GIT

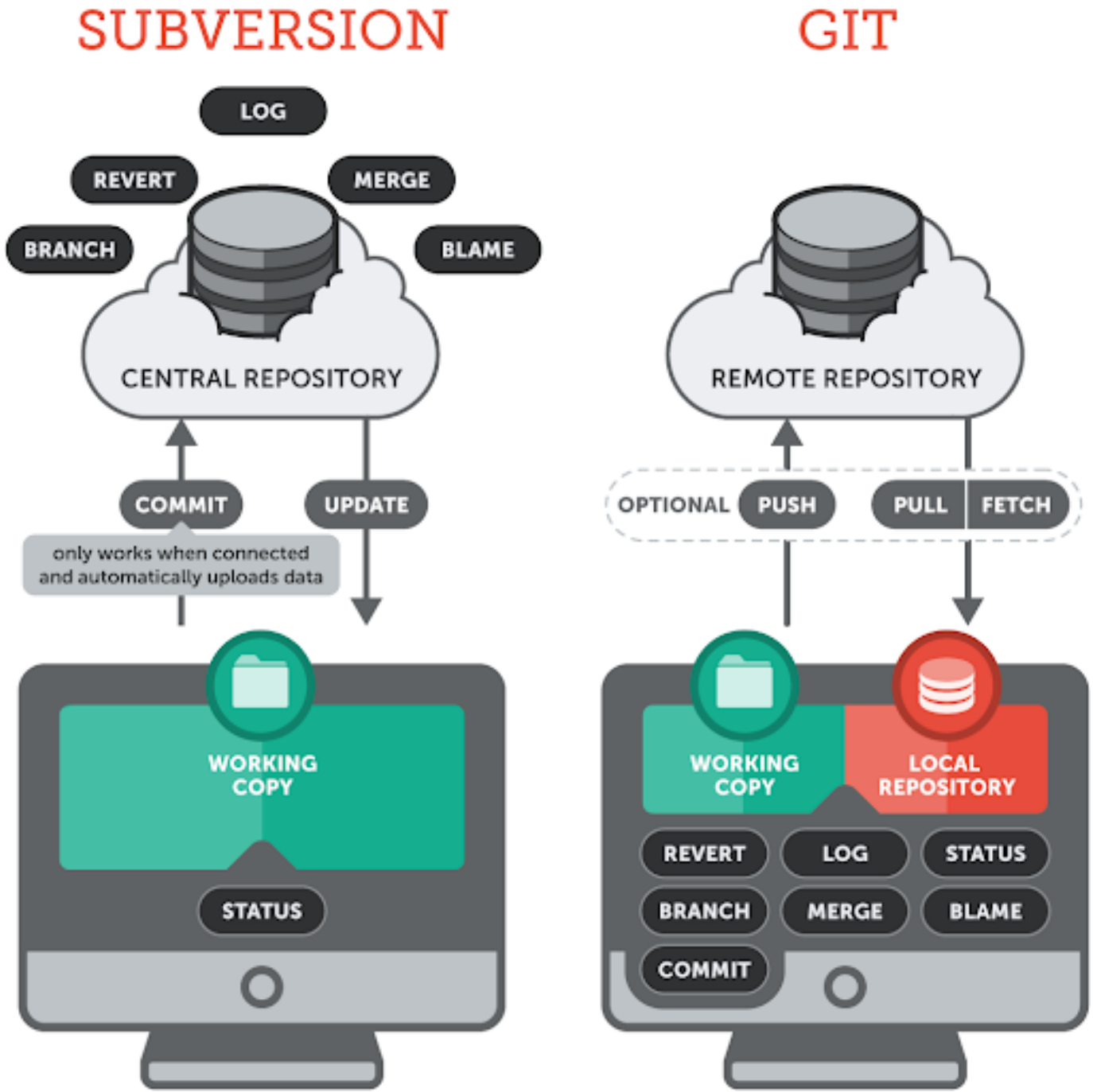
Git ismertető

- nyílt forráskódú
- elosztott verziókezelő szoftver
- offline történik a fejlesztés
- mindenki saját másolattal rendelkezik

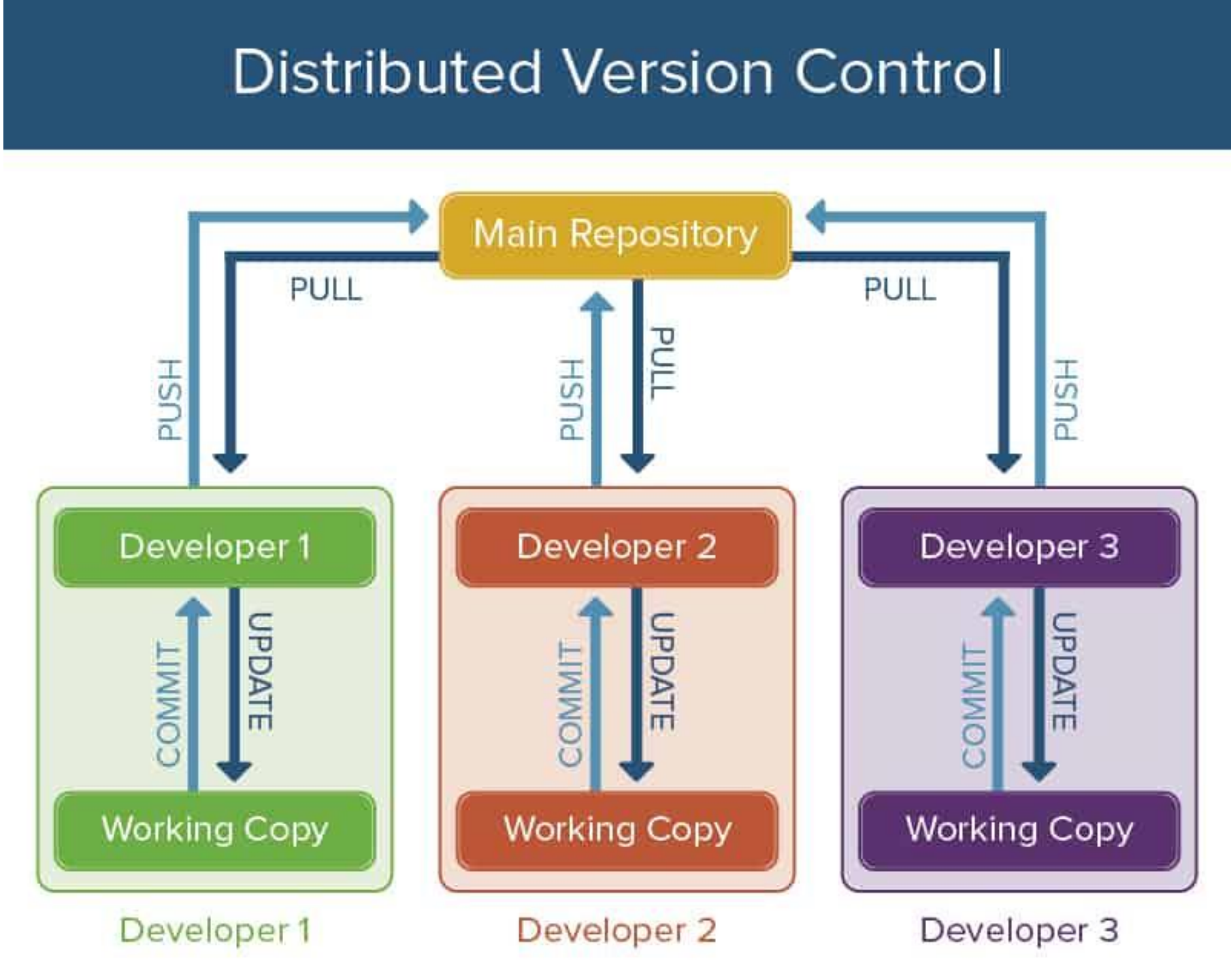
Git config

- `Verziószám lekérdezése: git -v VAGY git --version`
- `git config --list`
- `git config --global user.name "John Doe"`
- `git config --global user.email "johndoe@example.com "`
- `git config --global core.editor "code --wait"`

--wait: parancssori szövegszerkesztőként lesz futtatva, tehát a parancssor blokkolva lesz)


SUBVERSION
LOG
REVERT
MERGE
BRANCH
BLAME
CENTRAL REPOSITORY
COMMIT
UPDATE
only works when connected
and automatically uploads data
WORKING
COPY
STATUS
GIT
REMOTE REPOSITORY
OPTIONAL
PUSH
PULL
FETCH
WORKING
COPY
LOCAL
REPOSITORY
REVERT
LOG
STATUS
BRANCH
MERGE
BLAME
COMMIT

Git


Distributed Version Control
Main Repository
PULL
PUSH
PULL
PUSH
PULL
PUSH
Developer 1
COMMIT
UPDATE
Working Copy
Developer 2
COMMIT
UPDATE
Working Copy
Developer 3
COMMIT
UPDATE
Working Copy
Developer 1
Developer 2
Developer 3

Git workflow

WORKING DIRECTORY
WORKING DIR
WORK DIR
WORKSPACE
WORKING FOLDER
WORKING COPY
WORKING TREE

project/

STAGING AREA
STAGING DIRECTORY
STAGE
STAGING
INDEX
CACHE

project/.git/index

REPOSITORY
LOCAL REPO
LOCAL DB
REPO
GIT DIRECTORY
.GIT FOLDER
HISTORY

project/.git/objects

Git workflow

WORKING DIRECTORY
WORKING DIR
WORK DIR
WORKSPACE
WORKING FOLDER
WORKING COPY
WORKING TREE

STAGING AREA
STAGING DIRECTORY
STAGE
STAGING
INDEX
CACHE

LOCAL REPOSITORY
REPOSITORY
LOCAL DB
REPO
GIT DIRECTORY
.GIT FOLDER
HISTORY

UPSTREAM REPOSITORY

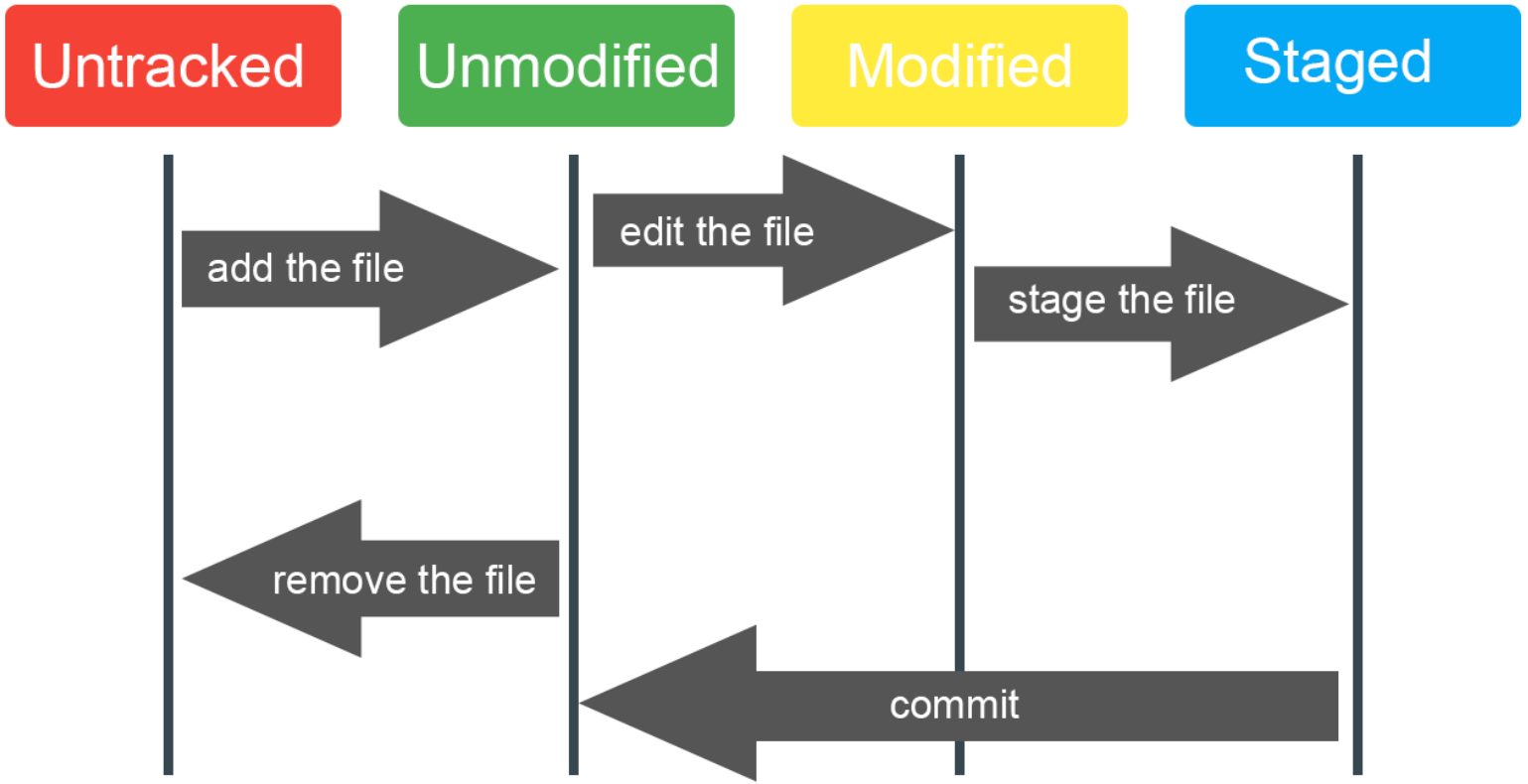
Github

- private/public repo
- licence
- gitignore
- REAMDE.md

Alapparancsok

- **git clone** – távoli repo klónozása
- **git init** – új repo inicializálása
- **git status** – work tree fájljainak státusza
- **git log --graph --oneline --all** - commit history
- **git add** – fájlok mozagatása a Stage-be
  - fileName
  - .
  - folderName/*
- **git rm <filename>** - fájlok törlése a work tree-ből és a stage-ből
- **git rm --cached <filename>** - csak a stage-ből törli
- **git commit --m „message”** – fájlok mozgatása a stage-ből a db-be

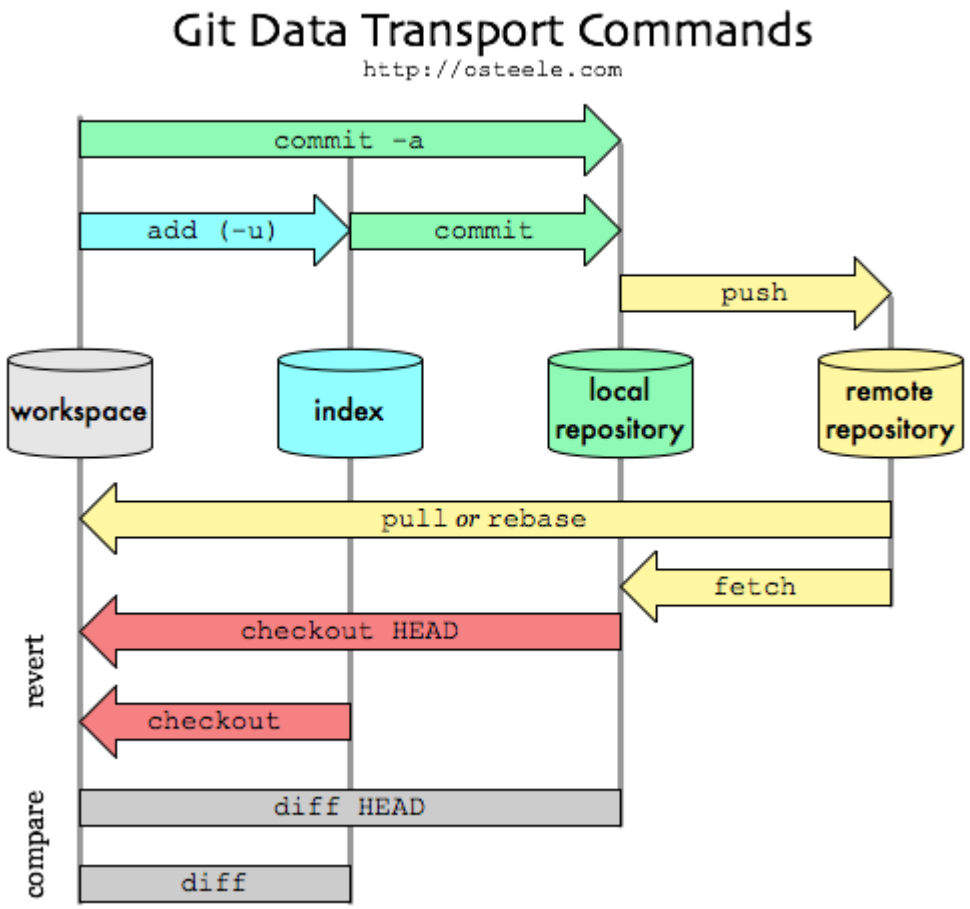
File status


Untracked
Unmodified
Modified
Staged
add the file
edit the file
stage the file
remove the file
commit

Alapparancsok

- git pull
- git push
  - --force
  - --tags
- git merge
- git fetch

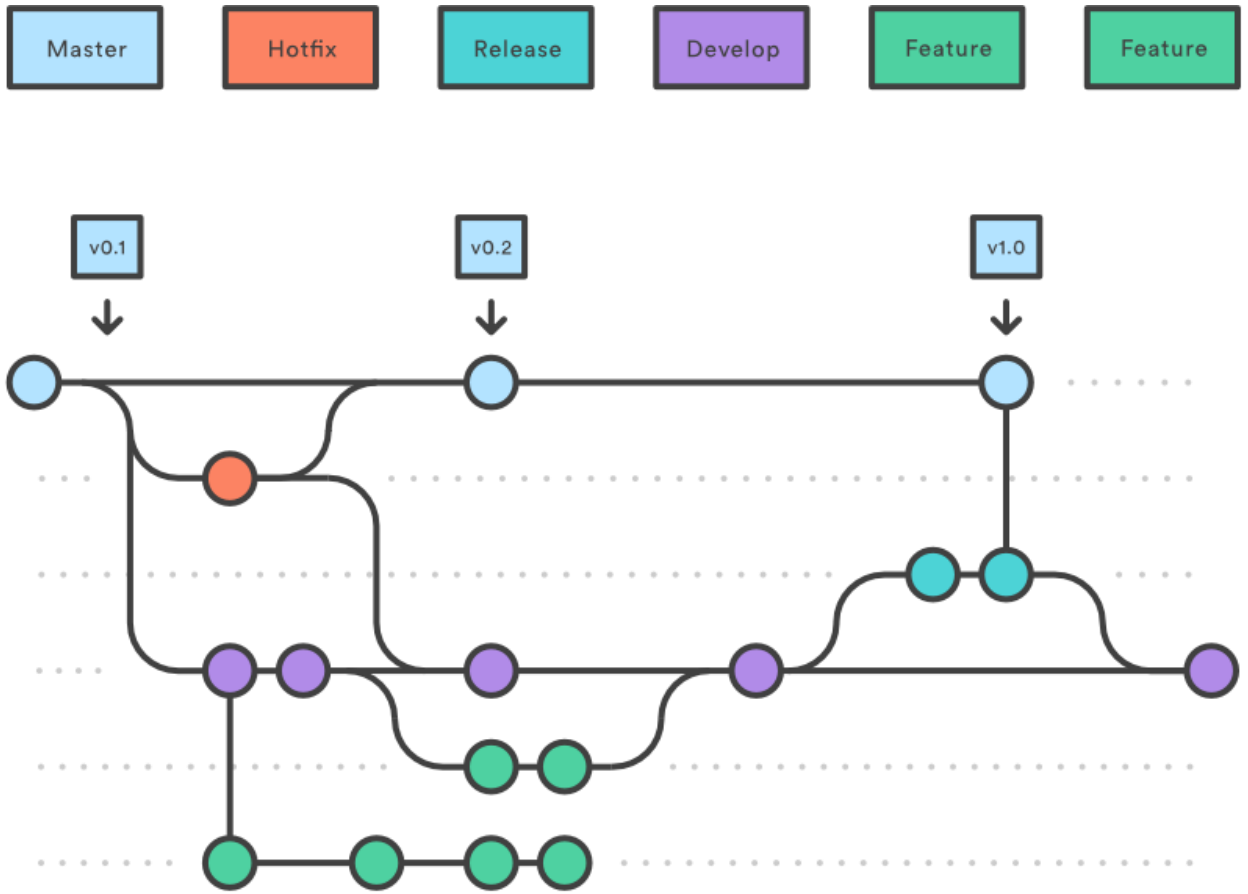
Git commands



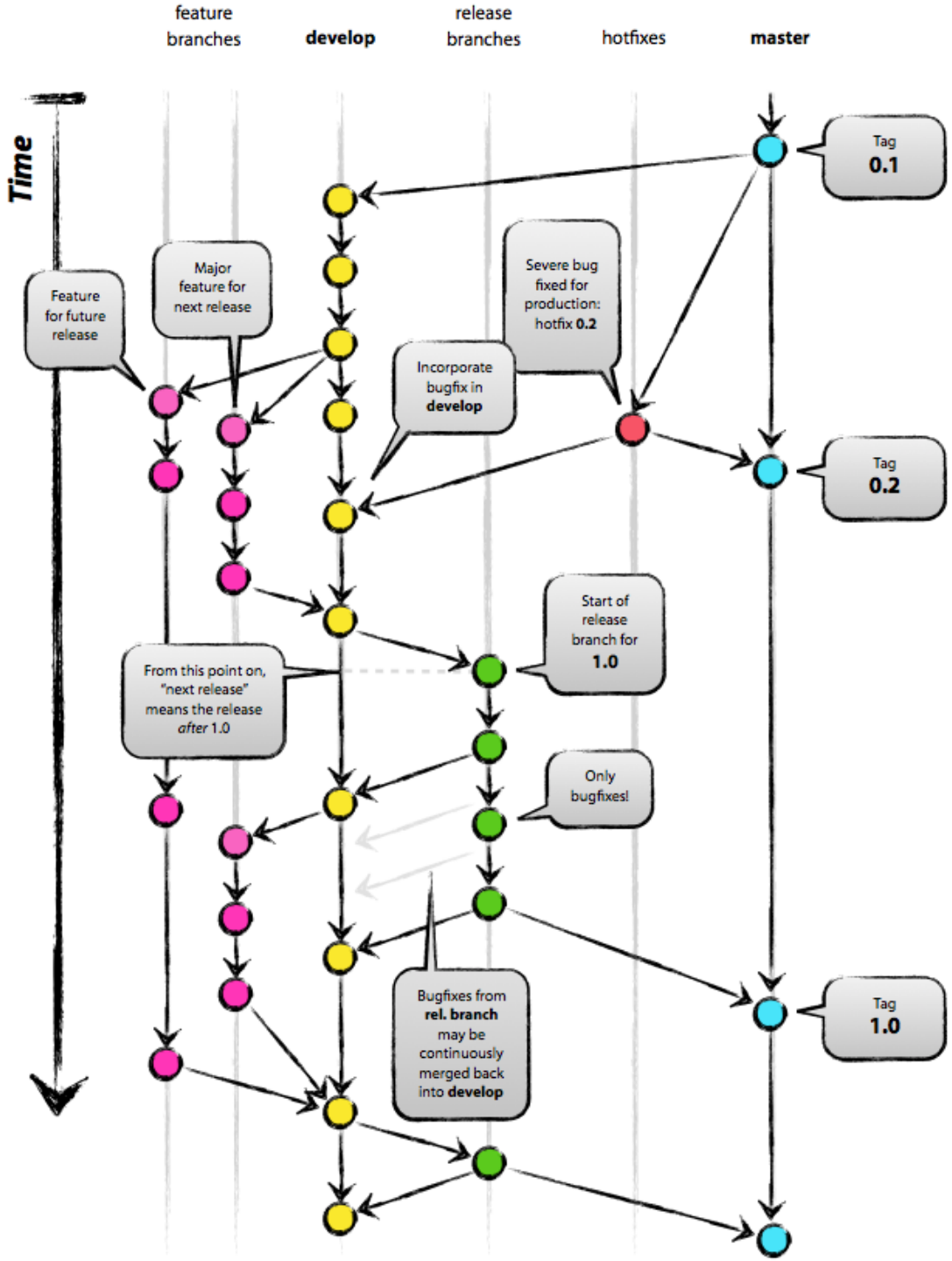
Branches

- `git branch`
- `git branch <branchname>`
- `git checkout branchname`
- `git push --set-upstream origin develop`

Gitflow Workflow



Gitflow Workflow



Branch-ek

- **main (master)**: a release history van benne
- **develop**: a **feature** brancheket integráljuk bele
- **feature**: minden feature-nek van egy-egy kitüntetett branch-e. Ha kész merge-eljük a **develop**-ba aztán töröljük.
- **release**: amikor a következő kiadáshoz kész az összes feature és be vannak merge-elve a **develop**-ba akkor bemerge-ljük a release-be. Ha van hiba javítjuk, ha minden kész bemerge-eljük a master-be.
- **hotfix**: Ha hiba van azt külön ágon javítjuk. Utána merge-eljük vissza a **master**-be és a **develop**-ba is. Utána töröljük.

Develop branch

- **Develop**: a feature brancheket integráljuk bele
  - **git branch develop**
  - **git push -u origin develop**

Feature branch

- **Feature**: minden feature-nek van egy-egy kitüntetett branch-e. Ha kész merge-eljük a Develop-ba aztán töröljük.

- git checkout develop
- git checkout -b feature/featurename

Release branch

Lépések:

- `git checkout develop`
- `git checkout -b release/0.1.0`
- `git checkout master`
- `git merge release/0.1.0`
- `git tag -a v.0.1.0 -m "version v.0.1.0„`
- `git push --tags`
- `git branch -D release/0.1.0`

Hotfix branch

- **Hotfix**: Ha hiba van azt külön ágon javítjuk. Utána merge-eljük vissza a **Master**-be és a **Develop**-ba is. Utána töröljük.
- git checkout master
- git checkout -b hotfix_branch

- git checkout master
- git merge hotfix_branch
- git checkout develop
- git merge hotfix_branch
- git branch -D hotfix_branch

checkout vs reset vs revert

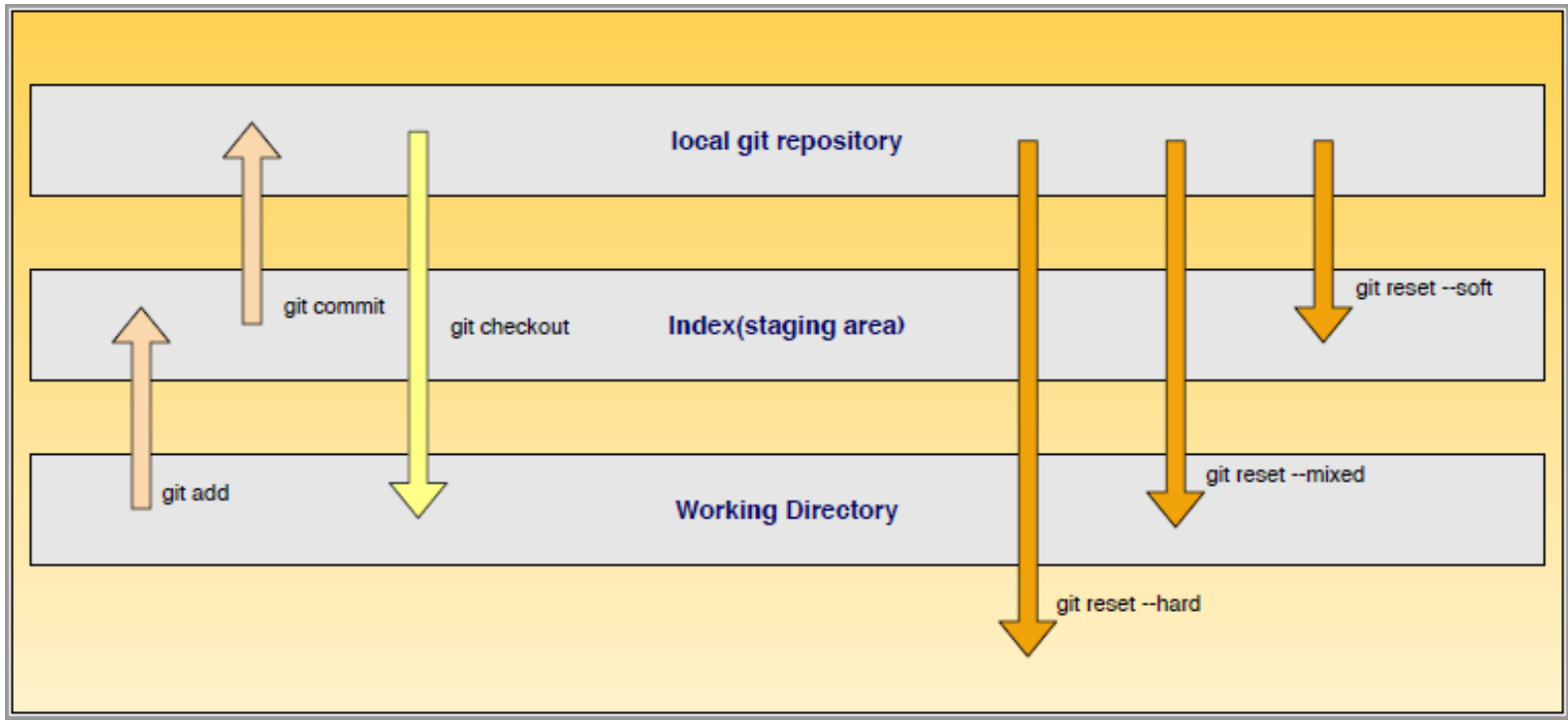
Command	Scope	Common use cases
git reset	Commit-level	Discard commits in a private branch or throw away uncommited changes
git reset	File-level	Unstage a file
git checkout	Commit-level	Switch between branches or inspect old snapshots
git checkout	File-level	Discard changes in the working directory
git revert	Commit-level	Undo commits in a public branch

checkout vs reset vs revert

- checkout <commitID>:
  - egy adott commit-ra lehet visszaállni
  - pl kísérletezés, ellenőrzés alkalmával, egy új branch-be érdemes
    - git checkout -b newBranchName commitID
- revert <commitID>:
  - Egy vagy több commit módosításait lehet visszavonni
  - publikus branch-en használjuk
  - pl ha csak egy adott commit-ban lévő módosítás okoz bugot
    - egy commit: git revert HEAD vagy commitID
    - több commit: git revert HEAD~3..
    - --no-commit

checkout vs reset vs revert

- reset:
  - Commitok és fileok törlésére szolgál
- git reset <hash> vagy HEAD~number
  - --hard
  - --mixed
  - --soft



fetch + merge vs pull

- **A**
  - git fetch
  - git diff master origin/master
  - git merge

- **B**
  - git pull

Duplikált bejegyzés

- git config --global –unset-all configName

  VAGY

- git config --list --show-origin

  és manuálisan szerkesztem a file-t

- 1 fájl összehasonlítás különböző brancheken:
  - git diff branch1 branch2 -- filename

Linkek

- https://www.atlassian.com/git/tutorials/setting-up-a-repository
- https://desoft.hu/downloads/git/git_v1.0.pdf