、 Azul zuluのjava8 がインストールされます



# pythonのバージョン

1. DEMO 実機にインストール

Python

hailの要求は3.7以上

Intel processor上で動くLinuxの場合はどれでもだいたい動きます

今回はroot権限の要らないインストールを目指しますので、  
condaのpythonを使用します。

Pyrhonはcondaのデフォルトを使用



# HailとJupyterlabのバージョン

1. DEMO 実機にインストール

Hail

Jupyterlab

ここまで選択を間違えなければ、これらは最新(=バージョン指定無し)でよい

Hailは最新を選択

Jupyterlabも最新を選択



# では、インストールしていきましょう

## 1. DEMO 実機にインストール

今日打つコマンド一覧です。先にチャット欄に貼り付けます。

```
# Miniforgeをインストール
curl -LO https://github.com/conda-forge/miniforge/releases/latest/download/Miniforge3-Linux-x86_64.sh
bash Miniforge3-Linux-x86_64.sh

(ライセンスに同意し、インストール先もそのままEnter。initするかどうかもy)
source ~/.bashrc

# pythonのバージョン確認 (3.9.7と返ってくるはず)
python --version

# javaをインストール
conda install openjdk==8.0.312

# javaのバージョン確認 (azulu zuluのjava8 release 312であることが確認できるはず)
java -version

# hailとjupyterlabをインストール
pip3 install hail jupyterlab

# 動作確認 jupyterlabの起動
jupyter-lab --ip=0.0.0.0 --no-browser --NotebookApp.token=''

# ブラウザで
http://「LinuxマシンのIPアドレス」:8888
を開く
```

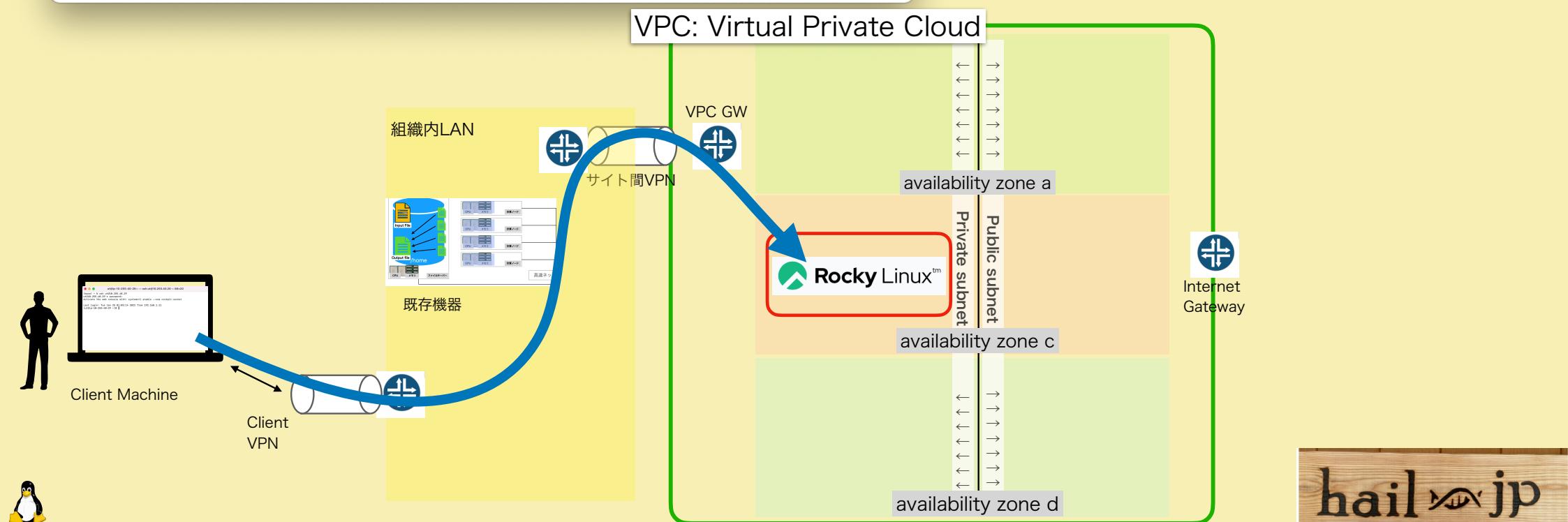


# Rocky8.5上へhail環境をインストール

## 1. DEMO 実機にインストール

まずログインしましょう

```
ut@ip-10-255-40-29:~ ssh ut@10.255.40.29
ut@10.255.40.29's password:
Activate the web console with: systemctl enable --now cockpit.socket
[ut@ip-10-255-40-29 ~]$
```



# Rocky8.5上へhail環境をインストール

## 1. DEMO 実機にインストール

現状確認をします

```
ut@ip-10-255-40-29:~ ssh ut@10.255.40.29 - 88x17
[[ut@ip-10-255-40-29 ~]$ java -version
-bash: java: コマンドが見つかりません
[[ut@ip-10-255-40-29 ~]$ python --version
-bash: python: コマンドが見つかりません
[[ut@ip-10-255-40-29 ~]$ python3 --version
-bash: python3: コマンドが見つかりません
[ut@ip-10-255-40-29 ~]$
```

例えば以下のように、既存のpython3やjavaがある環境かもしれません。  
バージョン等を確認し、使えるものは使っても構いませんが、今回はないものとして進めます。

```
root@rock:
[[root@rock-      # which python
/usr/bin/which: no python in (/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:
/root/bin)
[[root@rock      ]# which python3
/usr/bin/python3
[[root@rock      ]# which java
/usr/bin/java
[root@rock-      ]#
```



# Rocky8.5上へhail環境をインストール

## 1. DEMO 実機にインストール

# Miniforgeをインストール

```
ut@ip-10-255-40-29:~ ssh ut@10.255.40.29 - 122x17
[ut@ip-10-255-40-29 ~]$ curl -LO https://github.com/conda-forge/miniforge/releases/latest/download/Miniforge3-Linux-x86_64.sh
.
.
.
% Total    % Received % Xferd  Average Speed   Time   Time     Time  Current
.          .          .       Dload  Upload Total Spent   Left Speed
100  160  100  160    0     0   474      0 --:--:--:--:--:--:--:--:--:--: 474
100  665  100  665    0     0  1211      0 --:--:--:--:--:--:--:--:--: 1211
100 74.4M  100 74.4M    0     0  9829k      0 0:00:07  0:00:07 --:--:-- 9437k
[ut@ip-10-255-40-29 ~]$ bash Miniforge3-Linux-x86_64.sh
```

ライセンスに同意し、

```
ut@ip-10-255-40-29:~ ssh ut@10.255.40.29 - 122x17
.
.
.
may be used to endorse or promote products derived from software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Do you accept the license terms? [yes|no]
[no] >>> yes
```

インストール先もそのままEnter

```
Do you accept the license terms? [yes|no]
[no] >>> yes

Miniforge3 will now be installed into this location:
/home/ut/miniforge3

- Press ENTER to confirm the location
- Press CTRL-C to abort the installation
- Or specify a different location below

[/home/ut/miniforge3] >>>
```

by running conda init ?  
をyes

```
ut@ip-10-255-40-29:
.
.
.
Linking idna-3.1-pyh3deb0d_0
Linking colorama-0.4.4-pyh9f0ad1d_0
Linking charset-normalizer-2.0.9-pyh8ed1ab_0
Linking cffi-1.15.0-py39h4bc2ebd_0
Linking tqdm-4.62.3-pyh8ed1ab_0
Linking cryptography-36.0.0-py39h95dccef6_0
Linking brotliipy-0.7.0-py39h3811e60_1003
Linking conda-package-handling-1.7.3-py39h3811e60_1
Linking pyopenssl-21.0.0-pyh8ed1ab_0
Linking urllib3-1.26.7-pyh8ed1ab_0
Linking requests-2.26.0-pyh8ed1ab_1
Linking conda-4.11.0-py39hf3d152e_0
Transaction finished
installation finished.

Do you wish the installer to initialize Miniforge3
by running conda init? [yes|no]
[no] >>> yes
```

source ~/.bashrc  
で環境を有効にしますと、  
condaが利用できるようになります

```
ut@ip-10-255-40-29:~ ssh ut@10.255.40.29 - 122x17
.
.
.
no change   /home/ut/miniforge3/etc/fish/conf.d/conda.fish
no change   /home/ut/miniforge3/shell/condabin/Conda.psm1
no change   /home/ut/miniforge3/shell/condabin/conda-hook.ps1
no change   /home/ut/miniforge3/lib/python3.9/site-packages/xonsh/compat/_sysconfigdata__py39h95dccef6_0
no change   /home/ut/miniforge3/etc/profile.d/conda.csh
modified    /home/ut/.bashrc

==> For changes to take effect, close and re-open your current shell.

If you'd prefer that conda's base environment not be activated on
set the auto_activate_base parameter to false:

conda config --set auto_activate_base false

Thank you for installing Miniforge3!
[ut@ip-10-255-40-29 ~]$ source ~/.bashrc
(base) [ut@ip-10-255-40-29 ~]$
```



# Rocky8.5上へhail環境をインストール

## 1. DEMO 実機にインストール

pythonのバージョンを確認。3.9.7が入りました。

```
ut@ip-10-255-40-29:~ --  
[(base) [ut@ip-10-255-40-29 ~]$ python --version  
Python 3.9.7  
[(base) [ut@ip-10-255-40-29 ~]$ which python  
~/miniforge3/bin/python  
(base) [ut@ip-10-255-40-29 ~]$ ]
```

java8のインストール

```
ut@ip-10-255-40-29:~ -- ssh ut@10.255.40.29 - 122x27  
[(base) [ut@ip-10-255-40-29 ~]$ conda install openjdk==8.0.312  
Collecting package metadata (current_repodata.json): done  
Solving environment: done  
  
## Package Plan ##  
  
environment location: /home/ut/miniforge3  
  
added / updated specs:  
- openjdk==8.0.312  
  
The following packages will be downloaded:  
  
package | build  
-----|-----  
openjdk-8.0.312 | h7f98852_0 97.6 MB  conda-forge  
-----  
Total: 97.6 MB  
  
The following NEW packages will be INSTALLED:  
  
openjdk conda-forge/linux-64::openjdk-8.0.312-h7f98852_0  
  
Proceed ([y]/n)? ]
```

javaのバージョン確認

Zuluのopenjdk, build 1.8.0\_312-b07が入っています

```
ut@ip-10-255-40-29:~ -- ssh ut@10.255.40.29 - 122x27  
[(base) [ut@ip-10-255-40-29 ~]$ java -version  
openjdk version "1.8.0_312"  
OpenJDK Runtime Environment (Zulu 8.58.0.13-CA-linux64) (build 1.8.0_312-b07)  
OpenJDK 64-Bit Server VM (Zulu 8.58.0.13-CA-linux64) (build 25.312-b07, mixed mode)  
(base) [ut@ip-10-255-40-29 ~]$ ]
```



# Rocky8.5上へhail環境をインストール

## 1. DEMO 実機にインストール

hailとjupyterlabをインストールします

```
ut@ip-10-255-40-29:~ — ssh ut@10.255.4  
(base) [ut@ip-10-255-40-29 ~]$ pip install hail jupyterlab
```

pip listを使うとインストールされたバージョンの確認ができます

```
ut@ip-10-255-40-29:~ — ssh  
(base) [ut@ip-10-255-40-29 ~]$ pip list |grep hail  
hail          0.2.81  
(base) [ut@ip-10-255-40-29 ~]$ pip list |grep jupyterlab  
jupyterlab      3.2.8  
jupyterlab-pygments 0.1.2  
jupyterlab-server 2.10.3  
(base) [ut@ip-10-255-40-29 ~]$
```



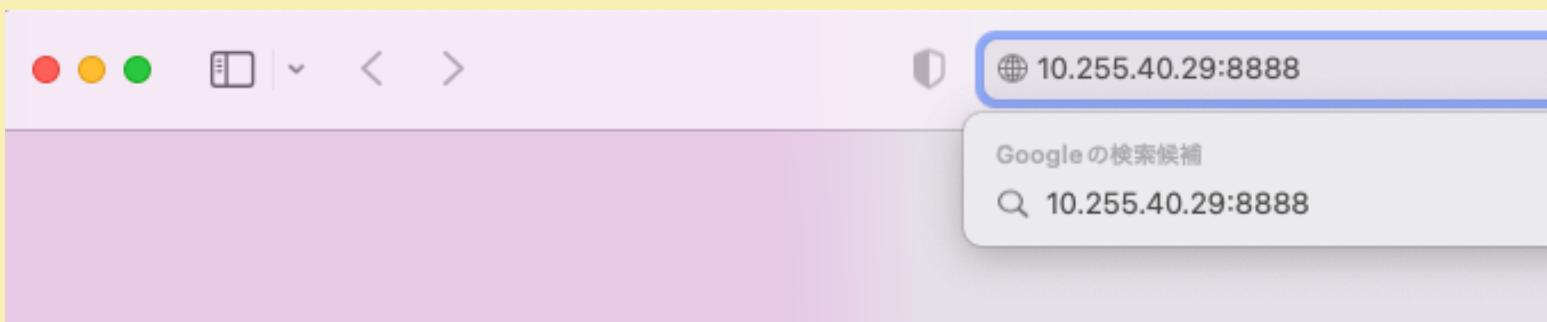
## 動作確認をしてみましょう

### 1. DEMO 実機にインストール

jupyter-labを起動

```
jupyter-lab --ip=0.0.0.0 --no-browser --NotebookApp.token=""  
ut@ip-10-255-40-29:~ ssh ut@10.255.40.29 - 145x27  
  
[base] [ut@ip-10-255-40-29 ~]$ jupyter-lab --ip=0.0.0.0 --no-browser --NotebookApp.token=''  
[W 2022-01-25 04:39:08.823 LabApp] 'token' has moved from NotebookApp to ServerApp. This config will be passed to ServerApp. Be sure to update your config before our next release.  
[I 2022-01-25 04:39:08.847 ServerApp] jupyterlab | extension was successfully linked.  
[I 2022-01-25 04:39:08.879 ServerApp] Writing Jupyter server cookie secret to /home/ut/.local/share/jupyter/runtime/jupyter_cookie_secret  
[I 2022-01-25 04:39:09.290 ServerApp] nbclassic | extension was successfully linked.  
[W 2022-01-25 04:39:09.310 ServerApp] All authentication is disabled. Anyone who can connect to this server will be able to run code.  
[I 2022-01-25 04:39:09.315 ServerApp] nbclassic | extension was successfully loaded.  
[I 2022-01-25 04:39:09.316 LabApp] JupyterLab extension loaded from /home/ut/miniforge3/lib/python3.9/site-packages/jupyterlab  
[I 2022-01-25 04:39:09.316 LabApp] JupyterLab application directory is /home/ut/miniforge3/share/jupyter/lab  
[I 2022-01-25 04:39:09.320 ServerApp] jupyterlab | extension was successfully loaded.  
[I 2022-01-25 04:39:09.320 ServerApp] ローカルディレクトリからノートブックをサーブ: /home/ut  
[I 2022-01-25 04:39:09.320 ServerApp] Jupyter Server 1.13.4 is running at:  
[I 2022-01-25 04:39:09.320 ServerApp] http://ip-10-255-40-29.ap-northeast-1.compute.internal:8888/lab  
[I 2022-01-25 04:39:09.320 ServerApp] or http://127.0.0.1:8888/lab  
[I 2022-01-25 04:39:09.321 ServerApp] サーバを停止し全てのカーネルをシャットダウンするには Control-C を使って下さい(確認をスキップするには2回)。
```

ウェブブラウザで、(LinuxマシンのIPアドレス):8888 を打ち込み、Enter



hail ✈ jp

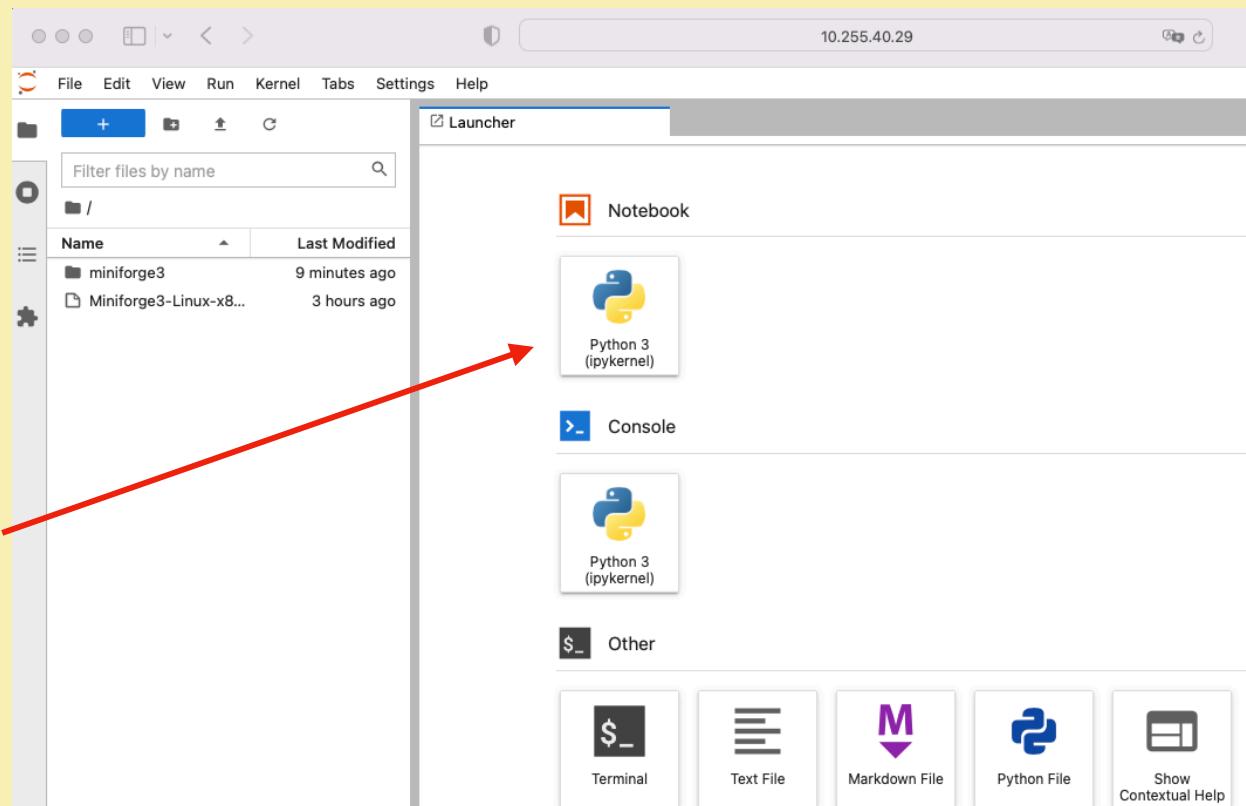


動作確認をしてみましょう

## 1. DEMO 実機にインストール

このような画面が出てきたらOK

Python3をクリックすると  
「Python kernel」が起動します



## 1. DEMO 実機にインストール

動作確認をしてみましょう

Python kernelが起動したら、

import hail as hl

を入力して「Shift + Enter」

次に、

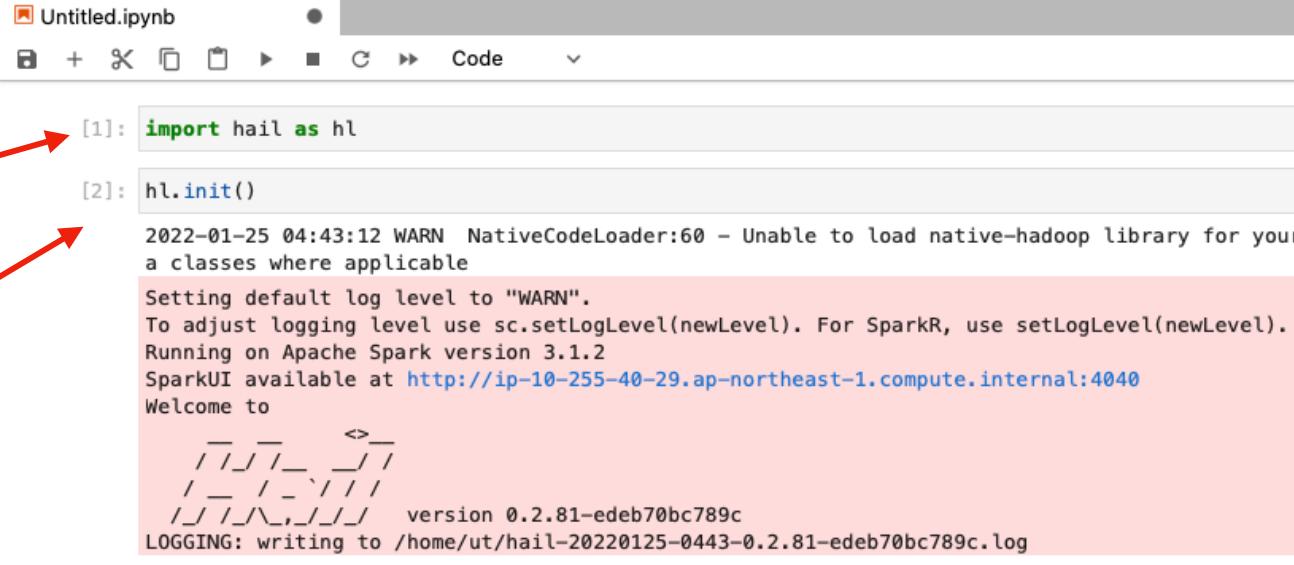
hl.init()

を入力して「Shift + Enter」

を押下します。

このように、「Welcome to」や

Hailのステキなアスキーアートが出力されたらOKです！



The screenshot shows a Jupyter Notebook interface with a single cell titled "Untitled.ipynb". The cell contains two lines of Python code: [1]: `import hail as hl` and [2]: `hl.init()`. The output of the second line is a series of ASCII art characters forming a stylized 'H' shape, followed by the text "version 0.2.81-edeb70bc789c". Below this, a log message indicates that logging is being written to a specific file path.

```
[1]: import hail as hl
[2]: hl.init()
2022-01-25 04:43:12 WARN NativeCodeLoader:60 - Unable to load native-hadoop library for your
a classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Running on Apache Spark version 3.1.2
SparkUI available at http://ip-10-255-40-29.ap-northeast-1.compute.internal:4040
Welcome to
   _/\_ _/\_ _/\_ _/\_ _/\_
  / \ / \ / \ / \ / \
 / / \ \ / \ / \ / \
 / / / \ \ / \ / \ / \
version 0.2.81-edeb70bc789c
LOGGING: writing to /home/ut/hail-20220125-0443-0.2.81-edeb70bc789c.log
```



## 2. Dockerを使ったHailを紹介

---



## Dockerを利用してhail環境を得る

Docker をインストール

RedHat系では次のコマンドでインストールできます

```
dnf install podman-docker
```

Docker がインストールできたら

```
docker run -it -p 8888:8888 -p 4040:4040 docker.io/utut/hail-jp-trial:latest
```

( -p 8888:8888はjupyterlabのポートを転送, -p 4040:4040はSparkUIのポートを転送しています)

docker.io/utut/hail-jp-trial:latestは今回のためを作った最小限のhail環境です

ブラウザにhttp://IP Address:8888 と打ち込めばjupyterlabを開けるようになります



## 2. Dockerを使ったHailの紹介

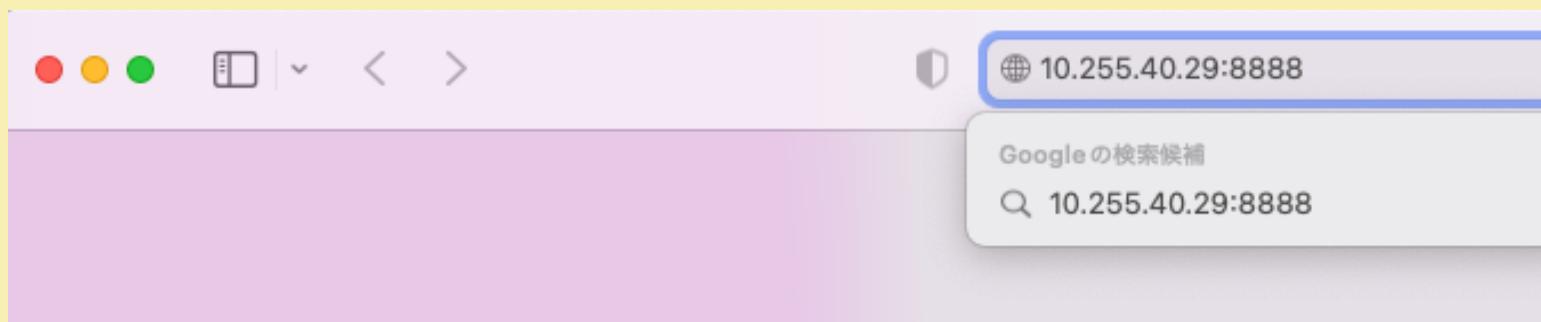
動作確認をしてみましょう

dockerコンテナを起動してjupyter-labを動かします

```
docker run -it -p 8888:8888 -p 4040:4040 utut/hail-jp-trial:latest
```

```
root@ip-10-255-40-202:~ — ssh ut@10.255.40.202 — 159x20
[[root@ip-10-255-40-202 ~]# docker run -it -p 8888:8888 -p 4040:4040 docker.io/utut/hail-jp-trial:latest
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.
Trying to pull docker.io/utut/hail-jp-trial:latest...
Getting image source signatures
Copying blob 8b7214509776 done
Copying blob 7b1a6ab2e44d done
Copying blob 8a9fd7be58d6 [=====>-----] 87.4MiB / 236.9MiB
Copying blob a3fb5c06c077 done
Copying blob 75136496f8e7 done
Copying blob 14e3a8b5e825 [==>-----] 128.2MiB / 1.1GiB
Copying blob c0eab4ba4e8e done
```

ウェブブラウザで、(LinuxマシンのIPアドレス):8888 を打ち込み、Enter

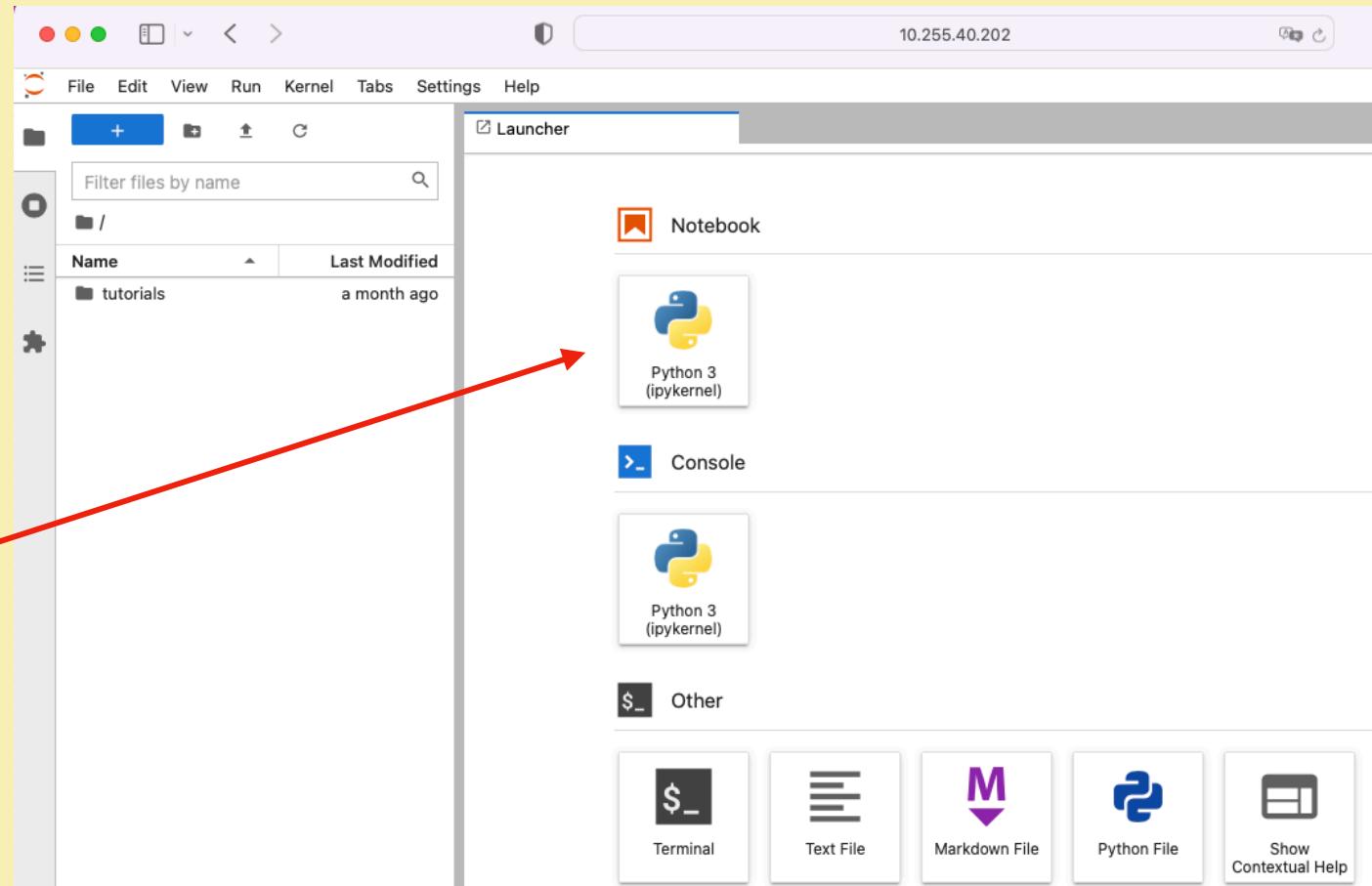


## 2. Dockerを使ったHailの紹介

動作確認をしてみましょう

このような画面が出てきたらOK

Python3をクリックすると  
「Python kernel」が起動します



動作確認をしてみましょう

Python kernelが起動したら、

import hail as hl

を入力して「Shift + Enter」

次に、

hl.init()

を入力して「Shift + Enter」

を押下します。

このように、「Welcome to」や

Hailのステキなアスキーアートが出力されたらOKです！

The screenshot shows a Jupyter Notebook interface with a single cell titled 'Untitled1.ipynb'. The cell contains two lines of Python code: '[1]: import hail as hl' and '[2]: hl.init()'. The output of line [2] is a series of log messages and a welcome message. The log messages include warnings about hostname resolution and Spark configuration. The welcome message starts with 'Welcome to' followed by a stylized ASCII logo consisting of various slashes and a double-headed arrow symbol. Below the logo, it says 'version 0.2.81-edeb70bc789c' and 'LOGGING: writing to /home/ut/hail-20220121-1520-0.2.81-edeb70bc789c.log'.

```
[1]: import hail as hl
[2]: hl.init()
2022-01-21 15:20:42 WARN  Utils:69 - Your hostname, DESKTOP-27N283U resolves to a loopback address instead of 172.28.89.162 (on interface eth0)
2022-01-21 15:20:42 WARN  Utils:69 - Set SPARK_LOCAL_IP if you need to bind to another address
2022-01-21 15:20:43 WARN  NativeCodeLoader:60 - Unable to load native-hadoop library for your platform built-in Java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Running on Apache Spark version 3.1.2
SparkUI available at http://172.28.89.162:4040
Welcome to
   _/\_ _/\_ _/\_ _/\_
  / \ / \ / \ / \ / \
 / / / \ \ , / / / \
version 0.2.81-edeb70bc789c
LOGGING: writing to /home/ut/hail-20220121-1520-0.2.81-edeb70bc789c.log
```



### 3. 質疑応答

宜しければ、Hail-jp のslackでもご相談ください



### 3. 質疑応答

宜しければ、Hail-jp のslackでもご相談ください

Hail-jp 「The 2nd」  
Hands on seminars

第2回  
Hail 公式 GWAS tutorial を動かしてみよう

日時  
2022年 2月下旬予定

会場  
オンライン開催

mac でも windows でも Linux でも OK  
全3回開催のオンラインセミナーで、インタラクティブでスケーラブルな遺伝統計研究のためのライブラリ Hail をお手元の環境で試せるようになります

第1回 Hail をインストールしてみよう！(1/27 開催)  
第2回 GWAS tutorial を動かしてみよう！(2月下旬開催予定)  
第3回 Hail を使ってポリジニックリスクスコアを計算してみよう！(3月下旬開催予定)

お問い合わせは [hail-jp-staff@googlegroups.com](mailto:hail-jp-staff@googlegroups.com) まで

会場等協力：  
**GAJ**  
**NABE International**

次回案内



# 補足. root権限を使ったインストールを紹介

---



# Rocky8.5上へhail環境をインストール

補足 root権限を使える場合

現状確認をします

```
● ○ ● ut@ip-10-255-40-29:~ — ssh ut@10.255.40.29 — 88x17
[[ut@ip-10-255-40-29 ~]$ java -version
-bash: java: コマンドが見つかりません
[[ut@ip-10-255-40-29 ~]$ python --version
-bash: python: コマンドが見つかりません
[[ut@ip-10-255-40-29 ~]$ python3 --version
-bash: python3: コマンドが見つかりません
[ut@ip-10-255-40-29 ~]$
```

例えば以下のように、既存のpython3やjavaがある環境かもしれません。  
バージョン等を確認し、使えるものは使っても構いませんが、今回はないものとして進めます。

```
● ○ ● root@rock:
[[root@rock-      # which python
/usr/bin/which: no python in (/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:
/root/bin)
[[root@rock      ]# which python3
/usr/bin/python3
[[root@rock      ]# which java
/usr/bin/java
[root@rock-      ]#
```



# Rocky8.5上へhail環境をインストール

補足 root権限を使える場合

dnfコマンドを使用してjava8をインストールします

dnf install java-1.8.0-openjdk-headless.x86\_64

```
[root@ip-10-255-40-202 ~]# dnf install java-1.8.0-openjdk-headless.x86_64
```

このように、/usr/bin/java がつくられ、  
OpenJDKの1.8.0\_312が入っていることがわかります

```
[root@ip-10-255-40-202 ~]# which java  
/usr/bin/java  
[root@ip-10-255-40-202 ~]# java -version  
openjdk version "1.8.0_312"  
OpenJDK Runtime Environment (build 1.8.0_312-b07)  
OpenJDK 64-Bit Server VM (build 25.312-b07, mixed mode)
```



# Rocky8.5上へhail環境をインストール

補足 root権限を使える場合

## pythonをインストール

同じくdnfコマンドでpythonをインストールしますが、次のようにバージョンが選択できたりします。  
ここでは特に依存関係のあるアプリケーションもありませんので最新の3.9を選択しました。

```
dnf install python3           ##### python3.6が入ります  
dnf install python38          ##### python3.8が入ります  
dnf install python39          ##### python3.9が入ります
```

```
root@ip-10-255-40-202:~ -- ssh ut@10.255.40.202 - 145x18  
[root@ip-10-255-40-202 ~]# dnf install python39  
Last metadata expiration check: 0:06:28 ago on Tue 25 Jan 2022 04:50:18 AM UTC.  
Dependencies resolved.  
=====  
 Package          Architecture      Version       Repository      Size  
=====  
Installing:  
 python39          x86_64          3.9.6-2.module+el8.5.0+673+10283621    appstream      31 k  
Installing dependencies:  
 python39-libs      x86_64          3.9.6-2.module+el8.5.0+673+10283621    appstream      8.2 M  
 python39-pip-wheel noarch          20.2.4-6.module+el8.5.0+673+10283621    appstream      1.3 M  
 python39-setuptools-wheel noarch          50.3.2-4.module+el8.5.0+673+10283621    appstream      496 k  
Installing weak dependencies:  
 python39-pip      noarch          20.2.4-6.module+el8.5.0+673+10283621    appstream      2.0 M  
 python39-setuptools noarch          50.3.2-4.module+el8.5.0+673+10283621    appstream      870 k  
Enabling module streams:  
 python39          3.9
```

```
root@ip-10-255-40-202:~ -- ssh ut@10.255.40.202 - 145x18  
[root@ip-10-255-40-202 ~]# /usr/bin/python3 --version  
Python 3.9.6
```

←3.9.6が入っています



# Rocky8.5上へhail環境をインストール

補足 root権限を使える場合

hailとjupyterlabをインストール

```
root@ip-10-255-40-202:~  
[[root@ip-10-255-40-202 ~]# /usr/bin/pip3 install hail jupyterlab
```

pip3 listでインストールされたバージョンを確認できます

```
root@ip-10-255-40-202:~  
[[root@ip-10-255-40-202 ~]# pip3 list |grep hail  
hail          0.2.81  
[[root@ip-10-255-40-202 ~]# pip3 list |grep jupyterlab  
jupyterlab    3.2.8  
jupyterlab-pygments 0.1.2  
jupyterlab-server   2.10.3
```

ここまでで、root権限を使うシステムレベルでのインストールが完了しました。

それぞれの一般ユーザーがhailやjupyterlabを使うことができるようになっています。



補足 root権限を使える場合

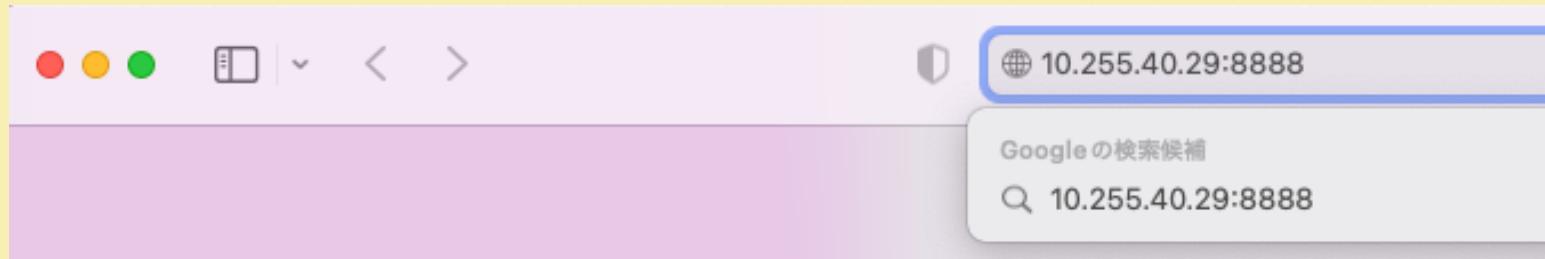
動作確認をしてみましょう

一般ユーザーでjupyter-labを起動

jupyter-lab --ip=0.0.0.0 --no-browser --NotebookApp.token=""

```
ut@ip-10-255-40-202:~ — ssh ut@10.255.40.202 — 145x19
[[ut@ip-10-255-40-202 ~]$ which jupyter-lab
/usr/local/bin/jupyter-lab
[[ut@ip-10-255-40-202 ~]$ jupyter-lab --ip=0.0.0.0 --no-browser --NotebookApp.token=''
[W 2022-01-25 05:09:20.495 LabApp] 'token' has moved from NotebookApp to ServerApp. This config will be passed to ServerApp. Be sure to update your config before our next release.
[I 2022-01-25 05:09:20.502 ServerApp] jupyterlab | extension was successfully linked.
[I 2022-01-25 05:09:20.512 ServerApp] Writing Jupyter server cookie secret to /home/ut/.local/share/jupyter/runtime/jupyter_cookie_secret
[I 2022-01-25 05:09:20.706 ServerApp] nbclassic | extension was successfully linked.
[W 2022-01-25 05:09:20.725 ServerApp] All authentication is disabled. Anyone who can connect to this server will be able to run code.
[I 2022-01-25 05:09:20.731 ServerApp] nbclassic | extension was successfully loaded.
[I 2022-01-25 05:09:20.732 LabApp] JupyterLab extension loaded from /usr/local/lib/python3.9/site-packages/jupyterlab
[I 2022-01-25 05:09:20.732 LabApp] JupyterLab application directory is /usr/local/share/jupyter/lab
[I 2022-01-25 05:09:20.735 ServerApp] jupyterlab | extension was successfully loaded.
[I 2022-01-25 05:09:20.736 ServerApp] Serving notebooks from local directory: /home/ut
[I 2022-01-25 05:09:20.736 ServerApp] Jupyter Server 1.13.4 is running at:
[I 2022-01-25 05:09:20.736 ServerApp] http://ip-10-255-40-202.ap-northeast-1.compute.internal:8888/lab
[I 2022-01-25 05:09:20.736 ServerApp] or http://127.0.0.1:8888/lab
[I 2022-01-25 05:09:20.736 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
```

ウェブブラウザで、(LinuxマシンのIPアドレス):8888 を打ち込み、Enter



## 動作確認をしてみましょう

補足 root権限を使える場合

複数のユーザーが同時に起動した場合の、jupyterlabのportについて

同じポートは複数のjupyterlabで重複して利用することができません。

別のユーザーが同時に使おうとすると、jupyterlabは別のポートをアサインしてくれます。

```
rocky@ip-10-255-40-202:~$ jupyter-lab --ip=0.0.0.0 --no-browser --NotebookApp.token=''
[W 2022-01-25 05:10:53.306 LabApp] 'token' has moved from NotebookApp to ServerApp. This config will be passed to ServerApp. Be sure to update your config before our next release.
[I 2022-01-25 05:10:53.314 ServerApp] jupyterlab | extension was successfully linked.
[I 2022-01-25 05:10:53.329 ServerApp] Writing Jupyter server cookie secret to /home/rocky/.local/share/jupyter/runtime/jupyter_cookie_secret
[I 2022-01-25 05:10:53.584 ServerApp] nbclassic | extension was successfully linked.
[W 2022-01-25 05:10:53.603 ServerApp] All authentication is disabled. Anyone who can connect to this server will be able to run code.
[I 2022-01-25 05:10:53.609 ServerApp] nbclassic | extension was successfully loaded.
[I 2022-01-25 05:10:53.610 LabApp] JupyterLab extension loaded from /usr/local/lib/python3.9/site-packages/jupyterlab
[I 2022-01-25 05:10:53.610 LabApp] JupyterLab application directory is /usr/local/share/jupyter/lab
[I 2022-01-25 05:10:53.613 ServerApp] jupyterlab | extension was successfully loaded
[I 2022-01-25 05:10:53.614 ServerApp] ポート 8888 は既に使用されています。他のポートで試して下さい。
[I 2022-01-25 05:10:53.614 ServerApp] ローカルディレクトリからノートブックをサーブ: /home/rocky
[I 2022-01-25 05:10:53.614 ServerApp] Jupyter Server 1.13.4 is running at:
[I 2022-01-25 05:10:53.614 ServerApp] http://ip-10-255-40-202.ap-northeast-1.compute.internal:8889/lab
[I 2022-01-25 05:10:53.614 ServerApp] or http://127.0.0.1:8889/lab
[I 2022-01-25 05:10:53.614 ServerApp] サーバを停止し全てのカーネルをシャットダウンするには Control-C を使って下さい(確認をスキップするには2回)。
```

この場合、表示されている通り、ウェブブラウザには (LinuxマシンのIPアドレス):8889 を打ち込みます。

