Development Environment Setup

Setting Up WSL

Preparation

> **NOTE**: Make sure that the PC is connected to 'aird' Wi-Fi Network, after every restart until the environment setup is done.

- Check for any pending updates/restarts in Windows Update.
- Restart PC.

Installation

- Open Terminal as Administrator and check for any existing installed distros.

```
wsl --list --verbose
```

- Open Microsoft Store and install 'Ubuntu'.
- Once the installation is complete, open the installed app.
- Create a username and password following the format given below.

▼ Username Format

  'tw' + '<EMPLOYEE_ID>'

  For example, if EMPLOYEE_ID is 12345, the username is **tw12345**.

▼ Password Format

  The password should be a alphanumeric sequence with a minimum length of 8 characters.

Running the Setup Script

Downloading and Configuring

- Download and extract the zip file 'dev-env-setup.zip' in Teams.
- Edit the values of these variables manually using Notepad, leaving the rest untouched.
  - EMPLOYEE_ID
  - EMPLOYEE_EMAIL
  - EMPLOYEE_FULL_NAME
  - WINDOWS_USER_ROOT_FOLDER
- Refer to the sample given below.

```
# Sample
BASE_APT_DEPS="curl ca-certificates git build-essential wget htop"
```

```
EMPLOYEE_ID="12345"
EMPLOYEE_EMAIL="i_lorem@twave.co.jp"
EMPLOYEE_FULL_NAME="Lorem Ipsum"
GPG_KEY_ID="XXXXXXXXXXXXXXXX"
GIT_DEFAULT_BRANCH_NAME="main"
WINDOWS_USER_ROOT_FOLDER="12345"
```

- Inside the Ubuntu Bash Shell, copy the files inside the WSL environment using the following commands.

```
cd $HOME
cp "/mnt/c/Users/<WINDOWS_USER_ROOT_DIR>/Downloads/<EXTRACTED_FOLDER>/setup.s
h" ./ -v
cp "/mnt/c/Users/<WINDOWS_USER_ROOT_DIR>/Downloads/<EXTRACTED_FOLDER>/conf" ./
-v
```

Running the Script

- Run the setup script using the following command.

```
./setup.sh
```

Disabling Services

- Inside the Ubuntu Bash Shell, run this commands sequentially to disable services temporarily that have auto-start enabled but are not used currently.

```
sudo systemctl disable mongod
sudo systemctl disable postgresql
```

Graceful Restart

- Exit all Ubuntu Bash Shells.
- Check the status of the Ubuntu WSL VM using the following command and wait until it is stopped.

  ```
  wsl --list --verbose
  ```

- Restart PC.

Keys Setup

> **NOTE**: In the instructions given below, wherever "x_xxxxxxx@twave.co.jp" occurs, replace it with your work email address.

Download the Keys

- Open the **keyring** repository’s page in your GitHub account and download the following key files.
    - git.gpg.key
    - git.ssh.key
- Open a new shell in Ubuntu WSL and navigate to the home directory.

```
cd $HOME
```

- Run the following commands to copy the keys inside the WSL environment.

```
cp "/mnt/c/Users/<WINDOWS_USER_ROOT_DIR>/Downloads/git.gpg.key" ~/ -v
cp "/mnt/c/Users/<WINDOWS_USER_ROOT_DIR>/Downloads/git.ssh.key" ~/ -v
```

GPG Key Setup

- Run the following command to print the contents of the public key and copy the output.

```
gpg --import git.gpg.key
```

SSH Key Setup

- Run the following commands to create the necessary folders.

```
mkdir -p ~/.ssh
mkdir -p ~/.ssh/keys
```

- Run the following commands to copy the key inside the configured folder.

```
cp ~/git.ssh.key ~/.ssh/keys/github
```

- Run the following command to set the correct file permissions for the key.

```
chmod 600 ~/.ssh/keys/github
```

Delete the Keys

- Run the following command to delete the keys.

```
rm -v git.*.key
```

SSH Config Setup

- Run the following command to open the file in the editor.

```
nano ~/.ssh/config
```

- Copy and paste this content into the file.

```
AddKeysToAgent yes
Host github.com
    Hostname github.com
    IdentityFile ~/.ssh/keys/github
    IdentitiesOnly yes
```

Git Config Setup

- Run the following command and copy the GPG Key ID.

```
gpg --list-secret-keys "x_xxxxxxx@twave.co.jp"
```

- Update the ~/.gitconfig file in such a way, that the value for the attribute ‘**user.signingkey**’ is the copied **GPG Key ID** and the value for the attribute ‘**commit.gpgsign**’ is ‘**true**’ as shown below.

```
[user]
    signingkey = XXXXXXXXXXXXXXXX
[commit]
    gpgsign = true
```

- Run the following command, to update the related environment variable.

```
[ -f ~/.bashrc ] && echo -e '\nexport GPG_TTY=$(tty)' >> ~/.bashrc
# note: check if the above command worked by running this: echo $GPG_TTY
#       if the output is just an empty line, then open a new bash shell, run the command again an
d continue doing the rest of the setup in the same shell
```

Backing Up Keys

- Create a new private git repository in your GitHub Account, by setting visibility to ‘Private’ and name to ‘keyring’.
- Run the following command to clone the repository locally and navigate to the cloned folder.

```
git clone git@github.com:TW-XXXXX/keyring.git
```

- Run the following command to export the private GPG key.

```
gpg --armor --export-secret-keys "x_xxxxxxx@twave.co.jp" > git.gpg.key
```

- Run the following command to copy the private SSH key.

```
cp ~/.ssh/keys/github ./git.ssh.key
```

- Run the following command to add the keys to git and commit them to the repository.

```
git add git.gpg.key git.ssh.key
git commit -m "add gpg/ssh keys for git"
```

- Run the following command to push the commit to GitHub.

```
git push origin main
```