IAIK TU Graz **Teaching Wiki**

SSH Setup

If you're taking part in a course at the IAIK you have to upload an ssh **public** key to https://git.teaching.iaik.tugraz.at/ [https://git.teaching.iaik.tugraz.at/] in order to get access to your remote account. If you don't have a ssh-keypair yet, or want to create a new one for use with your repository use (we assume you want to name your key `sweb`, feel free to find a better name):

```
ssh-keygen -t rsa -b 4096 -f ~/.ssh/sweb
```

Please ensure the private key has file access modes `600` (change with `chmod 600 <file>`).

And enter some passphrase (an empty passphrase is **not** recommended). You will then have the files `sweb` and `sweb.pub` in the .ssh-directory in your home directory. The **public** key is the one with the extension `.pub`.

Now we will inform our ssh-agent about the new key:

```
ssh-add ~/.ssh/sweb
```

Throughout the rest of the article we will use a shortcut for the repository domain `iaik-git`.

In addition to that, you have to tell ssh, that it should use this particular key for the remote machine, by editing `~/.ssh/config` and adding:

```
Host iaik-git
  User user
  IdentityFile ~/.ssh/sweb
  HostName repo-domain
```

For example:

```
Host iaik-git
  User git
  HostName git.teaching.iaik.tugraz.at
  IdentityFile ~/.ssh/sweb
```

Important: If the file `~/.ssh/config` does not exist, create it first. Also, do **not** replace the User git by your own username.

After you have uploaded the public key to https://git.teaching.iaik.tugraz.at/ [https://git.teaching.iaik.tugraz.at/] and set up your `~/.ssh/config`, you can start using git.

Using git

It is strongly advised to make oneself familiar with git to be able to use it effectively. Especially the ease of branching and merging comes as a big aid when developing different features at the same time. Although this tutorial was not written to teach you how to use git, a few quick words are probably worth mentioning. We will assume that you have already installed git, either with your operating system's packet manager or manually. To get the source code from the location described above, open a shell, go to the directory that you want to use as development directory and enter

```
git clone git@iaik-git:REPOSITORY_NAME.git .
```

This will then initialize a local copy of your remote repository in the current directory. Now you can start editing files to your heart's content. When you want to commit your changes to the repository, type

```
git commit -a
```

If you have added any new files, you will first have to add them to the repository with

```
git add <filename>
```

For more detailed usage of git, and how to restore older revisions, make branches, merge them, and so on, have a look on the internet, there is a multitude of git tutorials available. Two other, important things should be mentioned here, though.

First, you are required to configure git to use your real name and student e-mail address, so that you can be properly identified as the author of your commits. To do this, use

```
git config --global user.name "Max Mustermann"
git config --global user.email "max@mustermann.com"
```

If you're already familiar with merging and rebasing you can try configuring your git in a way such that pull **always** does a rebase instead of a merge (`–global` for system-wide):

```
git config --global branch.autosetuprebase always
```

Upstream

Upstream repositories are used to distribute patches, etc.

For “Systemnahe Programmierung” you are expected to use this upstream repository:
https://extgit.iaik.tugraz.at/snp/upstream.git [https://extgit.iaik.tugraz.at/snp/upstream.git]

In case of SWEB development you will use the following upstream repository: **https://github.com/IAIK/sweb.git [https://github.com/IAIK/sweb.git]**
If there are any patches for SWEB bugs, they will be provided through the Github upstream repository.

You can create a new remote location for this repository:

```
git remote add upstream <upstream URL>
```

When you are done with that, you can pull the upstream patches into your repository with

```
git pull upstream master
```

Branching

As we recommend working in feature branches we want to show you one more command:

```
git checkout -b my_feature
```

This creates a new branch `my_feature` based on the current branch. Using the command

```
git push origin my_feature
```

you can make your new branch available in your `origin` repository.

Next Chapter

Building SWEB (General / Linux)