Lucrare de laborator nr 1

Tema lucrării:

Pregătirea inițială a mediului Git. Crearea unui depozit Git și local

repertoriu.

Partea teoretica:

Git este un sistem de control al versiunilor distribuit gratuit, open-source cod sursă, conceput pentru a lucra rapid și eficient cu orice proiecte - de la mici la foarte mari. Git este ușor de învățat și ocupă puțin spațiu și are performanțe fulgerătoare. Este superior unor astfel de SCM-uri (Supply Managementul lanțului – trad. Supply Chain Management - instrumente precum Subversion, CVS, Perforce și ClearCase, cu caracteristici precum local low-cost ramificații, zone de depozitare convenabile și fluxuri de lucru multiple.

Ramificare și contopire

Caracteristica Git care îl face cu adevărat să iasă în evidență de aproape toți ceilalți alte SCM-uri, este modelul de ramificare. Git permite și încurajează creația mai multe ramuri locale, care pot fi complet independente unele de altele prieten. Crearea, îmbinarea și ștergerea acestor ramuri durează doar câteva secunde.

Aceasta înseamnă că puteți face lucruri precum:

• Comutare de context fără contact. Creați o ramură pe care să o încercați orice idee, fă câteva comite, întoarce-te la locul de unde ramificație a fost făcută, aplicați remedierea, întoarceți-vă unde au fost efectuate experimente și combinate.

• Linii de cod pentru jocuri de rol. Ai o ramură care conține întotdeauna doar ce trimis la producție, altul, în care munca este fuzionată pentru testare și câteva ramuri mici pentru munca de zi cu zi.

• Flux de lucru bazat pe funcții. Creați filiale noi pentru fiecare funcție nouă la care lucrați, astfel încât să puteți face fără probleme comuta între ele și apoi șterge fiecare ramură când funcția vor fi îmbinate cu linia principală.

• Experimente unice. Creați un fir pentru experimentare, înțelegeți asta nu funcționează și șterge-l, părăsește-ți jobul și nimeni nu o va mai vedea ea (chiar dacă în acest timp ai promovat alte ramuri).

Distribuire

Una dintre cele mai utile caracteristici ale oricărui SCM distribuit este inclusiv Git, este distribuția sa. Aceasta înseamnă că în loc de efectuați o „verificare” a vârfului codului sursă curent, efectuați o „clonă” întregul depozit.

Copii de rezervă multiple

Aceasta înseamnă că, chiar dacă utilizați un flux de lucru centralizat, fiecare utilizator are în esență o copie de rezervă completă a serverului principal.

Fiecare dintre aceste copii poate fi ridicată pentru a înlocui serverul principal în caz de defecțiune sau deteriorare. În esență, nu există un singur punct de eșec în Git decât dacă există o singură copie a depozitului.

Flux de lucru

Datorită naturii distribuite a lui Git și sistemului excelent de ramificare este posibil să se implementeze cu relativă uşurinţă un număr practic infinit procesele de lucru.

Flux de lucru în stil subversiune (Subversion)

Fluxul de lucru centralizat este foarte comun, mai ales în rândul utilizatorilor, care au trecerea dintr-un sistem centralizat. Git nu te va lăsa să realizați push dacă cineva a făcut un push de la ultima extracție, deci centralizat funcționează modelul în care toți dezvoltatorii realizează push pe același server pur si simplu minunat.

Instalarea Git

Descărcați programul de instalare (https://git-scm.com/downloads), rulați programul de instalare,

selectați opțiunile de care aveți nevoie (sau lăsați totul așa cum este, dacă nu prea mult înțelegeți), așteptați finalizarea procesului de instalare și ați terminat.

Utilizare Git

În folderul de lucru în care se află fișierele sursă ale aplicației/proiectului, prin terminal sau linia de comandă, utilizați comanda git init (documentația oficială - https://git-scm.com/docs/git-init) pentru a inițializa depozitul local, după ce Git va urmări modificările la fișierele sursă și le va remedia modificările - utilizați comanda git commit (documentație oficială - https://git scm.com/docs/git-commit).

În viitor, dacă trebuie să sincronizați modificările în comun pentru toate depozitele - utilizați comanda git push (documentație oficială - https://git-scm.com/docs/git-push)

Sarcina principală a lucrării:

1. Instalați mediul Git pe dispozitivul dvs., pregătiți un folder pentru aplicație/proiect și inițializați depozitul local din acest folder (comanda git init).

2. Implementați o aplicație simplă (din variantele propuse mai jos). Într-o echipă, atribuiți cine ce părțe a aplicației va implementa care va fi această parte și ce funcție (funcții) va îndeplini.

3. Efectuați commit pentru modificări în codul aplicației/proiectului (comanda git commit). Creați un cont pe GitHub (https://github.com/) (dacă nu aveți unul) și trimiteți numele de utilizator la profesor sau linkul de profil. După ce ați primit mai departe instrucțiuni de la profesor, introduceți modificările în ramura dvs. separată depozite de pe GitHub.

Variante de implementare a sarcinii:

1. O aplicație care efectuează operații aritmetice simple – adunare, scădere, înmulțire, împărțire, exponențiere, extracție rădăcină. Fiecare membru al echipei implementează o clasă care face unul sau două operatii aritmetice. În algoritmul principal (clasa) implementați relația dintre cel puțin două clase.
2. O aplicație care extrage anumite informații din text introdus. Fiecare membru al echipei implementează o clasă care recupereaz informații solicitate în mod specific. În algoritmul principal (clasa) implementează o relație între cel puțin două clase. (nota: încercați să folosiți expresii regulate)
3. Aplicație în care se efectuează manipularea textului – ștergere spații, linii noi, text în oglindă, cuvinte separate, schimba cuvintele, literele în cuvinte în locuri. Fiecare membru al echipei implementează o clasă, care efectuează manipulări specifice cu textul. În algoritmul principal (clasă) implementează o relație între cel puțin două clase.
4. O aplicație în care fișierele sunt manipulate – exemple manipulările sunt similare cu cele din versiunea anterioară. Fiecare membru al echipei implementează o clasă care efectuează manipulări specifice cu un fișier. În algoritmul principal (clasa) pentru a implementa relația, cel puțin, între doua clase.
5. O aplicație care calculează proprietăți geometrice, cum ar fi suprafața și perimetrul, raza pentru diferite forme (de exemplu, pătrate, dreptunghiuri, cercuri, triunghiuri). Fiecare membru al echipei implementează o clasă care calculează o proprietate specifică pentru o anumită formă. În cea mai mare parte algoritm (clasă) pentru a implementa relația dintre cel puțin două clase.
6. O aplicație care recomandă rețete pe baza preferințelor utilizatorului și ingredientele disponibile. Fiecare membru al echipei implementează o clasă pentru recomandări pentru rețete dintr-o anumită bucătărie sau categorie. În cea mai mare parte algoritm (clasă) pentru a implementa relația dintre cel puțin două clase.
7. Aplicație pentru conversia diferitelor valute. Fiecare membru al echipei implementează o clasă care se ocupă de conversia într-o anumită monedă. În cea mai mare parte algoritm (clasă) pentru a implementa relația dintre cel puțin două clase.
8. Aplicație care simulează aruncarea zarurilor și ține evidenț rezultate. Fiecare membru al echipei poate implementa o clasă pentru modelare un anumit tip de zaruri (de exemplu: cu șase fețe, cu douăsprezece fețe). În algoritmul principal (clasa) implementați relația, cel putin intre doua clase.
9. O aplicație care simulează alegerea unei fantome (fiecare fantomă este clasă). Pe baza parametrilor introduși de utilizator (parametri – numere sau text) programul va afișa un anumit tip de fantomă. Fiecare membru al echipei trebuie să implementeze o clasă pentru fantomă cu trei parametri de definit. În algoritmul principal (clasa) pentru a implementa relația, cel puțin, între doua clase.

La finalizarea lucrărilor, elaborați un raport care trebuie să conțină - numele dvs. de familie, prenumele, grup, subiect de lucru, opțiunea dvs. pentru implementarea sarcinii, scurtă descriere

implementarea sarcinii, un link către codul sursă de pe GitHub. Salvați raportul în format

PDF sau WORD și trimiteți la ELSE.