Introduction to Git

Arda Onur

1) What is Version Control?

- A version control system is a program or set of programs that tracks changes to a collection of files.
- Another name for a VCS is a <u>software configuration management (SCM) system</u>

2) Distributed version control

• Git is *distributed*, which means that a project's complete history is stored both on the client *and* on the server.

4) Git Terminology

- Working tree: The set of nested directories and files that contain the project that's being worked on.
- Repository (repo): The directory, located at the top level of a working tree, where Git keeps all the
 history and metadata for a project.
- **Hash**: A number produced by a hash function that represents the contents of a file or another object as a fixed number of digits. Git uses hashes that are 160 bits long.
- Object: A Git repo contains four types of objects, each uniquely identified by an SHA-1 hash. A <u>blob</u> object contains an ordinary file. A *tree* object represents a directory; it contains names, hashes, and permissions.
- Commit: Commit means to make a commit object. It means you are committing the changes you have made.
- **Branch**: A branch is a named series of linked commits. The most recent commit on a branch is called the *head*. The default branch, which is created when you initialize a repository, is called main often named master in Git.
- **Remote**: A remote is a named reference to another Git repository. When you create a repo, Git creates a remote named origin that is the default remote for push and pull operations.

6) What is GitHub?

- GitHub is a cloud platform that uses Git as its core technology. GitHub simplifies the process of collaborating on projects and provides a website .
 - Key features provided by GitHub
 - Issues
 - Discussions
 - · Pull requests
 - Notifications
 - Labels
 - Actions
 - Forks
 - Projects

7) Configuring Git

To configure Git, you must define some global variables: user.name and user.email.

```
Bash
git config --global user.name "<USER_NAME>"
```

```
Bash

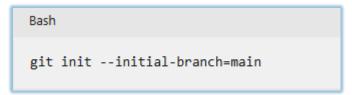
git config --global user.email "<USER_EMAIL>"
```

• Run the following command to check that your changes worked.

```
Bash
git config --list
```

8) Setting up your Git repository

- Create a folder
- Initialize your new repository and set the name of the default branch to main.



Use a git status command to show the status of the working tree.



9) Getting help from Git

Type the following command to get help.

```
Bash
git --help
```

• The command displays the following output .

```
usage: git [--version] [--help] [-C <path>] [-c name=value]
       [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
       [-p | --paginate | --no-pager] [--no-replace-objects] [--bare]
       [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
      <command> [<args>]
These are common Git commands used in various situations:
start a working area (see also: git help tutorial)
            Clone a repository into a new directory
  clone
   init
             Create an empty Git repository or reinitialize an existing one
work on the current change (see also: git help everyday)
           Add file contents to the index
              Move or rename a file, a directory, or a symlink
  reset
             Reset current HEAD to the specified state
             Remove files from the working tree and from the index
examine the history and state (see also: git help revisions)
  bisect
             Use binary search to find the commit that introduced a bug
             Print lines matching a pattern
  grep
             Show commit logs
   log
             Show various types of objects
           Show the working tree status
   status
grow, mark and tweak your common history
  branch    List, create, or delete branches
checkout    Switch branches or restore working tree files
  commit
diff
             Record changes to the repository
             Show changes between commits, commit and working tree, etc
  merge
             Join two or more development histories together
  rebase Forward-port local commits to the updated upstream head
             Create, list, delete or verify a tag object signed with GPG
collaborate (see also: git help workflows)
  fetch Download objects and refs from another repository
             Fetch from and integrate with another repository or a local branch
            Update remote refs along with associated objects
'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
```

10) Resources

https://learn.microsoft.com/en-us/training/modules/intro-to-git/