**Work-Case №1**

**Притула Владислав**



*Git- is software for tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code during software development. Its goals include speed, data integrity, and support for distributed, non-linear workflows (thousands of parallel branches running on different systems)*

*Git was created by Linus Torvalds in 2005 for development of the Linux kernel, with other kernel developers contributing to its initial development. Since 2005, Junio Hamano has been the core maintainer. As with most other distributed version control systems, and unlike most client–server systems, every Git directory on every computer is a full-fledged repository with complete history and full version-tracking abilities, independent of network access or a central server. Git is free and open-source software distributed under GNU General Public License Version 2.*

*Git is a set of command line utilities that allow you to track and record changes to files (usually code, but you can track anything). With it, you can restore old versions of your project, compare, analyze, combine changes and more. This process is called version control. There are many similar version control systems. You may have heard some of them: SVN, Mercurial, Perforce, CVS, Bitkeeper and others.*

*Git works decentrally, which means it is independent of the central server. The data is stored locally in a folder on your hard drive called a repository. However, you can also save a copy of your repository online, allowing a team of people to work on a single code at a time. Sites such as GitHub and BitBucket are used for this purpose.*

*Github service is an effective tool for teamwork*

*on the web-product development project. Possession of skills of use*

*service allows web-developer to work more efficiently with sites. Yes*

*competence is a requirement of employers for modern web-product developers. *

*Git is one of the most efficient, reliable and high-performance systems*

*version control, which provides flexible non-linear development tools based*

*on the branch and merging of branches. To ensure the integrity of history and sustainability*

*to changes in retrospect cryptographic methods are used, it is possible*

*linking digital signatures of developers to tags and committees.*

*The system is designed as a set of programs specifically designed with*

*given their use in scripts. This makes it easy to create*

*specialized version control systems based on Git. For example, Cogito is*

*just such an example of a frontend to Git repositories. And StGit uses Git*

*to manage a collection of patches.*

*The system has a number of interfaces: for example, gitk and git-gui.*

*Remote access to Git repositories is provided by a git daemon, SSH*

*or HTTP server. TCP service git-daemon is part of the Git distribution and is together*

*with SSH the most common and reliable method of access. HTTP access method,*

*despite a number of limitations, very popular in controlled networks,*

*because it allows the use of existing network filter configurations.*

**

*Трегуб Ольга***2. What is a "commit", how does it allow you to track changes in files?**

**A commit in git is an object that contains a reference to a snapshot previously recorded in the index, author metadata, comments, zero or more pointers to other commits that are direct parents of this commit. Depending on the situation, a commit may have the following number of parents: 0 for the first commit; 1 for a normal commit; multiple for a commit resulting from a merge of two or more branches.**

**Creating a commit: When you run the git commit command, Git creates a commit. At the same time, Git calculates the checksum for each subdirectory and stores these tree objects in the Git repository. After that, Git creates an object for the commit, which contains metadata and a pointer to the tree object. This way, Git will be able to recreate the current state when needed.**

**Main actions and commands of gt**

* **git config**

**One of the most commonly used git commands. It can be used to specify settings such as email, username, format, etc. For example, this command is used to set an e-mail address:**

**git config --global user.email адрес@gmail.com**

* **git init**

**This command is used to create a GIT repository. Example of use:**

**git init**

* **git add**

**The git add command can be used to add files to the index. For example, the following command will add a file called temp.txt present in the local directory to the index:**

**git add temp.txt**

* **git clone**

**The git clone command is used to clone the repository. If the repository is on a remote server, the following command is used:**

**git clone name.user@host:/way/to/repository**

**Conversely, use to store local storage:**

**git clone /way/to/repository**

* **git commit**

**The git commit command is used to commit changes to project files. Note that commits do not immediately land on a remote repository. application:**

**git commit –m “The message goes along with the commit”**

* **git status**

**The git status command displays a list of modified files, along with files that have not yet been added to the index or are awaiting commit. Application:**

**git status**

* **git push**

**The git push command is another commonly used git command. Allows you to place changes to the main branch of the remote repository associated with the working directory. Example:**

**git push origin master**

* **git checkout**

**The git checkout command can be used to create branches or switch between them. For example, the following code will create a new branch and switch to it:**

**command git checkout -b <name-branch>**

**To just switch between branches use:**

**git checkout <name-branch>**

* **git remote**

**The command allows the user to connect to a remote repository. This command will display a list of remote repositories currently configured:**

**git remote –v**

**This command will allow the user to connect a local repository to a remote server:**

**git remote add origin <remote.server.address>**

* **git branch**

**The git branch command can be used to display, create, or delete branches. To display all existing branches in the repository, enter:**

**git branch**

**To remove a branch:**

**git branch –d <name-branch>**

* **git pull**

**The git pull command is used to combine changes present in a remote repository into a local working directory. Application:**

**git pull**

* **git merge**

**The git merge command is used to merge a branch into an active branch. Application:**

**git merge <name-branch>**

* **git diff**

**The git diff command is used to detect differences between branches. To identify differences with the base files, use:**

**git diff --base <name-bracnh>**

**The following command is used to view the differences between the branches to be merged before merging them:**

**git diff <branch-source> <branch-goal>**

*Гоголь Анастасія*

**To easily display existing differences, use:**

**git diff**

* **git tag**

**Used to mark certain commits with simple labels. An example is this command:**

**git tag 1.1.0 <insert-commitID-here>**

* **git log**

**Running the git log command to display a list of all commits in the git at the same time with all the available views. Butt to the result:**

**commit 15f4b6c44b3c8344caasdac9e4be13246e21sadw**

**Author: Alex Hunter <alexh@gmail.com>**

**Date: Mon Oct 1 12:56:29 2016 -0600**

* **git reset**

**The git reset command is used to reset the index and working directory to the last commit state. application:**

**git reset --hard HEAD**

* **git rm**

**The git rm command will search for files in the index and work directory. Gagging:**

**git rm namefile.txt**

* **git stash**

**One of the most small git commands is possible. Vona additional help from the saved changes on a time-based basis, so that you do not consume the money in the shop at once. Gagging:**

**git stash**

* **git show**

**For a glance at information about any git about'єkti, vikorist the git show command. For the butt:**

**git show**

* **git fetch**

**Command git fetch allows koristuvachev to deliver all objects from the remote control, as they are not present in the local working catalosis. Butt stocking:**

**git fetch origin**

* **git ls-tree**

**The git ls-tree command wicks to view the object tree at once by name, skin object mode and SHA-1 values. For example:**

**git ls-tree HEAD**

* **git cat-file**

**Use the git cat-file command to check the object type for the SHA-1 value. For example:**

**git cat-file –p d670460b4b4aece5915caf5c68d12f560a9fe3e4**

* **git grep**

**git grep allows the user to search for phrases and words in the contents of the trees. For example, to search for https://sites.google.com/view/kkz in all files, use this command:**

**git grep "https://sites.google.com/view/kkz"**

* **gitk**

**gitk — it is the GUI for the local repository. You can call it by running this command:**

**gitk**

* **git instaweb**

**Using the git instaweb command, you can start the web server associated with the local repository. The browser will also automatically redirect to it. For example:**

**git instaweb –httpd=webrick**

* **git gc**

**To optimize the repository use the command git gc. It will help you remove and optimize unnecessary files:**

**git gc**

* **git archive**

**Command git archive allows the user to create a .zip or .tar file containing the components of one of the repository trees. For example:**

**git archive --format=tar master**

* **git prune**

**The git prune command removes objects that do not have any incoming pointers. Application:**

**git prune**

* **git fsck**

**To check the integrity of the git filesystem, use the git fsck command, which will identify any corrupted objects:**

**git fsck**

* **git rebase**

**The git rebase command is used to apply commits on a different branch. For example:**

**git rebase master**