# Lecture 1

* Github
  + Used for version control.
  + 创建新repository:
    - 网页上操作
    - 在文档里Git init – git add – git commit -m ‘message’ – git push -u origin branch\_name(origin是本地仓库对远程仓库对别名)(-u后就可以直接push，不需要加后缀)
  + Clone:
    - Cd folder – git clone https://
  + Commit changes:
    - Git status查看文件状态
    - Git diff查看变化
    - Git commit -a -m ‘message’
      * -a:跳过git add, 直接commit修改后的所有文件tracked files（不包括新创建的文件untracked,没有被git add过）
      * -m: commit message
  + Push local changes to remote repository:
    - Git push
  + 创建新文件：
    - Terminal中touch file\_name
    - Jupyter notebook：得先打开jupyter，再在网页中新建
  + .gitignore:
    - 创建：Touch .gitignore
    - 不会在git status里显示untracked,也不会被git commit .
  + Branch:
    - 创建并转到新分支：git switch -c new\_branch
    - 查看自己现在的分支：git branch
    - git checkout -b my\_feature\_branch
    - 删除合并的旧分支：git branch -d branch\_name
  + branch合并到main：
    - pull request
    - merge pull request
  + 更新本地文件：
    - Git checkout main
    - Git pull origin main
  + 处理merge conflict：
    - 先更新自己的文件：
      * Git checkout main
      * Git pull origin main(把其他人已经merge的文件pull到本地)
    - 切换回有conflict的分支：git checkout branch\_name
    - 变基：
      * Git rebase main
      * 可能会报错conflict,手动在报错文件中修改
      * 标记错误已解决，继续rebase:
        + Git add 文件名
        + Git rebase –continue
      * 如果还有冲突，重复上述步骤
    - Git push origin branch\_name --force-with-lease(可能需要安全的强制推送)
    - 再次pull request
* Virtual environments:
  + Venv:
    - 创建一个新的虚拟环境：python -m venv directory\_name
    - 激活环境：source directory\_name/bin/activate
    - 不激活：deactivate
    - Pip freeze/ pip list列出所有已安装的包（pip freeze > requirements.txt）
  + Anaconda:
    - 创建：conda env create -f environment.fml
    - 激活: conda activate directory\_name
    - 确认包是否安装成功：pip show package\_name
    - 安装包：pip install -r requirements.txt
    - 不激活：conda deactivate