UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

Projektmanagement & Tools

Exercise 04

Dr. Marc Kurz

HAGENBERG | LINZ | STEYR | WELS

Exercise

- **Setup (1/2)**
  - Work in groups of 2 (Member1, Member2)
  - **Watch each other** when you are doing stuff
  - Go to https://github.com
  - Each member creates a free account (if you do not already have one)
  - Member2, create a new public repository
    - Name your repo → „Tools2018Lastname1Lastname2“
    - Grant your teammate access (→ repo page, settings, collaborators)
  - Make sure you have GIT installed
    - https://git-scm.com/book/en/v2/Getting-Started-Installing-Git

Exercise

- **Setup (2/2)**
  - Generate and/or upload your SSH key (https://help.github.com/articles/connecting-to-github-with-ssh/)

1. **Generating a new key (if you do not have a key):**
   - From the command prompt in Windows or Terminal in MacOS and Linux run this command: `ssh-keygen -t rsa -C "YOUR EMAIL"` (Note: Email should match the mail you used in the previous steps)
   - You will be prompted to give the key a filename. Accept the default name by just hitting Enter
   - Passphrase can be skipped, too…
2. **Copy public key (id_rsa.pub) into your Github Profile:**
   - In Windows, the key pair will appear at C:\Users\name\.ssh by default. In MacOS and Linux, it will be in ~/.ssh
   - Open the public key file (default name is id_rsa.pub) in a text editor, seletct all and copy (Ctrl + A and Ctrl + C)
   - Log on to your GitHubaccount. Got to settings → SSH keys and add the copied key.

Exercise

- **Exercise Description (1/7)**
  - Step 1
    - Member1: Create a new folder called „GIT Repository“ somewhere in your filesystem → this will be the root directory of our repo
    - Member1: Navigate to the folder and use „git init“ to turn the folder into a local git repo
    - Member1: add the remote repo using

      `git remote add origin https://github.com/xxxx/xxxx.git`
    - Member1: Create a .gitignore file (ignore class files and bin, gen folder)
    - Member1: Use „git status“… what do you see?
    - Member1: Add a file to version control using „git add“
    - Member1: What is the status now?
    - Member1: commit the changes

      `git commit –m „useful commit message“`
    - Member1: Push the changes to the remote repo

      `git push –u origin master` (the „-u“ tells git to track the remote branch and allows later use of „git push“ without specifying the remote)

Exercise

- **Exercise Description (2/7)**
  - Step 2
    - Member2: Clone the repository
    - Member2: Create a folder „username1_username2_android“
    - Member2: what is the status? Why does the folder not show as an untracked file?
    - Member2: create a dummy android (or standard eclipse) project inside the folder. There should be at least one .java file
    - Member2: what is the status now? Why do you see the folder now?
    - Member2: add the project using the „git add“ command, commit and push
    - Member1: Pull the changes

Exercise

- **Exercise Description (3/7)**
  - Step 3 – Modifying different parts of the same file
    - Member1: add some code to an existing method in the Java File
    - Member2: write a new method in the Java File (leave it empty)

      ```
      public void thisIsANewMethod() { }
      ```
    - Member1: add, commit, push
    - Member2: add, commit, then pull the changes of Member1. Was the conflict resolved automatically? What is the status? Why are you ahead 2 commits when you only called commit once?
    - Member2: push the merged changes
    - Member1: pull

Exercise

- **Exercise Description (4/7)**
  - Step 4 – Modifying the same part of a file
    - Member1 & Member2: add a different line of code to the method that has been created previously (create a potential conflict)
    - Member2: add, commit, push
    - Member1: DO NOT COMMIT, only pull. What is the problem and how can you solve it?
    - Member1: add, commit and pull again. What is the problem now? Why do conflicts occur?
    - Member1: Take a look at your Java File → resolve conflicts, commit. push
    - Member2: pull
    - What is the status of each local repo? Are there any diferences between the members? Why/ Why not?

Exercise

- **Exercise Description (5/7)**
  - Step 5 – Branches

    Member1 will perform a „critical bugfix“ while Member2 will continue working „normally“ on the master branch and merge the bugfix into the master branch as soon as the fix is completed
    - Member1: create a branch called „bugfix“
    - Member1: Are you still on the master branch? If yes, switch to the newl created branch. Note: „git push“ alone will still push to the master branch. You have to specify the remote repo you want to push to („git push origin branchname“). The same applies to „git pull“.
    - How can you let git know to which branch to push to hen only using the „git push“ command without any further parameters?
    - Member1: work on the Java File and add some lines in the previously created method that „fixes the bug“. Add, commit, pull, push for each line you write
    - Why is it important to pull before each push?
    - Member2: work on the Java File and add some LoC in various methods. Add, commit, pull, push for each line you write

Exercise

- **Exercise Description (6/7)**
  - Step 5 – Branches

    This of course does not make much sense in this example, but simulates continuous workflow.
    - Why are you not seeing any conflicts or file changes when pushing or pulling (even though both members are working on the repo)?
    - Member1: Integrate your bugfix branch into the master branch.
      - Switch to the master branch
      - Get the latest state of the remote repo master branch
      - Merge your bugfix branch into the master branch
      - Fix merge conflicts
      - Add, commit, push to master
    - Member2: pull → you should now have all changesy made by the „bugfix“

Exercise

- **Exercise Description (7/7)**
  - Step 6 – Pull Request
    - Find another team
    - Exchange information about your repos
    - Now, you should perform changes to another teams code and ask them to include the changes you made in their repository. This is called a „pull request“, more information on how to do this can be found here:

      http://stackoverflow.com/a/14680805/1590502

      https://help.github.com/articles/using-pull-requests/
    - What is the use of Pull Requests?

Exercise

- **Tasks**
  - Submit an exercise protocol (document what you did, describe problems and solutions)
  - Answer the questions asked in the exercise
  - Provide screenshots (when adequate)
  - Show your commit history (e.g. by screenshots)
  - Provide the link to your repository and the group members in the submission
- **Notes**
  - Google is your friend ;-)
  - Use GIT in the **command line** for this exercise