Exercises about GitHub Scenarios

- Solve them in Visual Studio and GitHub.
- For most scenarios you need more than 1 user.
- Solve them within your team.

Exercise GitHub.01

- Person1 creates a working project. Keep this simple.
- This project is placed in a GitHub repository (web based).
- For person1, this project is local and web based.
- Person1 gives in GitHub person2 (or more) the right to collaborate in the repository.
- All other persons (person3, …) makes sure that they have the same project also locally.
- The result must be that all persons have a local project, all connected to the same web based project.

Notes

Notes

....

....

....

....

....

....

....

Exercise GitHub.02

	The starting point is the endpoint of GitHub 01. *This exercise is about committing one change at the time.* *Put your changed application into the Master Branch with Commit.* *Push and Pull (Synchronize) when you need synchronization with the changes of other project contributors.*

- Person1 changes something in the working project.
  - Do not use a branch here. Just change it directly on the master branch.
  - Do not use an issue in GitHub.
- This change is committed and pushed to the branch in GitHub (web based).
- All other persons makes sure that they have the change locally. You can do this with Fetch (check if there are changes) and Pull.

Extra exercise

- Redo this, but another person changes something.

Exercise GitHub.03

	The starting point is the endpoint of GitHub 01 or 02. *This exercise is about committing multiple changes (by different contributors), but in different files.* *Put your changed application into the Master Branch with Commit.* *Push and Pull (Synchronize) when needed. Look very carefully to the messages on the screen.*

- Every person adds a different file to the local project.
    - Do not use a branch here. Just do every change on the master branch.
    - Do not use an issue in GitHub.
- Then all the persons do a Push and a Pull at the same time. Start with 2 computers. Redo it after this with 3 or 4 computers.
- These changes are committed to the branch in GitHub (web based).
- All other persons makes sure that they have all the changes locally.

	Did you succeed in doing this, in one time shot? Can you explain why? • *Everybody has the correct latest version in one shot (only 1 push and pull is needed for everybody).* • *Everybody has the correct latest version in more shots (multiple pushes and pulls were needed for everybody).*

	Show to each other what message you have on the screen. *Learn from the different situations.*

Notes

....

....

....

....

....

....

....

Exercise GitHub.04

	The starting point is the endpoint of GitHub 01, 02 or 03. *This exercise is about committing multiple changes (by different contributors), but in the same file.* *Put your changed application into the Master Branch with Commit.* *Fetch, Push and Pull (Synchronize) when needed.*

- Every person adds a different method to the project, but in the same file.
    - Do not use a branch here. Just change it directly on the master branch.
    - Do not use an issue in GitHub.
- Person1 commits and pushes to the master branch. (Try also fetch)
- Person2 commits and pushes to the master branch. (Try also fetch)
- All other persons too, after each other.
- This change is committed to the branch in GitHub (web based).
- All other persons makes sure that they have all the changes locally.

- Redo the exercise by pushing and pulling at the same time.

	Show to each other what you have on the screen.

Notes

....
....
....
....
....
....
....

Exercise GitHub.05

	The starting point is the endpoint of GitHub 01, 02, 03 or 04. *This exercise is about committing multiple changes (by different contributors), but in the same method.* *Put your changed application into the Master Branch with Commit.* *Push and Pull (Synchronize) when needed.*

- Every person adds a line of code inside 1 method to the project.
  - Do not use a branch here. Just change it directly on the master branch.
  - Do not use an issue in GitHub.
- The end goal is that all added lines are now in that method and that everybody has the latest version locally.

Notes

....

....

....

....

....

....

....

Exercise GitHub.06

- Person1 does 2 changes, every change in a different file.
- But only 1 change will be committed to the branch.
    - You can use Stage to do this.
- The end goal is that all committed changes are in the latest version locally for everybody of the team.

	Look in the Git Repository in the master branch for the complete history of the solution, project, code.

Notes

....

....

....

....

....

....

....

Exercise GitHub.07 (Working with branches)

	Remove all the projects and remove the GitHub repository. *Repeat Exercise GitHub.01.* *From here on we will do the same exercises but with Branches, commits to that branches with pull requests.* *Put your changed application into a Branch with Commit. Push and Pull (Synchronize) when needed. A person must validate your changes, so use Pull Requests. Everybody works with the new branch or even with several branches. You, as team, decide on how you will work for this exercise.* *Make sure you, personally, work in the correct branch as decided by the team, and then when ready, and tested, merge it to Master.* *You have to work with pull requests, meaning, somebody must "validate" / "check" the code you want to add to the branch.* *The code in the Master branch must work. What you do in the other branches, I don't care. You can experiment as much you want, but the Master has working code in it.*

- Repeat Exercise GitHub.02, but with using branches, you have to work with pull requests and when it works, you merge into master.
- Repeat Exercise GitHub.03, but with using branches, you have to work with pull requests and when it works, you merge into master.
- Repeat Exercise GitHub.04, but with using branches, you have to work with pull requests and when it works, you merge into master.
- Repeat Exercise GitHub.05, but with using branches, you have to work with pull requests and when it works, you merge into master.
- Repeat Exercise GitHub.06, but with using branches, you have to work with pull requests and when it works, you merge into master.

Notes

....

....

....

....

....

....

....

Exercise GitHub.08 (Working with issues)

	Remove all the project and remove the GitHub repository. *Repeat Exercise GitHub.01.* *From here on we will do the same exercises but with Branches, commits to that branches with pull requests and you use the "issues" functionality in GitHub.* *Consider "issues" as tasks that must be done, to get somewhere.* *The goal is to keep track of your changes with issues in GitHub. Every change (task) by every person must have an issue, and you add one or more pull requests to that issue. Issues are closed when everything is "live" on the correct branch.* *This is the same routine as you did in the GitHub lab.* *Put your changed application into a new Branch with Commit. Push and Pull (Synchronize) when needed. Merge the branch to Master. Make sure that the issues are closed when everything works fine. (The team decides what branch will be used to try things out)* *Discuss on how your workflow will be. It is this process that you will use in your TimeSeries project.*

- Repeat Exercise GitHub.02 but with using branches and issues.
- Repeat Exercise GitHub.03 but with using branches and issues.
- Repeat Exercise GitHub.04 but with using branches and issues.
- Repeat Exercise GitHub.05 but with using branches and issues.
- Repeat Exercise GitHub.06 but with using branches and issues.

Notes

....
....
....
....
....
....
....

Exercise GitHub.09 (Experiment)

- See how the tool reacts when you try to do things that are conflicting each other.
- See how the tools reacts when you work with different branches.

Example

- One person changes a file.
- The other person deletes the file.
- …

	Whatever you do with GitHub and the projects later. Make sure you are knowing what you are doing. *When you are not sure what the end result will be, try it first out in a test project, before doing this in a larger project, with the possibility to destroy months of work.* *This exercise has the goal that you experiment with the tool GitHub or any source control you want to use.*

Example for branches

- One person works in one branch.
- Another person works in another branch.
- Later you merge both branches into master.
- …

	Try to predict the behaviour before you do it.

Notes

....

....

....

....

....

....

....