Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava 4

Metodika používania verziovacieho nástroja GIT

Michal Ševčík

Študijný program: Informačné systémy
Ročník: 1
Predmet: Manažment informačných a softvérových systémov
Ak. rok: 2013/14

1 Úvod

Cieľom tohto dokumentu je naučit vývojový tím, ktorý rieši projekt Virtuálna FIIT na predmete tímový projekt, verziovať kód. Verziovať kód sa bude prostredníctvom verziovacieho nástroja git a všetky príklady príkazov budú pre OS linux, ktorý používaju všetci členovia nášho tímu – tím 6.

2 Inicializácia GIT

Aby sme mohli samostatný git vôbec používať je potrebné si ho stiahnuť a nainštalovať. Toto učiníme po zadaní príkazu:

```
$sudo apt-get install git-core
```

Po tejto inštalácií stiahneme najnovšiu verziu nášho programu zadaním príkazu

```
$git clone
https://<registracne_meno>@bitbucket.org/dorny/virtfiit.git
```

3 Práca s GIT

Aby sme mohli začať pracovať na vývojí našej aplikácie, je potrebné vytvoriť vetvu:

```
$git checkout -b <názov>
```

Features:

<názov> je tvorený číslom tasku v jire, znakom „_“ a sekcia z produkt

backlogu kde sa nachádza task resp. čoho sa to týka {mojrozvrh, vyhladavanie, mapy, mhd, jedalne, uzitocnewww, usability, grafika} príklad: 61_usability

Bugy:

<názov> je tvorený číslom tasku v jire, znakom „_“ a slovom „bug“. Príklad: 12_bug

Ak úloha, ktorú sme dostali je rozsiahla, tak ju rozdelíme na menšie podúlohy a pre každu podúlohu si spravime branch. Tieto branche budu iba lokálne a nepôjdu na repozitár. Ich názvy budú

obsahovať na konci čísla(.1,.2,.3,...) napr. 61_usability.1

Môže sa začať upravovať kód.

Po úprave kódu si pozrieme zmenené súbory:

```
$git status
```

V časti modified sa zobrazia súbory, ktoré boli upravené a v untracked files su súbory, ktoré sú nové – pribudli. Upravené súbory commitneme príkazom:

```
$git add <súbor/>
```

respektívne interaktívnou metódou:

```
$git add –i
```

Po tom ako ich všetky vložime, commitneme ich.

```
$git commit –m
```

Po zadaní tohto príkazu správa bude obsahovať tieto informácie:

<čo sa upravilo>;

Akcia:<čo treba spraviť aby sa to otestovalo>;

Výsledok:<aký ma byť výsledok>;

Hotovo:<0-10>

Ak sa jedná len o grafickú časť aplikácie, ktorú netreba testovať, do akcie a výsledku vložíme znak „-“.

Pre commit všetkých modified súborov použijeme:

```
$git commit –am
```

Pre odoslanie zmenených súborov na vzdialený repozitár:

$git checkout <hlavná_lokálna_branch> (napr. 16_usability)

$git merge <lokálna_branche> --> toto spravíme pre všetky lokálne branche

$git push origin <hlavna_lokálna_branch>

3.1 Stiahnutie najnovšej verzie počas vývoja

Na stiahnutie novej verzie použijeme:

$git pull --rebase origin master

3.2 Dokončenie vývoja na danej branchy

Po tom ako pushneme našu verziu, spravíme merge na branch sprint<cislo>(cislo je číslo aktuálneho šprintu):

$git checkout sprint<cislo>

$git merge <hlavná_lokálna_branch>

$git branch -d <hlavná_lokálna_branch>

3.3 Riešenie konfliktov pri merge

Ak pri merge vznikne konflikt použijeme príkaz

$git mergetool

ktorý nám zobrazí dva dokumenty v ktorých nastal konflikt. Prejdeme tieto dva dokumenty a vždy na mieste konfliktu vyberieme riadok, ktorý je správny.

Prejdú sa dokumenty až do konca a stlačí sa klávesova skratka „ctrl + m“.