Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
Iľkovičova 2, 842 16, Bratislava 4

Tímový projekt

MOB-UX

Metodika verziovania kódu a mergovania

Vedúci projektu: Ing. Eduard Kuric, PhD.
Názov tímu: MOB-UX
Členovia tímu: Bc. Tomáš Anda
Bc. Dávid Beňo
Bc. Matúš Buzássy
Bc. Martin Nagy
Bc. Patrik Pindéš
Bc. Ľuboš Štefunko
Bc. Igor Vereš
Vypracoval: Bc. Patrik Pindéš
Kontakt: team11fiittp@gmail.com
Akademický rok: 2017/2018, zimný semester

Obsah

1. Účel dokumentu

Dokument slúži ako metodika na verziovanie kódu a všetky s tým súvisiace činnosti, ako sú mergovanie vetiev, vytváranie vetiev, konvencie názvoslovia, práca a pomocné funkcie ponúkané integrovaným verziovacím systémom v PHPStorm.

2. Predpoklady

1. Git
2. PHPStorm

3. Počiatočné kroky

Všetky operácie s VCS (Version Control System) sú dostupné z menu položky VCS. Niektoré sa dajú vyvolať kontextovým oknom (pravé tlačidlo myšky) alebo existujúcimi ikonkami v okne **Version Control**, na hornej **nástrojovej lište** alebo na **dolnej lište okien.**

3.1. Stiahnutie kódu (checkout)

Kód sa dá stiahnuť pomocou IDE PHPStorm z nášho BitBucker repozitára:

1. Spustite PHPStorm alebo zatvorte aktuálne otvorený projekt pre návrat na úvodnú obrazovku
2. Check out from Version Control -> Git
3. Do dialógového okna zadajte url repozitára projektu (každý člen tímu má vlastné)
    a. Napr.: https://xpindesp1@bitbucket.org/uxmobileteam/uxmobile-repo.git
    b. Vypýta to to prihlasovacie heslo pri potvrdení.
4. Môžete si zvoliť cestu a názov priečinku na uloženie (odporúčam čo najkratšiu)
5. Ak máte Git nainštalovaný a pripojenie na internet, úspešne sa vám stiahne projekt

3.2. Nastavenie PHPStorm

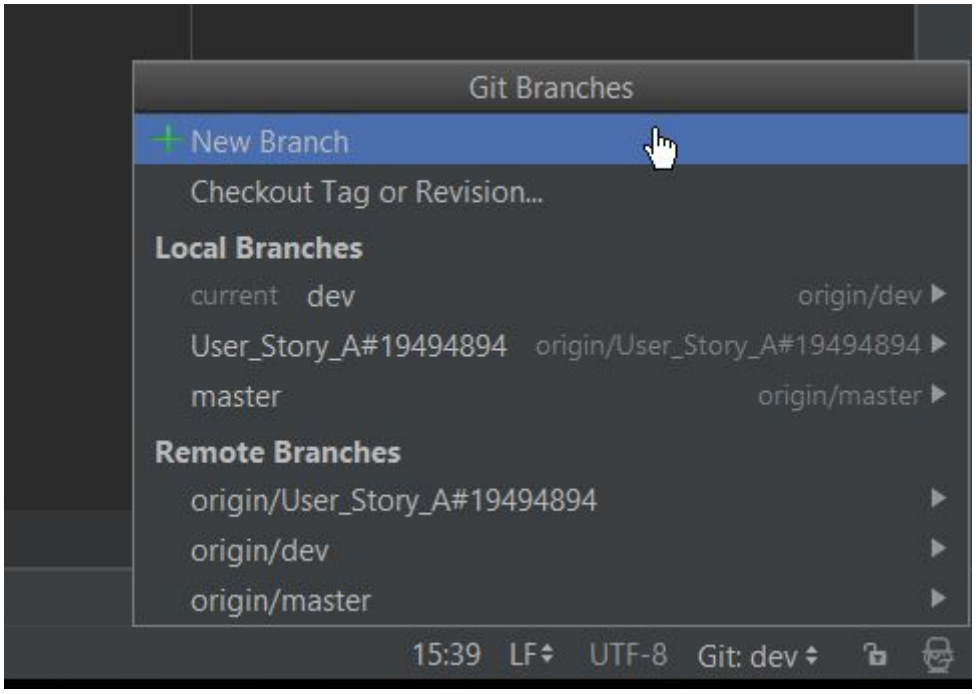
Zapnite si nástrojové okno verziovania View->Tool Windows->**Version Controll**

4. Vytvorenie branch(vetvy)

Každý, kto pracuje na nejakej **User Story** si vytvorí na to vlastný vedľajší **branch** z najaktuálnejšej verzie **dev branch**. Na svojom branch bude vyvíjať svoju funkcionalitu.

Každý branch môže byť **local** (lokálne na počítači) a **remote** (v BitBucket repozitári)**.** Vedľajší branch vytvoríte najprv lokálne, následne sa **push**-ne aj na remote.

1. Pre vytvorením branche si aktualizujte **dev** cez **Update Project.**
    a. musíte mať aktuálne checkout-nutý **dev** branch
2. Vytvorte si vedľajší branch (New Branch), pomenujte ho Nazov_User_Story#Cislo_user_story_zo_Scrumdesk



5. Updatovanie projektu

Existuje viacero spôsobov ako udržiavať svoje lokálne vetvy aktuálne, byť zosynchronizovaný s remote vetvami: **Fetch, Update Projet, Pull.**

Podrobný popis vo webStorme je tu :

https://www.jetbrains.com/help/phpstorm/2017.2/using-git-integration.html?utm_campaign=PS&utm_content=2017.2&utm_medium=help_link&utm_source=from_product#sync-with-remote-repository

5.1 Pull

V prípade, že chcete mať zmeny z hocijakej vetvy, je dobré použiť metódu Pull, pri ktorej sa zadefinuje, z ktorej vervy sa majú potiahnuť zmeny. Vetva sa vyberá v dialógovom okne **Pull Changes** v kolonke **Branches to merge**(nie, nie je to preklep).

6. Commit & Push

Pre commit kódu postupujte nasledovne:

1. Over, či si na správnom branch
2. Skompiluj a spusti kód, zisti či funguje
3. Klikni na **Commit Changes**
4. Presvedč sa, či sú v strome dokumentov zvolené len tie lokálne zmenené súbory, ktoré chceš commit-núť
    a. V sekcii **Diff** môžeš vidieť náhľad na rozdiely vo verziách zvoleného súboru
5. Napíš krátku správu o aplikovanej zmene do **Commit Message**
    a. Klikni na **Commit and Push**
6. Ak sa zobrazí **warning** správa, neignoruj ju! Oprav varovania a chyby a opakuj postup od bodu 2.

7. Merge-ovanie vetiev

Dev branch predstavuje branch, na ktorej sa bude pracovať v rámci šprintu. To znamená, že všetky vyriešené User Story branch sa mergnú do dev a na konci šprintu, ak je všetko v poriadku, tak Scrum Master mergne dev do master branch.

7.1. Mergovanie vetvy User Story do dev

1. Uisti sa, že sa nachádzaš na svojej User Story vetve
2. Uisti sa, že všetky chcené zmeny sú push-nuté, nechcené revert-nuté alebo stash-nuté
3. Skompiluj a spusti kód. Over či funguje.
4. Checkout dev.
5. Aktualizuj dev cez **Update Project**
6. Merge svoj branch do dev
    a. Vyrieš potenciálne konflikty
7. Skompiluj a spusti kód. Over či funguje.
8. Vykonaj **Push.**

7.2. Mergovanie vetvy dev do master

1. Uisti sa, že sa nachádzaš na dev
2. Uisti sa, že všetky chcené zmeny sú push-nuté, nechcené revert-nuté alebo stash-nuté
3. Skompiluj a spusti kód. Over či funguje.
4. Checkout master.
5. Aktualizuj master cez **Update Project**
6. Merge dev do master
    a. Vyrieš potenciálne konflikty
7. Skompiluj a spusti kód. Over či funguje.
8. Vykonaj **Push.**

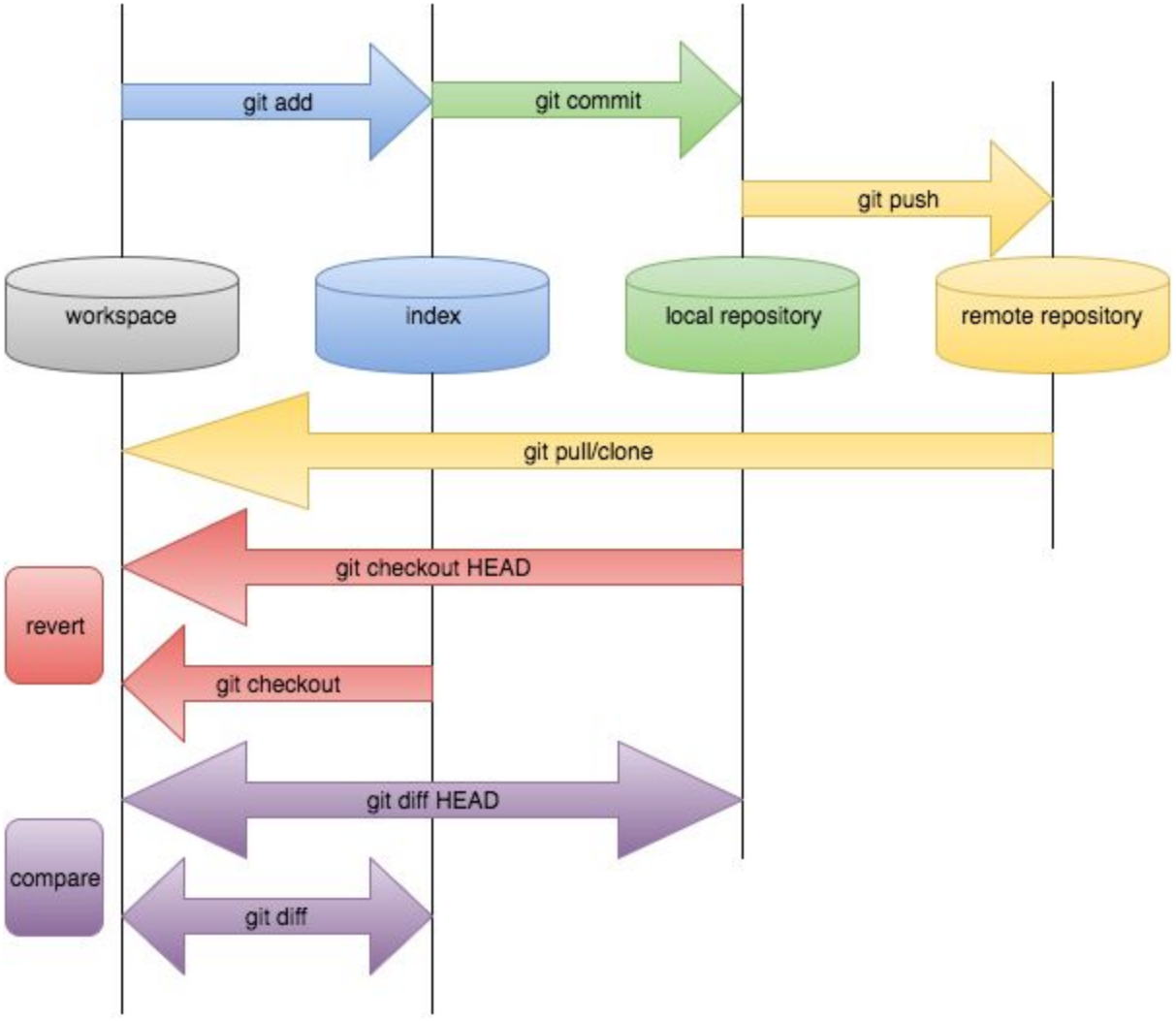
8. Tipy

1. Pri zmene branch lokálne zmeny ostávajú.
2. Ak si chcete z nejakého dôvodu odložiť lokálne zmeny, dá sa to pomocou VCS->Git->**Stash Changes**
    a. Lokálne zmeny sa dajú obnoviť VCS->Git->**UnStash Changes**
    b. Záložku lokálnych zmien (**Local Changes**) treba občas obnoviť manuálne **Refresh**
3. Všetky branch-e (ktoré boli aspoň raz checkout-nuté z remote) sa dajú aktualizovať pomocou VCS->Git->**Fetch**
4. Lokálne ne-Commit-nuté zmeny sa dajú navrátiť pomocou **Revert**
    a. Pri Revert sa volia konkrétne súbory
5. Git umožňuje porovnávanie zmien v súboroch

9. Dodatočné Informácie

http://rogerdudler.github.io/git-guide/

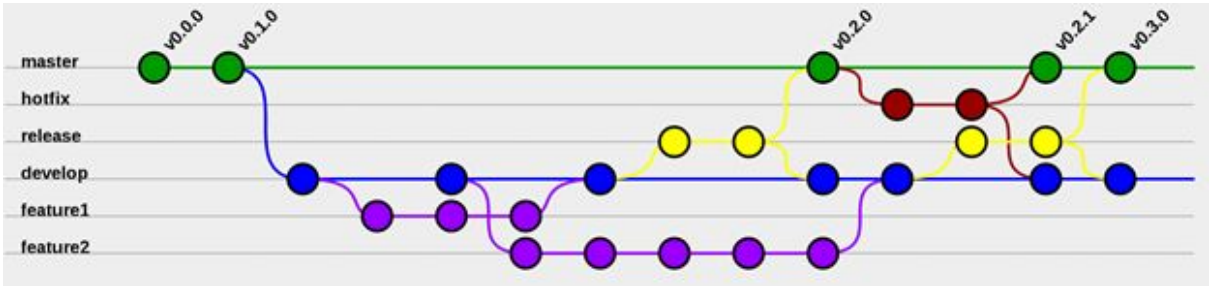
8.1. Git Workflow



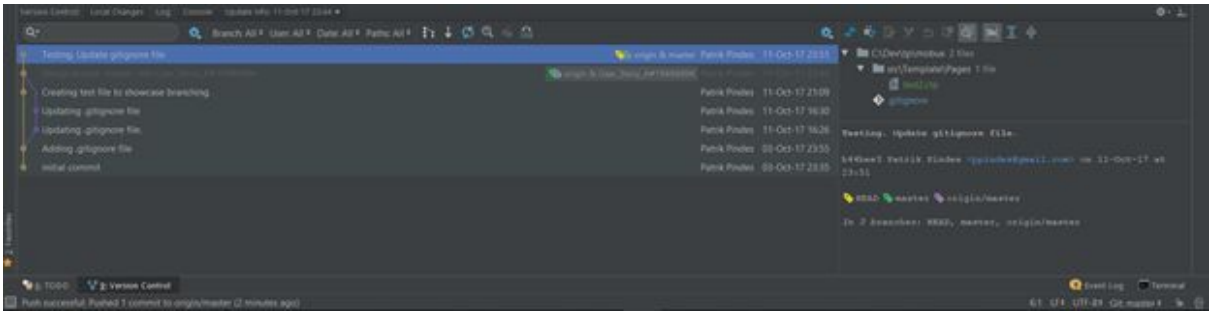
https://wuxiaomin98.files.wordpress.com/2016/04/gitflow.png

8.2. Git branch log history

V okne **Version Control** na záložke **Log** je možné sledovať históriu akcií. Na obrázku je názorná ukážka mnohých vetiev. Kruhy predstavujú akcie commit&push, v prípade, že sa nachádzajú na rozmedzí 2 rôznych farieb, tak predstavujú merge.

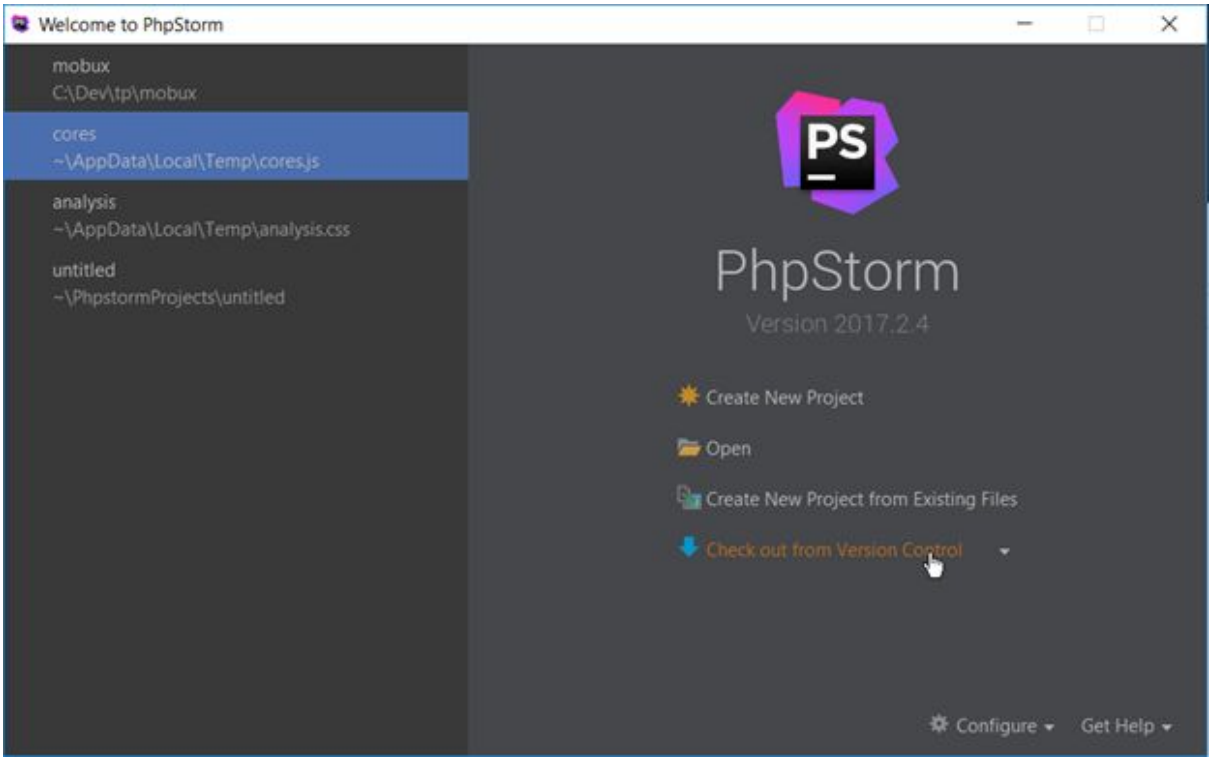


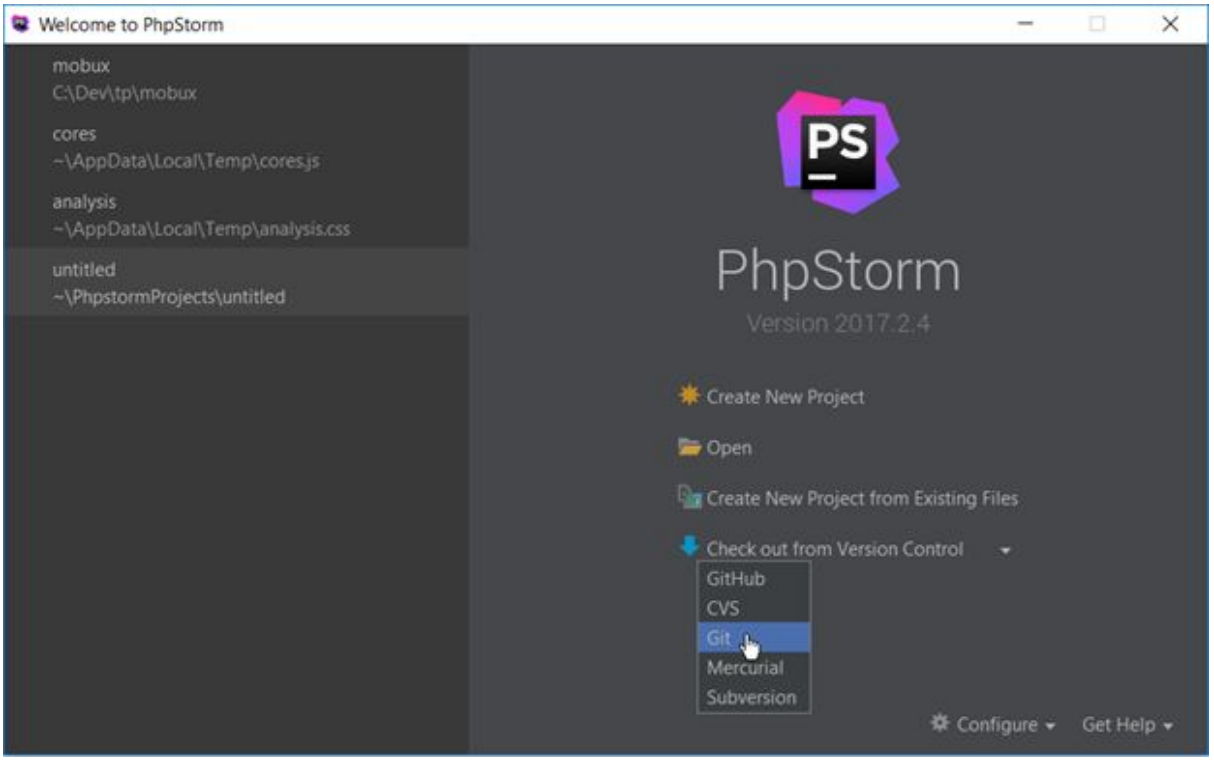
https://nurelm.com/wp-content/uploads/2015/05/git-flow.png

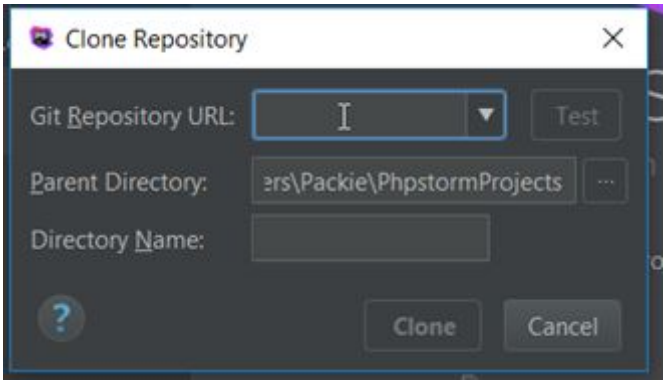


9. Obrázky akcií v PHPStorm

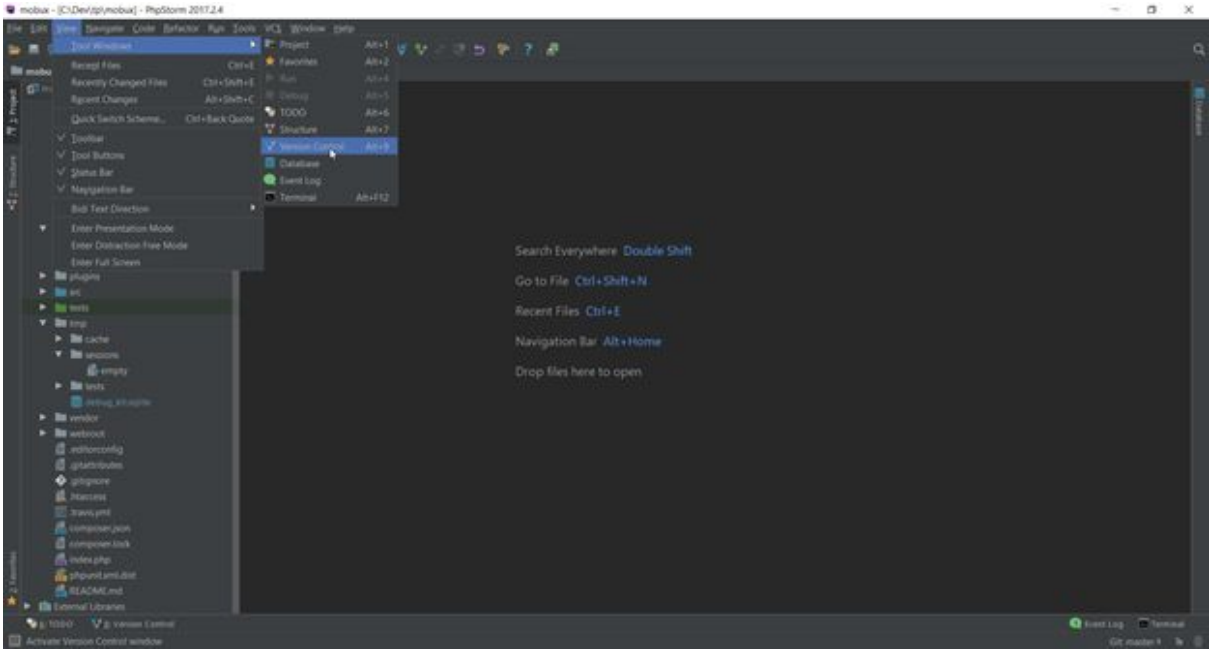
9.1. Projekt checkout z BitBucket

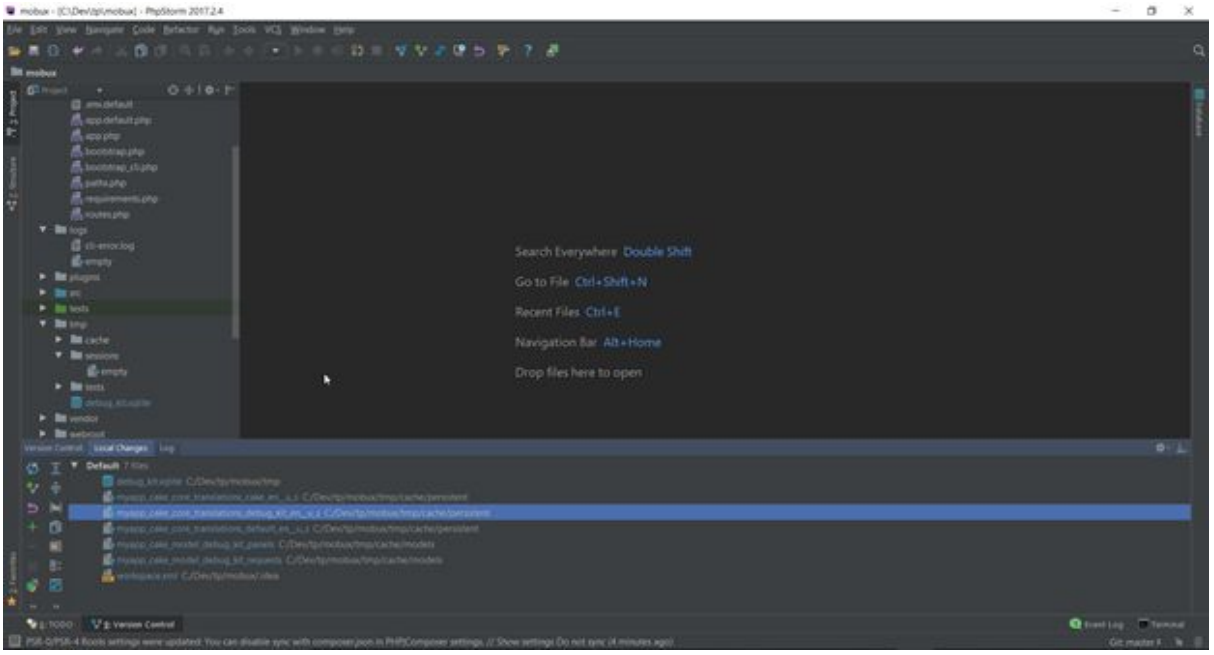



Welcome to PhpStorm
mobux
C:\Dev\tp\mobux
cores
~\AppData\Local\Temp\cores.js
analysis
~\AppData\Local\Temp\analysis.css
untitled
~\PhpstormProjects\untitled
PhpStorm
Version 2017.2.4
Create New Project
Open
Create New Project from Existing Files
Check out from Version Control
GitHub
CVS
Git
Mercurial
Subversion
Configure
Get Help

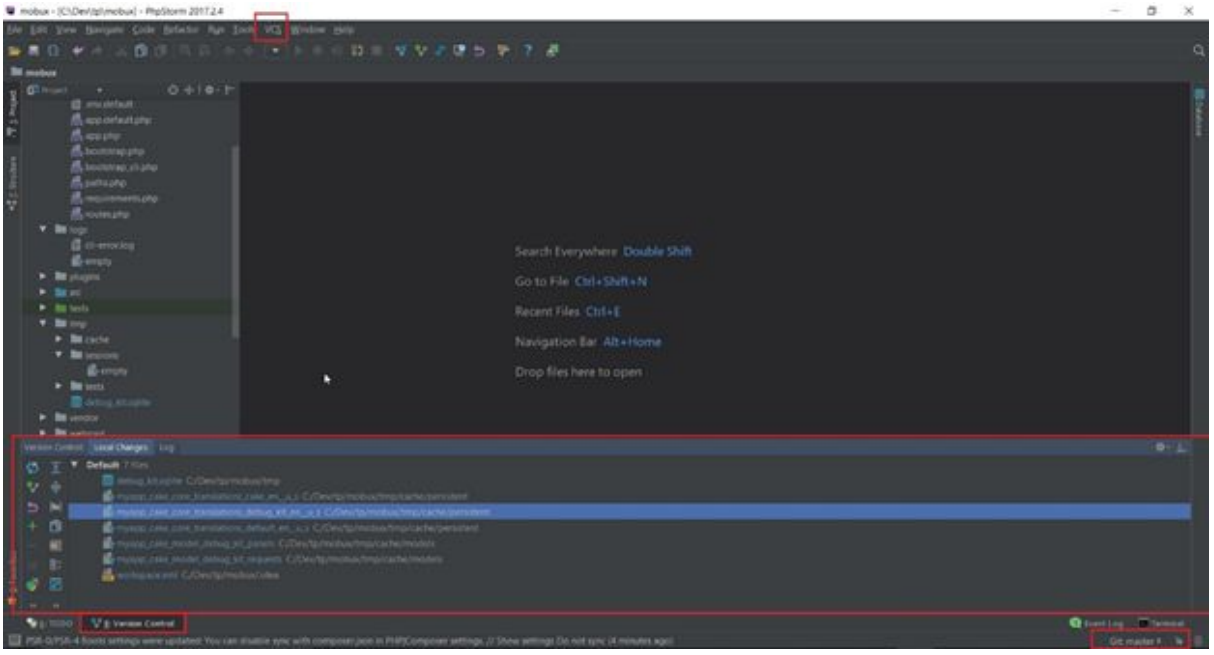

Clone Repository
Git Repository URL:
Test
Parent Directory:
ers\Packie\PhpstormProjects
Directory Name:
Clone
Cancel

9.2. Zapnutie “Version Control” okna

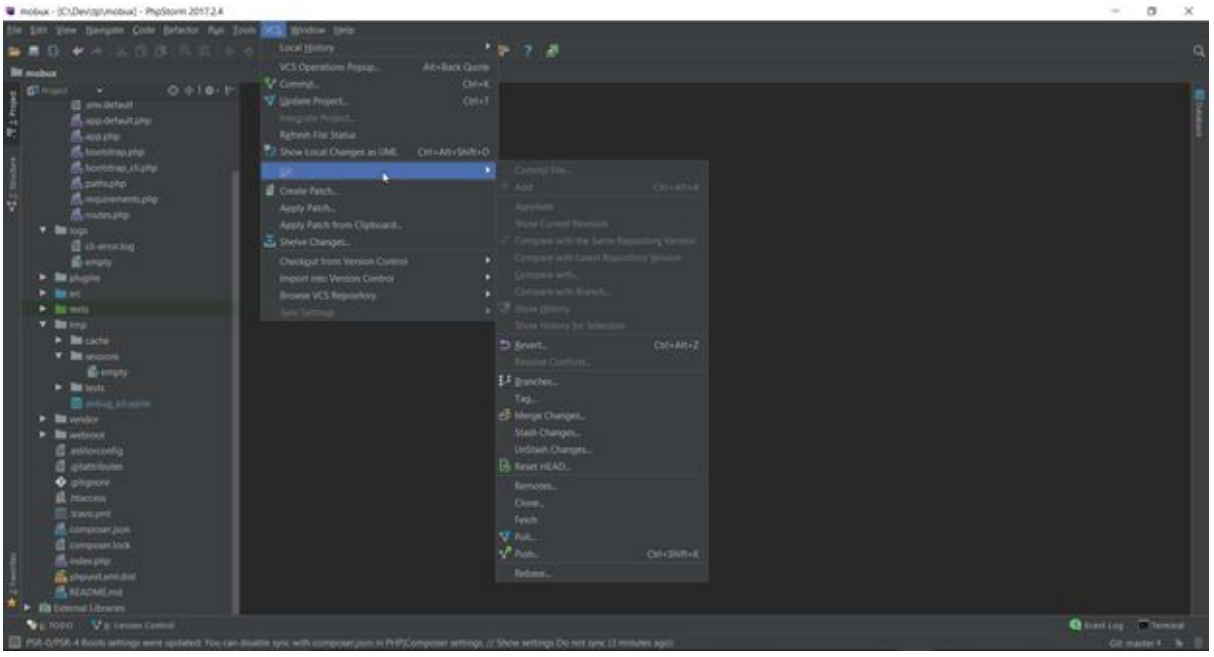




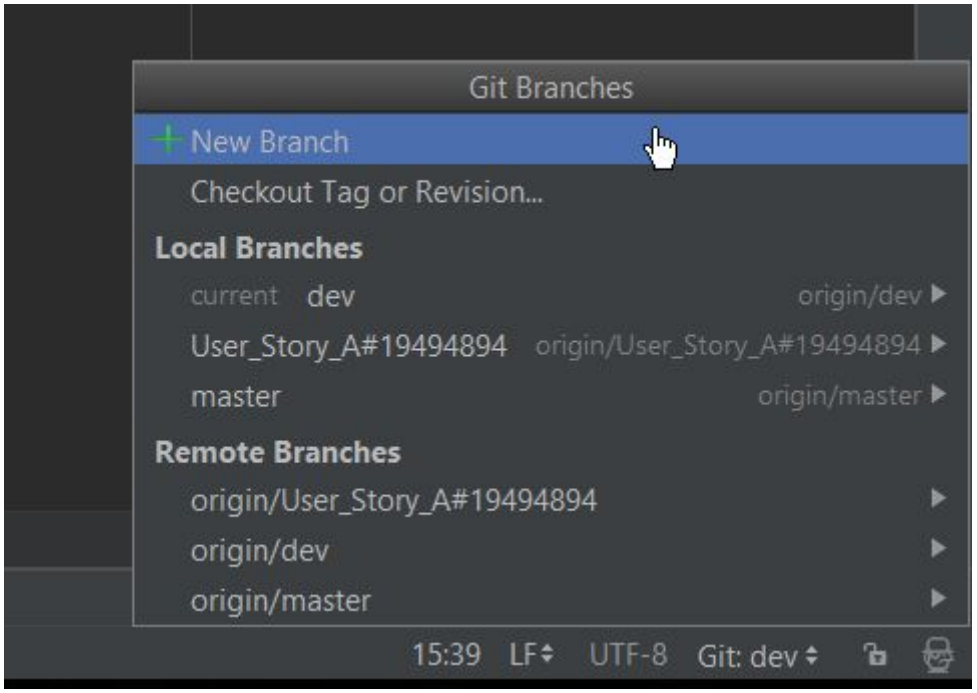
9.3. VCS elementy

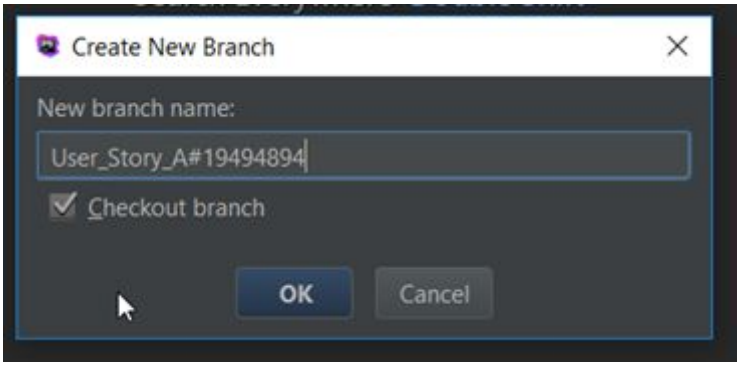


9.4. VCS menu

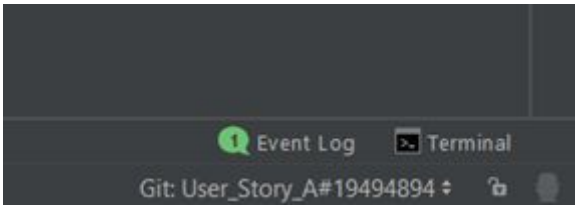


9.5. Vytvorenie vedľajšej vetvy (branch)

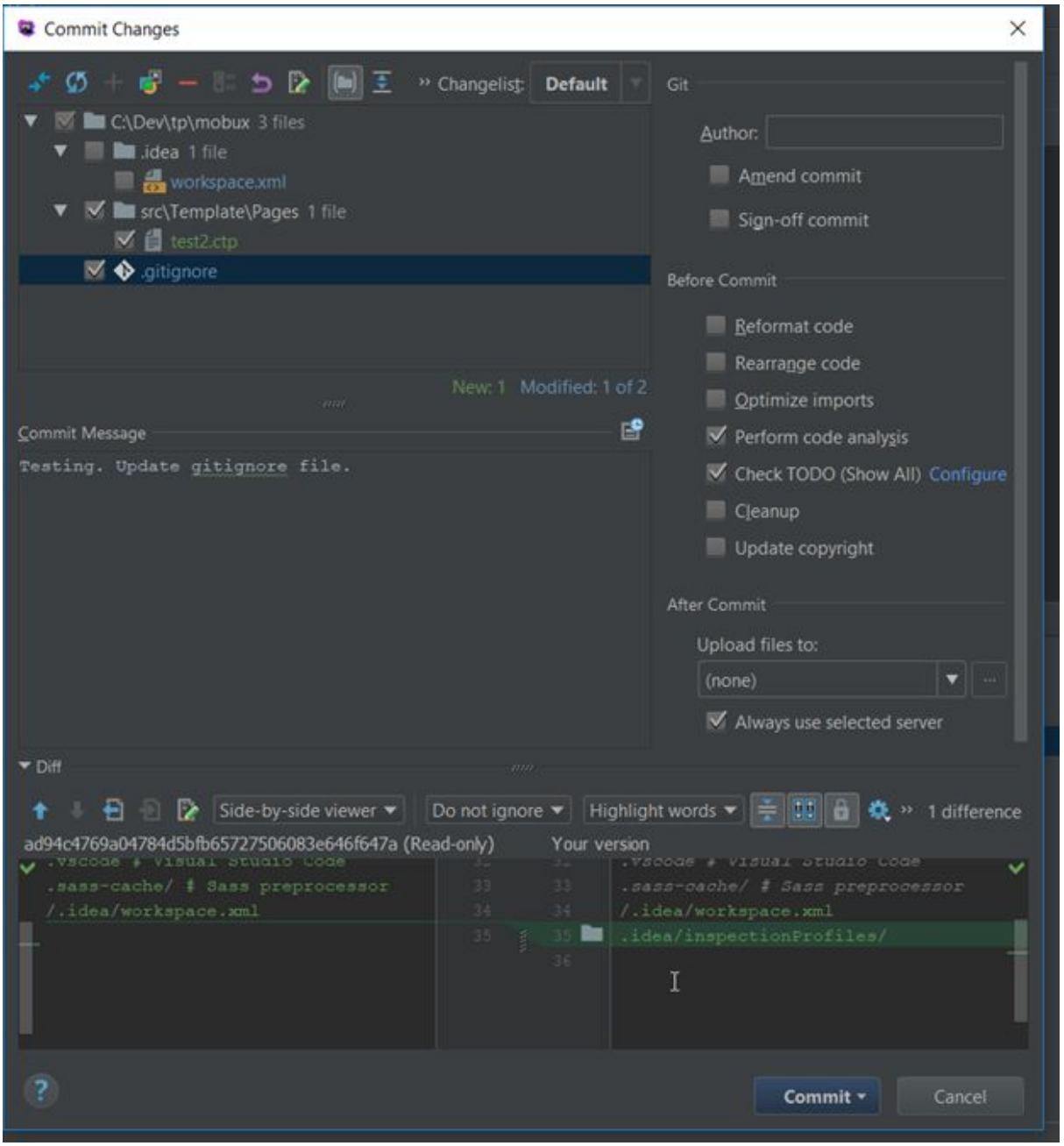




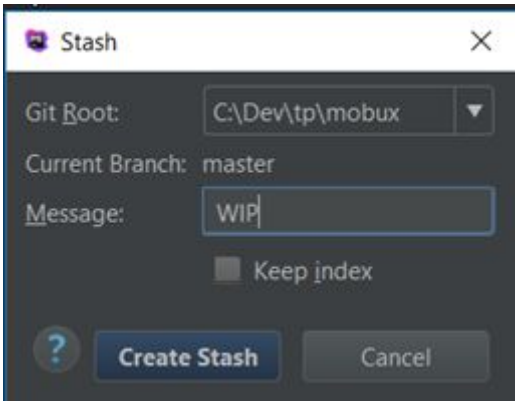
9.6. Ukazovateľ aktívnej vetvy

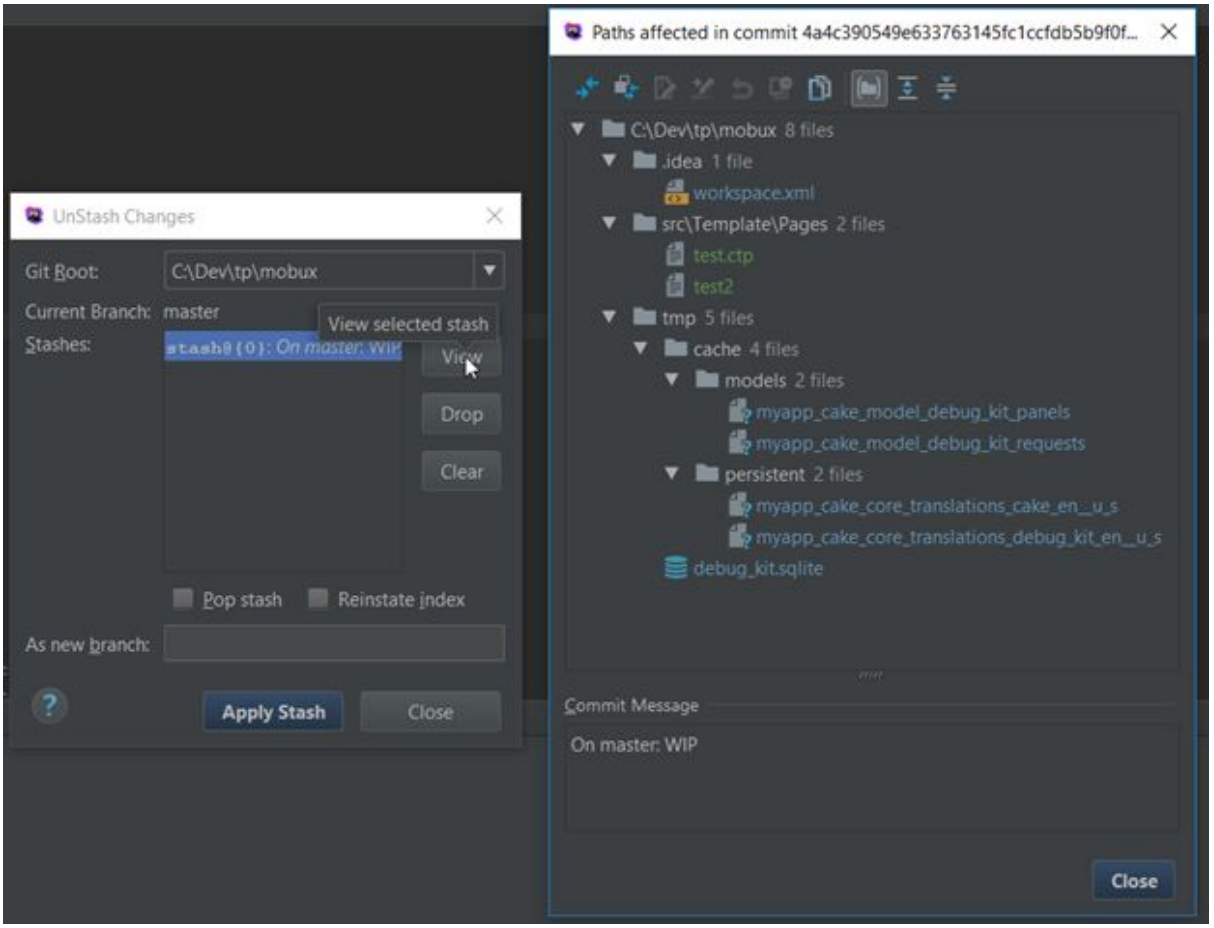


9.7. Commit & push

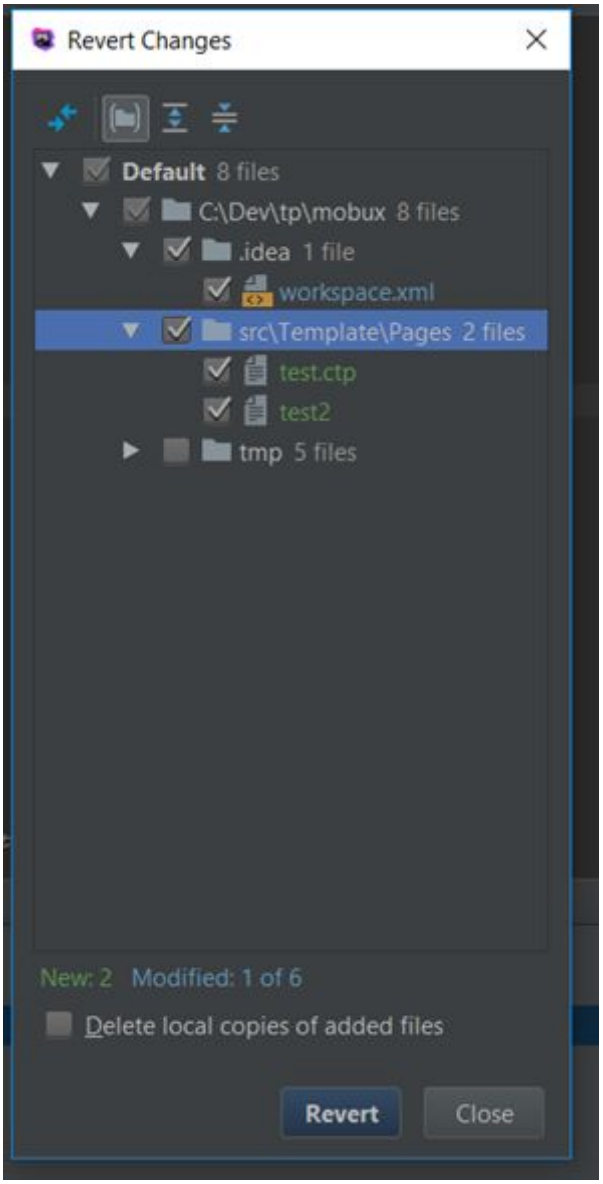


9.8. Stash & UnStash

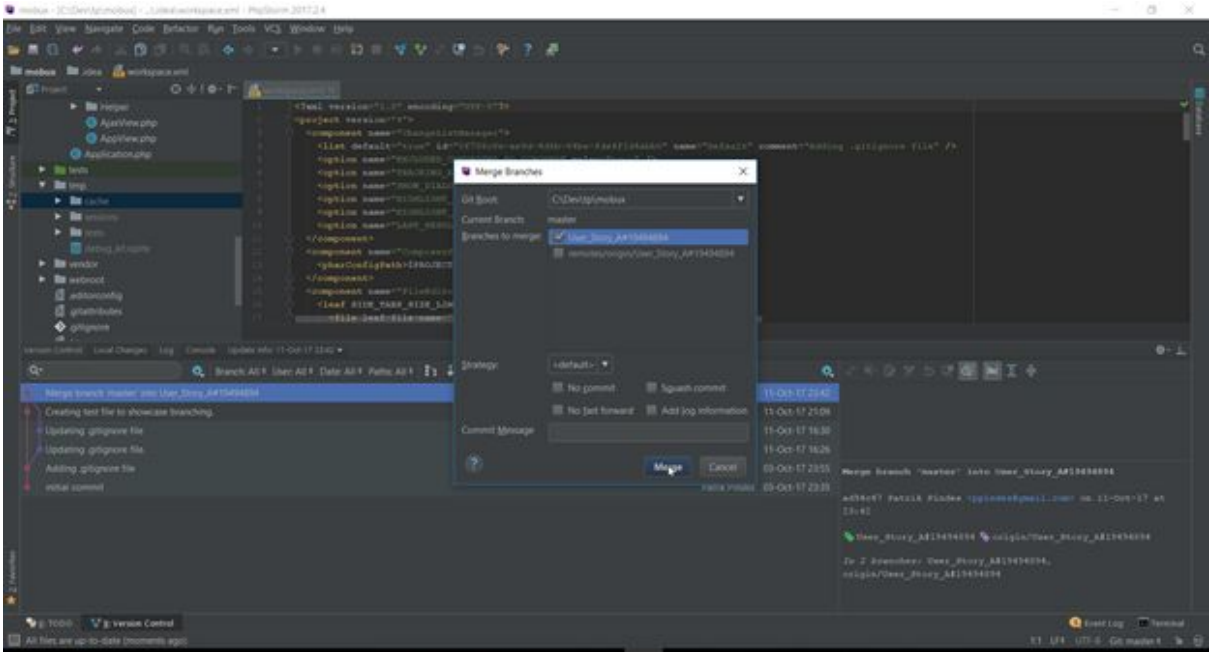


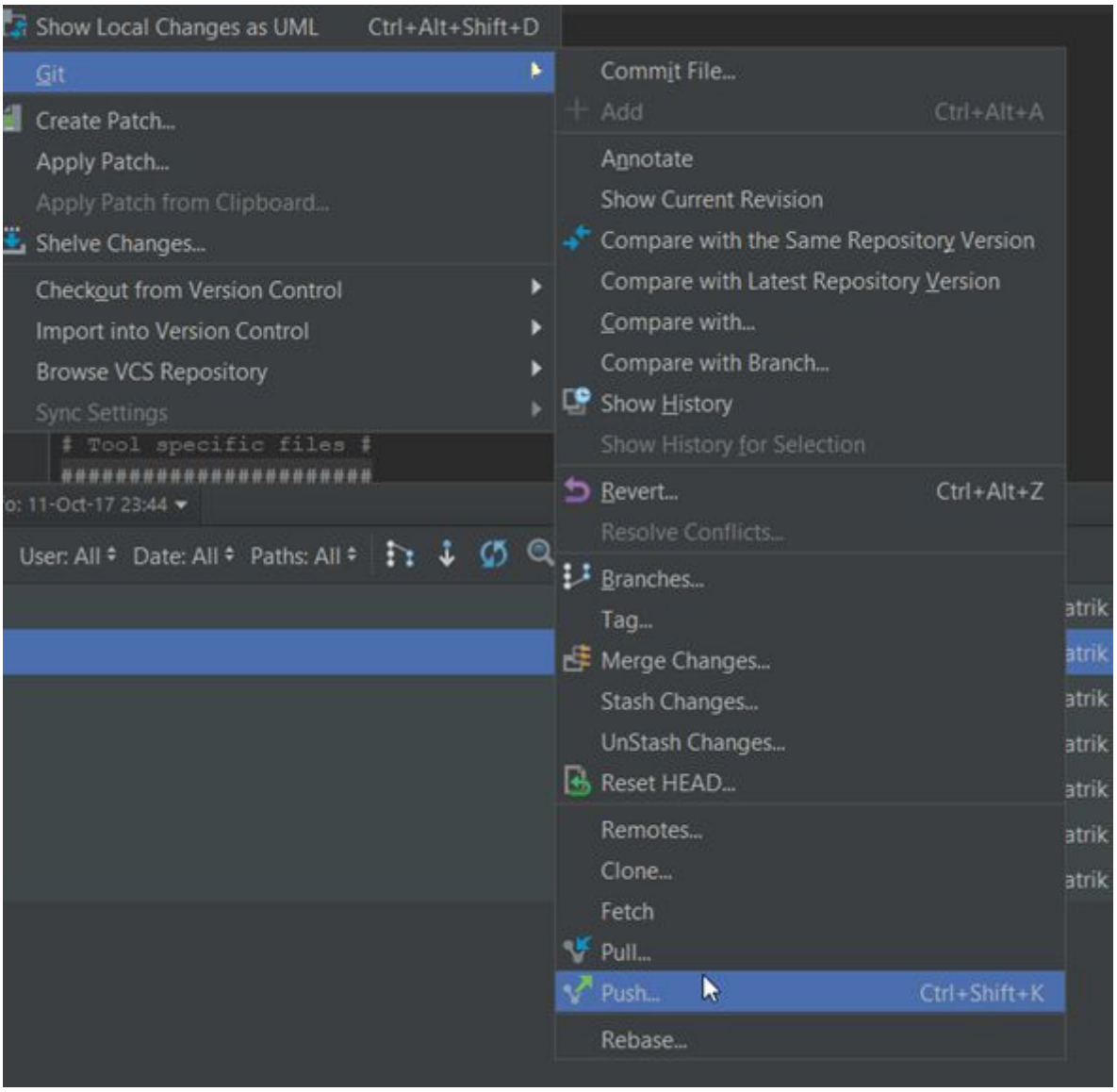


9.9. Revert


Revert Changes
Default 8 files
C:\Dev\tp\mobux 8 files
.idea 1 file
workspace.xml
src\Template\Pages 2 files
test.ctp
test2
tmp 5 files
New: 2 Modified: 1 of 6
Delete local copies of added files
Revert
Close

9.10. Merge


Merge Branches
Git Root:
Current Branch:
master
Branches to merge:
Strategy:
<default>
No commit
Squash commit
No fast forward
Add log information
Commit Message:
Merge
Cancel


Show Local Changes as UML Ctrl+Alt+Shift+D
Git
Create Patch...
Apply Patch...
Apply Patch from Clipboard...
Shelve Changes...
Checkout from Version Control
Import into Version Control
Browse VCS Repository
Sync Settings
Tool specific files
o: 11-Oct-17 23:44
User: All
Date: All
Paths: All
Commit File...
Add Ctrl+Alt+A
Annotate
Show Current Revision
Compare with the Same Repository Version
Compare with Latest Repository Version
Compare with...
Compare with Branch...
Show History
Show History for Selection
Revert... Ctrl+Alt+Z
Resolve Conflicts...
Branches...
Tag...
Merge Changes...
Stash Changes...
UnStash Changes...
Reset HEAD...
Remotes...
Clone...
Fetch
Pull...
Push... Ctrl+Shift+K
Rebase...