Ministerul Educaţiei al Republicii Moldova

Universitatea Tehnică a Moldovei

Facultatea Calculatoare Informatică şi Microelectronică

Disciplina: Medii Interactive de Dezvoltare a Produselor Soft

Lucrare de laborator Nr. 1

Tema: Version Control Systems și modul de setare a unui server

A efectuat student: **Baltag Maxim**

A controlat lect asist**: Victor Gojin**

Chişinău 2017

**Scopul lucrării**

Version Control Systems

**Sarcinile temei**

De utilizat drept VSC github sau bitbucket pentru a realiza sarcinile:

1. Basic level:

* De inițializat un nou repositoriu;
* De configurat VCS;
* De creat cel puțin 2 branch-uri;
* De făcut commit pe ambele branch-uri (cel puțin 1 commit per branch);

1. Normal level:

* De setat un branch to track a remote origin pe care se va putea face push;
* De resetat un branch la commit-ul anterior;
* De salvat temporar schimbările care nu se vor face commit imediat;
* De folosit fișierul .gitignore;

1. Advanced level:

* De făcut merge la 2 branch-uri;
* De rezolvat conflictele a 2 branch-uri;
* De studiat comenzile git;

1. Bonus point:

* Folosirea tag-urilor pentru marcarea schimbărilor semnificative precum release-ul.

**Realizarea lucrării de laborator**

Lucrarea de laborator se găsește sub numele [MIDPS][TI-154][Lab#1][PruniciAlexandru]

Repozitoriul se găsește pe https://github.com/AlexYooP/MIDPS

VSC utilizat: Github

Sarcinile realizate:

1. Basic level:

* De inițializat un nou repositoriu;
* De configurat VCS;
* De creat cel puțin 2 branch-uri;
* De făcut commit pe ambele branch-uri (cel puțin 1 commit per branch);

1. Normal level:

* De setat un branch to track a remote origin pe care se va putea face push;
* De resetat un branch la commit-ul anterior;
* De salvat temporar schimbările care nu se vor face commit imediat;
* De folosit fișierul .gitignore;

1. Advanced level:

* De făcut merge la 2 branch-uri;
* De rezolvat conflictele a 2 branch-uri;
* De studiat comenzile git;

1. Bonus point:

* Folosirea tag-urilor pentru marcarea schimbărilor semnificative precum release-ul.

**Descrierea lucrării de laborator**

Etapele realizării lucrării de laborator:

* S-a creat un profil pe github.com;
* S-a creat un repozitoriu cu numele MIDPS care se află pe
* https://github.com/Apatpoh;
* În repozitoriu s-a adăugat fișierele .gitignore și README;
* S-a instalat Git Bash cu ajutorul căruia s-au realizat următorii pași;
* S-a generat un public key cu ajutorul comenzii *ssh-keygen*;
* Cheia generată s-a introdus pe profilul din github în *Setting/SSH and GPG keys*;
* S-a copiat din repozitoriu link-ul SSH;
* S-a creat în disk-ul E pe calculator mapa *MIDPS*;
* S-a clonat repozitoriul pe calea *cd /e/MIDPS* cu ajutorul comenzii *git clone ”*link-ul SSH*”*;
* S-a configurat repozitoriul cu numele și adresa electronică respectivă cu ajutorul comenzilor *git config –global user.email ”yourmail@domain.com”* și *git config –global user.name ”YourName”*, unde global arată nivelul la care se face configurația și înseamnă că configurațiile sunt valabile pentru toate proiectele de la utilizatorul curent, dar nu și pentru toți utilizatorii;
* S-a creat folder-urile necesare pentru fiecare laborator;
* S-au încărcat schimbările pe repozitoriu cu ajutorul comenzilor *git add*, *git commit –m ”* și *git push*;
* Crearea a 2 branch-uri, commit pe ambele branch-uri:

S-a creat primul branch cu ajutorul comenzii *git checkout –b branch1*, unde *–b* înseamnă că se crează un branch nou, iar *branch1* este denumirea acestuia. Este o variantă mai scurtă care combină comenzile *git branch* și *git checkout*.

S-a creat un document docx gol cu numele *First commit*.

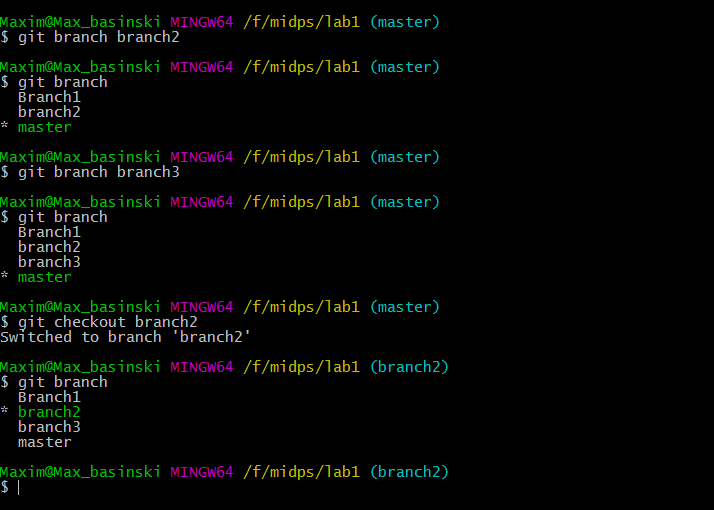
S-a verificat cu ajutorul comenzii *git status* dacă s-au efectuat ceva schimbări în repozitoriu, după care s-au adăugat aceste comenzi în *staging area* prin comanda *git add*.

S-a făcut commit la schimbări prin comanda *git commit –m ”First commit branch1”*.

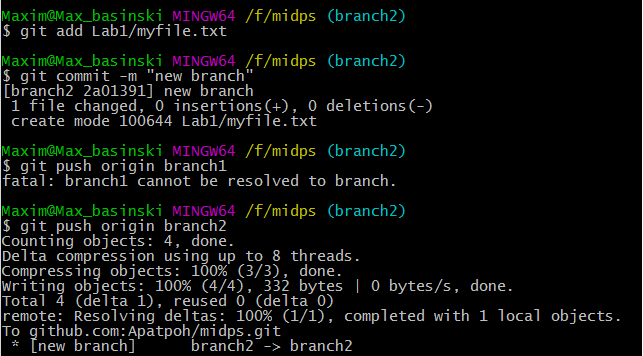
S-au introdus schimbările în repozitoriul în serverul remote cu ajutorul comenzii *git push origin branch1*, unde *origin* este numele implicit al repozitoriului remote de unde s-a făcut clonarea inițială.

**Imagini**

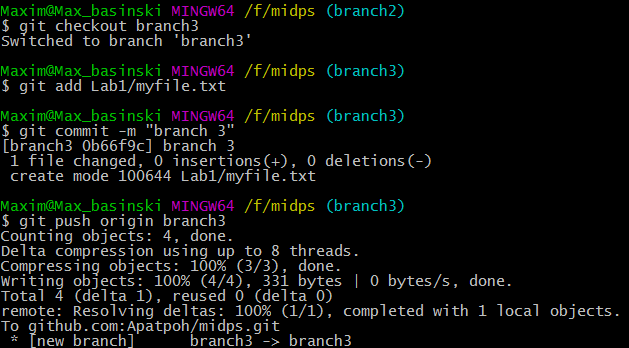
1. Crearea branch-lor noi



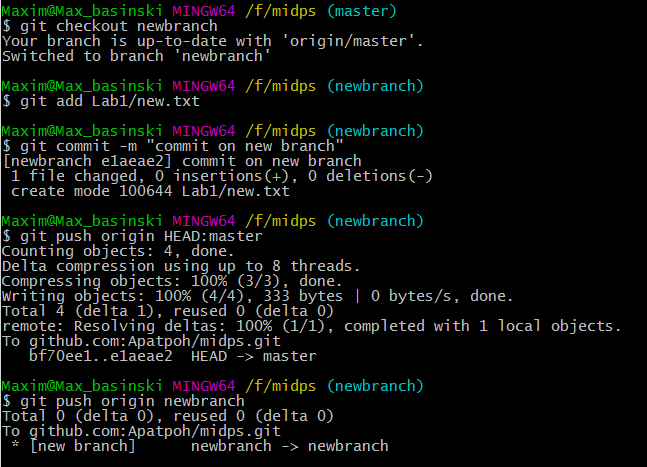
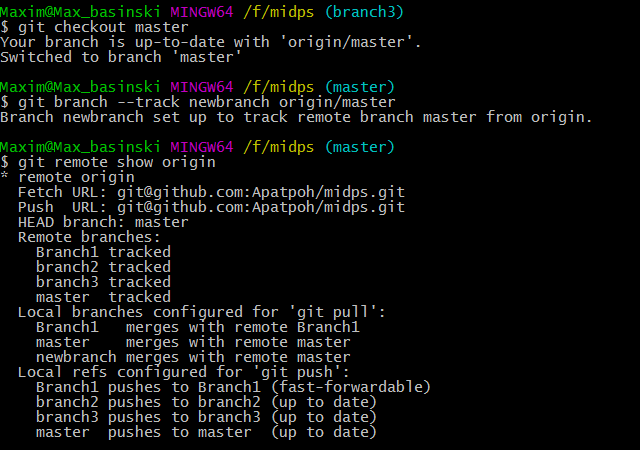
1. Commit in branch2



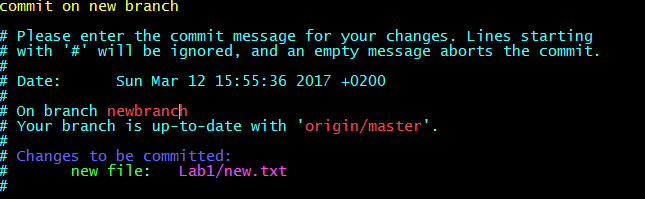
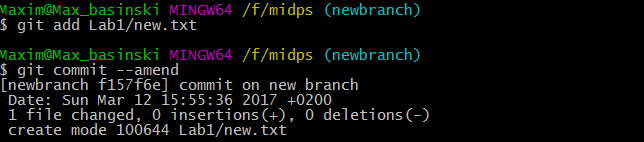
1. Commit in branch3



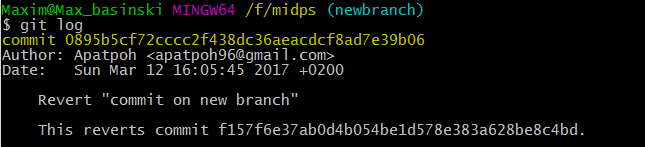
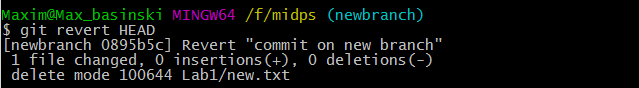
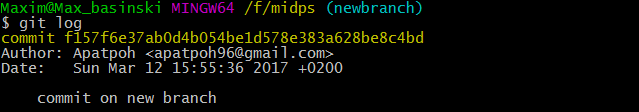
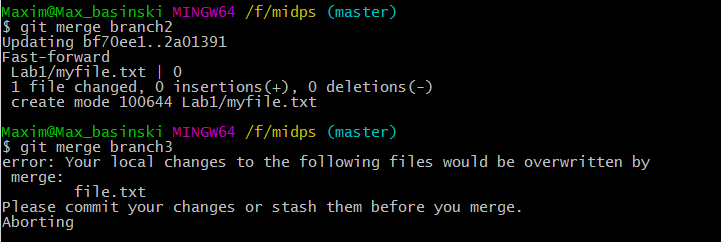
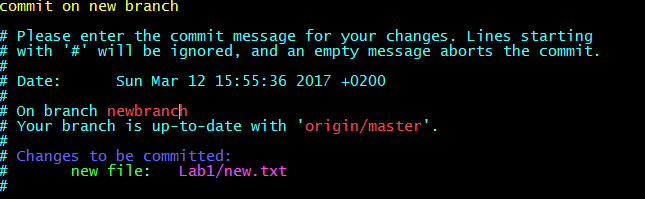
1. Track remote origin



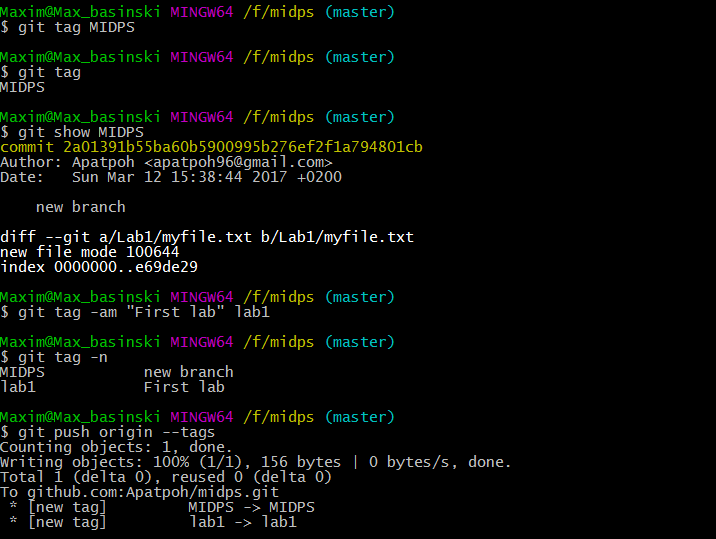
1. Redenumirea unui commit



1. Merge 2 branches



1. Tags



**Concluzie**

Un sistem de control al versiunilor reprezinta un produs software ce ajuta dezvoltatorii dintr-o echipa software sa colaboreze in cadrul diferitelor proiecte si de asemenea pastreaza un jurnal complet al muncii ﬁecaruia. Un sistem de control al versiunilor ( SCV ) are trei scopuri principale : 1. Sa ofere posibilitatea muncii simultane, nu seriale 2. Atunci cand mai multe persoane lucreaza in acelasi timp se asigura ca modiﬁcarile realizate de acestia nu intra in conﬂict unele cu altele 3. Sa ofere o arhiva a ﬁecarei versiuni continand informatii despre cine, unde si din ce motiv a fost facuta ﬁecare modiﬁcare. Sunt 2 forme de organizarea a sistemelor de control al versiunilor: centralizata si distribuita. Principiul de baza al sistemelor centralizate se bazeaza pe relatia clientserver. Un depozit (repository) este situat intr-un singur loc iar mai multi clienti au acces la el. Toate modiﬁcarile utilizatorilor si toate informatiile legate de aceste modiﬁcari (utilizator, data, revizie) sunt transmise si preluate de la un depozit (repository) central. Sistemele de control al versiunilor distribuite sunt o optiune mai noua. In cadrul acestora ﬁecare utilizator are propria copie a intregului repository, nu doar ﬁsiere, ci intregul jurnal. Aceasta abordare foloseste modelul peer-to-peer spre deosebire de modelul client-server folosit de sistemele centralizate. In acest caz sincronizarea dintre repository-uri este realizata prin schimbul de changeseturi sau patch-uri dintre statii. Doua dintre cele mai folosite SCV-uri de acest fel sunt Git si Mercurial. Principalele beneﬁcii ale sistemelor Git sunt : - O urmarire a schimbarilor mai avansata si mai detaliata, lucru care duce la mai putine conﬂicte - Lipsa necesitatii unui server-toate operatiile cu exceptia schimbului de informatii intre repository-uri se realizeaza local - Operatiile de branching si merging sunt mai sigure, si prin urmare folosite mai des - Rapiditate mai mare a operatiilor datorita lipsei necesitatii comunicarii cu serverul Principalele dezavantaje ale sistemelor Git : - Modelul distribuit este mai greu de inteles - Nu exista asa de multi clienti GUI datorita faptului ca aceste sisteme sunt mai noi - Reviziile nu sunt numere incrementale, lucru ce le face mai greu de referentiat - Riscul aparitiei de greseli este mare daca modelul nu este familiar