

人狼ゲームセットアップマニュアル

目次

| | |
|--------------|---|
| 1. サーバの用意 | 1 |
| 2. クライアントの設定 | 7 |

1. サーバの用意

サーバは表 1 の設定を用いてログインします。サーバは以下の(1)~(3)の手順でセットアップを行います。

表 1 サーバログイン情報

| | |
|-------------|----------------|
| サーバ IP アドレス | 150.89.233.203 |
| ログインユーザ名 | isdev22 |
| ログインパスワード | isDev22?203 |

- (1) 仮想サーバにログインします。ログインには表 1 の設定を用います。図 1 のコマンドを用いてログインします。

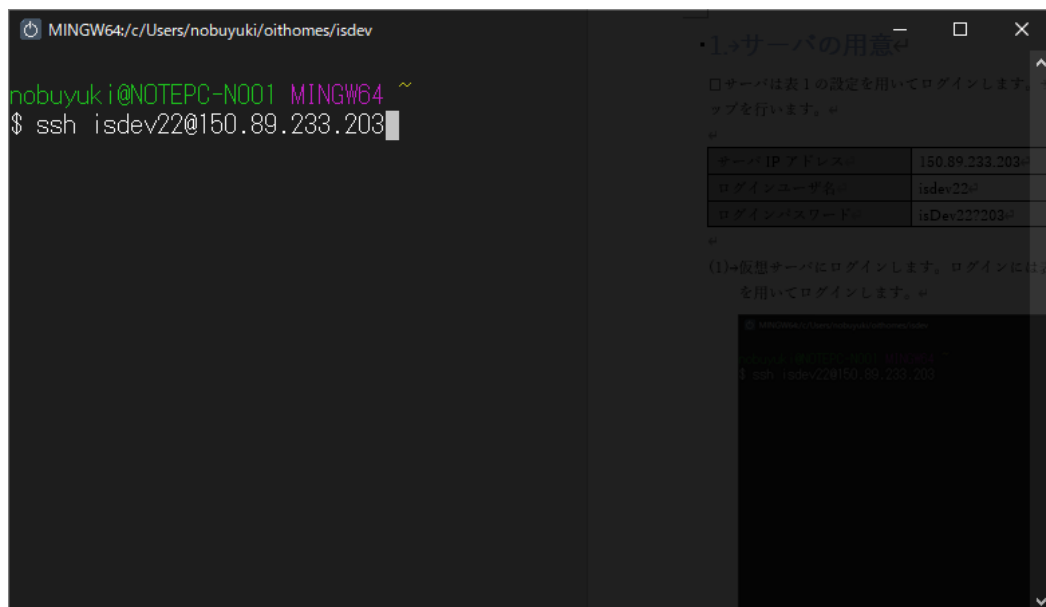


図 1 ログインコマンド

実行するとパスワードの入力を求められるので、表 1 のログインパスワードを入力してください。

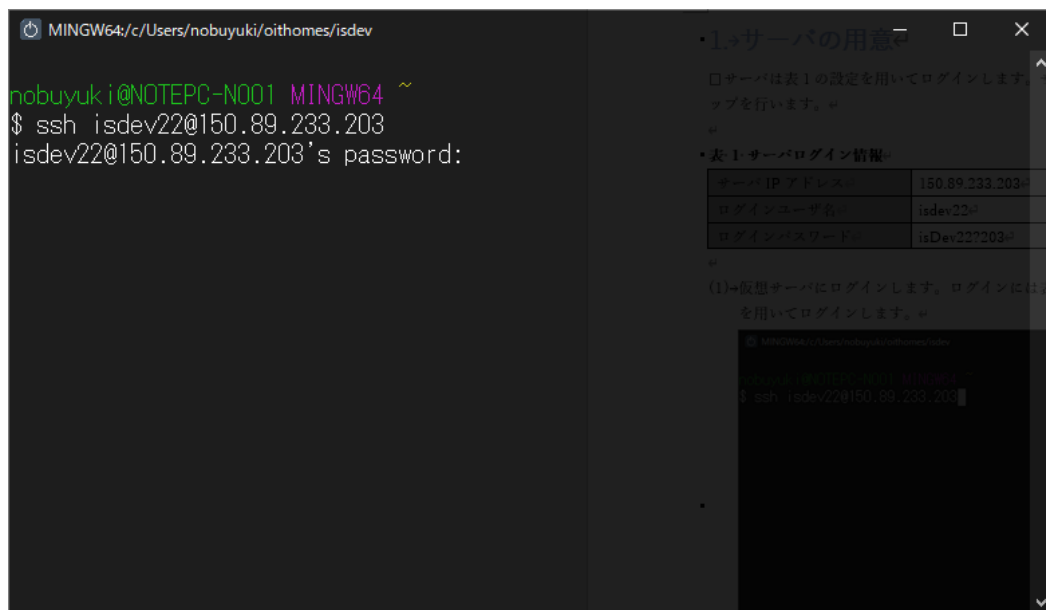


図 2 パスワード入力画面

ログインに成功すると図 3 のような画面になります。

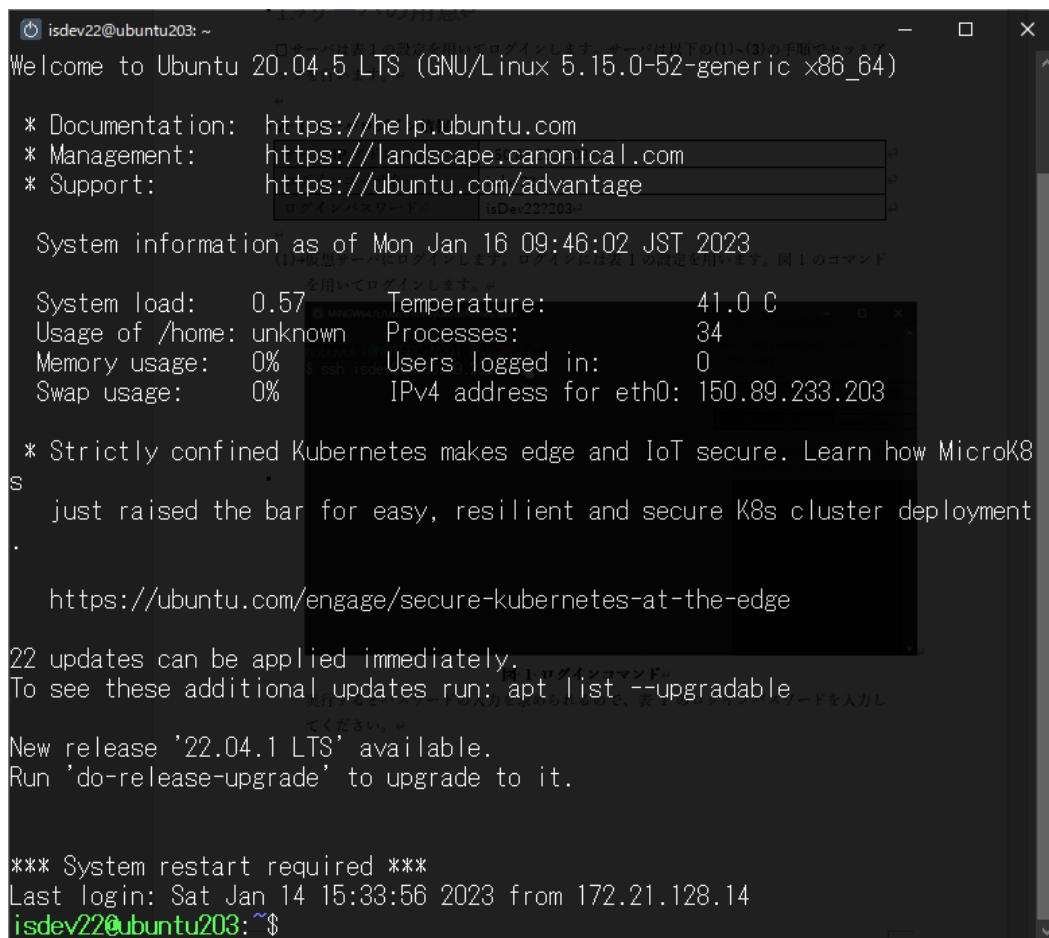


図 3 ログインの成功画面

(2) タイムゾーンの設定をします。設定には図 4 のコマンドを用います。設定には管理者

権限が必要なため、sudo で実行してください。設定後、timedatectl コマンドで Time Zone が Asia/Tokyo になっていることを確認してください。

```
isdev22@ubuntu203: ~  
isdev22@ubuntu203:~$ sudo timedatectl set-timezone Asia/Tokyo  
[sudo] password for isdev22:  
isdev22@ubuntu203:~$ timedatectl  
Local time: Mon 2023-01-16 09:48:56 JST  
Universal time: Mon 2023-01-16 00:48:56 UTC  
RTC time: n/a  
Time zone: Asia/Tokyo (JST, +0900)  
System clock synchronized: yes  
NTP service: inactive  
RTC in local TZ: no  
isdev22@ubuntu203:~$
```

図 4 タイムゾーン設定画面

(3) Java のインストールに必要なリポジトリのダウンロードと追加を行います。次のコマンドを実行してください。

```
$ wget -O- https://apt.corretto.aws/corretto.key | sudo apt-key add -
```

```
$ sudo add-apt-repository 'deb https://apt.corretto.aws stable main'
```

written to stdout の後に次に進まない場合は sudo 権限のパスワードを求めている場合があります。必要に応じて sudo で「add-apt-repository」コマンドを再実行してください。次に、apt コマンドで Java のインストールを行います。以下のコマンドを実行してください。apt update も同時に実行して下さい。

```
$ sudo apt-get update; sudo apt-get install -y java-11-amazon-corretto-jdk
```

インストール終了後、java のバージョンを確認してください。インストールに成功し

ていると図5のようになります。

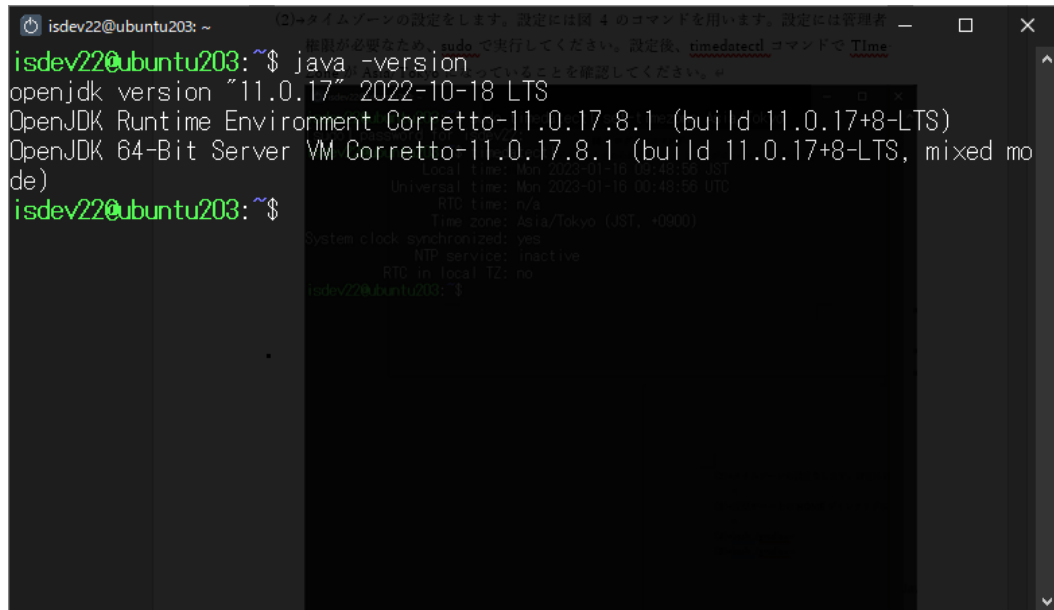
A terminal window titled 'isdev22@ubuntu203: ~' showing the output of the 'java -version' command. The output indicates 'openjdk version "11.0.17" 2022-10-18 LTS' and 'OpenJDK Runtime Environment Corretto-11.0.17.8.1 (build 11.0.17+8-LTS)'. Below this, the system time is displayed: 'Local time: Mon 2023-01-16 00:48:56 JST', 'Universal time: Mon 2023-01-16 00:48:56 UTC', 'RTC time: n/a', 'Time zone: Asia/Tokyo (JST, +0900)', 'System clock synchronized: yes', 'NTP service: inactive', and 'RTC in local TZ: no'. The prompt returns to 'isdev22@ubuntu203: ~\$'.

図5 インストール成功時画面

- (4) つぎに、準備した環境にアプリをデプロイします。仮想サーバ上の HOME ディレクトリにて git clone を行います。手順は図6を参考にしてください。

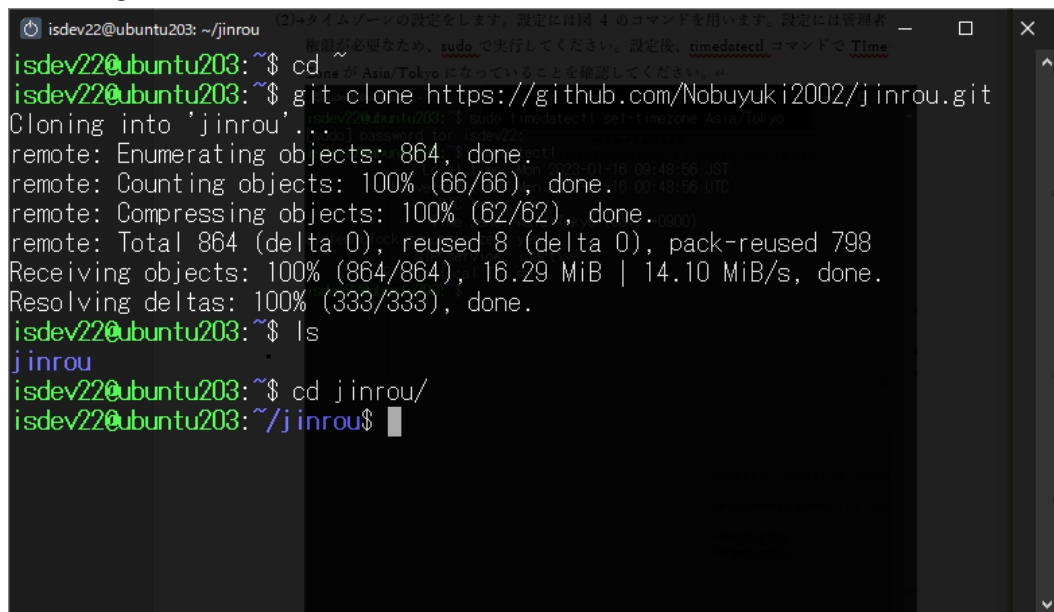
A terminal window titled 'isdev22@ubuntu203: ~/jinrou' showing the execution of 'git clone https://github.com/Nobuyuki2002/jinrou.git'. The output shows the cloning process: 'Cloning into 'jinrou'...', 'remote: Enumerating objects: 864, done.', 'remote: Counting objects: 100% (66/66), done.', 'remote: Compressing objects: 100% (62/62), done.', 'remote: Total 864 (delta 0), reused 8 (delta 0), pack-reused 798', 'Receiving objects: 100% (864/864), 16.29 MiB | 14.10 MiB/s, done.', and 'Resolving deltas: 100% (333/333), done.'. After the clone, the user runs 'ls', showing 'jinrou' as the only directory. Then, they run 'cd jinrou/' and 'cd ~/jinrou\$'.

図6 アプリのデプロイ手順

- (5) clone ができたら、リポジトリのディレクトリへ移動します。上記手順にて clone を行った場合には jinrou ディレクトリになります。ディレクトリを移動した後、アプリが動作できるようにビルドを行います。ビルドは図7のようにしてください。

```
isdev22@ubuntu203: ~/jinrou
isdev22@ubuntu203:~/jinrou$ bash ./gradlew
Starting a Gradle Daemon (subsequent builds will be faster)

> Task :help

Welcome to Gradle 7.5.1.

To run a build, run gradlew <task>
To see a list of available tasks, run gradlew tasks
To see more detail about a task, run gradlew help --task <task>
To see a list of command-line options, run gradlew --help

For more detail on using Gradle, see https://docs.gradle.org/7.5.1/userguide/command\_line\_interface.html

For troubleshooting, visit https://help.gradle.org

BUILD SUCCESSFUL in 4s
1 actionable task: 1 executed
isdev22@ubuntu203:~/jinrou$
```

図 7 ビルド画面

- (6) (5)までが成功した場合、`bash ./gradlew bootrun` を行い、アプリを起動します。起動に成功すると図8のような画面になります。

```

oisdev22@ubuntu203: ~/jinrou
org.springframework.security.web.context.SecurityContextPersistenceFilter@5d
bbb292, org.springframework.security.web.header.HeaderWriterFilter@2abafa97,
org.springframework.security.web.authentication.logout.LogoutFilter@43d65a8
1, org.springframework.security.web.authentication.UsernamePasswordAuthentic
ationFilter@9cfc77, org.springframework.security.web.authentication.ui.Default
LoginPageGeneratingFilter@412ebe64, org.springframework.security.web.authen
tication.ui.DefaultLogoutPageGeneratingFilter@217235f5, org.springframework
.web.servlet.mvc.method.annotation.RequestMappingHandlerMapping@565aa4ac, org.springframe
work.security.web.servletapi.SecurityContextHolderAwareRequestFilter@1c046c9
2, org.springframework.security.web.authentication.AnonymousAuthenticationFi
lter@4b41587d, org.springframework.security.web.session.SessionManagementFi
lter@40717ed, org.springframework.security.web.access.ExceptionTranslationFi
lter@5e50df2e, org.springframework.security.web.access.intercept.Authorization
Filter@15e1f8fe]
2023-01-16 10:37:17.656 INFO 110049 --- [ main] o.s.b.w.embedded
tomcat.TomcatWebServer : Tomcat started on port(s): 80 (http) with context
path ''
2023-01-16 10:37:17.663 INFO 110049 --- [ main] oit.is.ouchi.jinrou
JinrouApplication : Started JinrouApplication in 1.175 seconds (JVM ru
nning for 1.41)
<=====--> 80% EXECUTING [53s]
> :bootRun

```

図8 アプリの起動成功時

2. クライアントの設定

クライアント側は、ブラウザに接続できる環境を用意してください。動作テストはGoogle Chrome を用いています。ブラウザで、「<http://150.89.233.203>」へアクセスすると、図 9 の画面が表示されます。表示されなかった場合はセットアップが失敗している可能性があるため、再度「1 サーバの用意」手順(1)から実行してください。

接続に成功している場合は別途マニュアル(ユーザマニュアル)に従ってゲームをプレイしてください。

