Table of Contents

Using Multiple Github Accounts on One Computer

I am writing for the Windows operating system, but the logic should work for other operating systems as well. With 5 simple steps we will be able to switch between accounts whenever we want.

1. Generate SSH keys for all accounts
2. Adding SSH keys to SSH Agent
3. Adding SSH public keys to Github
4. Creating setting file and preparing Host inputs
5. Using different accounts when copying Github repos

Generating SSH Keys for all accounts

For Windows, you need to be in the user folder. But we don't need to worry about that since we will use the Git Bash command line. We can get things done using Unix code.

```
C:\Users\__SENIN_KULLANICI_ADIN__\
```

1. Open Git Bash and type the following codes in order.

```
cd ~
mkdir .ssh
# If the folder already exists, the above will give an error. If it already exists, we will use that folder.
cd .ssh
eval `ssh-agent -s`
```

1. Create ssh keys for each account in the relevant folder using the following code.

```
ssh-keygen -t rsa -C "__MAIL_ADDRESS__" -f "__GITHUP_USER_NAME__"
```

The `-C` key here is for comments to help with identification. `-f` is the name of the file where the key will be saved.

The mail address you write here should be the non-emailable mail address that github defines for us. It is not mandatory, but it is better not to use our main mail address in any git process to avoid confusion.

> After adding the keys, there should be a private and a public file for each account in the **.ssh** folder.
>
> The public key will have the extension `.pub` and the private key will not have an extension. Both will have the same name as the one after `-f`.

Once you have created a key for each of your accounts, you can continue.

Adding SSH Keys to SSH Agent

While still in .ssh on the command line, you can add the switches to the agent with the following command.

```
ssh-add __GITHUP_USER_NAME__
ssh-add __GITHUP_USER_NAME__
```

Adding SSH Public Key to Github

Open the resulting `.pub` files with any text editor and copy the contents.

https://github.com/settings/keys by going to Add from **New SSH key**. You can write whatever you want in *Title*.

Creating and preparing a Config File

1. Go to Bash and enter the command `touch config`.
2. The above code will create a file named **config** in that folder without extension. Open that file with any text editor and paste the following into it.

```
#__GITHUP_USER_NAME__ account
Host github.com-__GITHUP_USER_NAME__
    HostName github.com
    User git
    IdentityFile ~/.ssh/__GITHUP_USER_NAME__

#__GITHUP_USER_NAME2__ account
Host github.com-__GITHUP_USER_NAME2__
    HostName github.com
    User git
    IdentityFile ~/.ssh/__GITHUP_USER_NAME2__
```

Edit the username parts to suit you. You can increase the number as much as you want.

Cloning Github Repos Using Different Accounts

You can clone with the command below or you can clone directly over HTTPS.

```
git clone git@github.com-
```

```
{__GITHUP_USER_NAME__}:{__REPO_OWNER_USER_NAME__}/{__REPO_NAME__}.git
```

You need to set user settings for each repo separately to determine which account will be used for commits. Global settings can only be used for a single account. Always do the following for all your existing repos and any repos you clone in the future.

```
    git config user.email "__GITHUP_MAIL_ADRESI__"
    git config user.name "__GITHUP_USER_NAME__"

    git config user.email "__GITHUP_MAIL_ADRESI_2__"
    git config user.name "__GITHUP_USER_NAME2__"
```

Of course you can increase the number.

Finally, we need to add remote origin to be able to use **pull** and **push**. Run the following command for each repon.

```
git remote add origin git@github.com-
__GITHUP_USER_NAME__:__GITHUP_USER_NAME__

git remote add origin git@github.com-
__GITHUP_USER_NAME2__:__GITHUP_USER_NAME2__
```

Source:
https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

Taken from UCH Wiki.