Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológií

Ilkovičova 2, 842 16 Bratislava 4

Metodika dokumentácie

Tím č.20 – RavensTeam

Účel metodiky

Pri práci na rozsiahlejšom projekte treba počítať s väčším počtom ľudí, ktorí do neho budú môcť určitým spôsobom zasahovať. Pre vývoj softvéru to znamená, že každý programátor má prístup k zdrojovému kódu a môže ho upravovať. Ak by každý progrmátor pracoval na tej istej verzii zdrojového kódu, hrozilo by, že svojimi zmenami naruší funkcionalitu implementovanú iným vývojárom a tým napr. znemožní prácu celému tímu. Z tohto dôvodu treba zabezpečiť, aby mal každý developer v reálnom čase svoju verziu kódu, vďaka čomu budú môcť pracovať a vykonávať zmeny nezávisle. Táto metodika pojednáva o použitých systémoch, princípoch a pravidlách verziovania pri vývoji nášho softvéru.

Verziovací systém - Git

Git je voľne dostupný, distribuovaný systém pre správu verzií nie len zdrojových kódov, ale rôznych typov dokumentov a súborov, plne kompatibilný s TFS.

Link: https://git-scm.com/

Slovník pojmov

Branch – vetva projektu, reprezentuje jeho určitú verziu

Merge – spojenie dvoch vetiev do jednej

Push – príkaz pre presunutie zmien z jednej vetvy do inej (remote)

Pull – príkaz pre získanie vetvy a jej obsahu

Checkout – príkaz na presúvanie sa medzi vetvami

Princíp – GitFlow Workflow

Tiež nazývaný „Branching model“, je súbor odporúčaných pravidiel verziovania pri vývoji softvéru. Jeho základom sú dve vetvy, podľa konvencií zvyčajne s názvom „master“ a „develop“. Vetva master reprezentuje hlavný zdrojový kód softvéru a slúži na uchovávanie údajov o „releas-och“ resp. o zmenách vykonávaných v hlavnom zdrojovom kóde. Develop je integračná vetva, prostredníctvom ktorej sa vykonávajú zmeny v systéme. Z nej si každý vývojár pri implementácií novej funkcionality vytvorí novú vetvu, na ktorej pracuje a tú potom merge-ne naspäť do develop-u. Odtiaľto sú neskôr zmeny push-nuté do vetvy master.

Postup a základné príkazy

Pre prácu s Git-om existuje niekoľko voľne dostupných nástrojov, ktoré poskytujú buď grafické používateľské rozhranie, alebo základné konzolové rozhranie. Vo všeobecnosti je však ich fungovanie založené na rovnakom princípe.

Každý developer zodpovedný za svoju user-story si vytvorí novú vetvu z vetvy dev. Na tejto vetve budú vykonávané zmeny spojené s task-ami pre danú user-story.

1. Presunieme sa do vetvy dev
    - git checkout dev
2. Uistíme sa, že pracujeme na aktuálnej verzii branch-e dev
    - git pull
3. Developer zodpovedný za user-story vytvorí novú vetvu pre túto user-story
    - git branch <BranchNameByUserStory>
4. Presunieme sa do danej vetvy
    - git checkout <BranchNameByUserStory>
5. Po dokončení práce na task-u sa požadované zmeny pridajú na commit-nutie
    - git add <FilesToBeCommited>
6. Commit-nutie zmien do vetvy
    - git commit –m “<CommitMessage>”
7. Presunieme sa do vetvy dev
    - git checkout dev
8. Uistíme sa, že máme aktuálnu verziu
    - git pull origin dev
9. Merge zmien z user-story vetvy do dev
    - git merge <BranchNameByUserStory>

Pravidlá

Reprezentujú pravidlá pre pomenovávanie jednotlivých vetiev a commit správ kvôli prehľadnosti a konzistentnosti práce počas vývoja softvéru. Zároveň vďaka nim dokážeme ľahšie odhaliť prípadné zlyhania systému a tým pádom ich aj rýchlejšie riešiť.

Vytváranie novej branche

- Pre novú **UserStory**:
    - **US-<UserStoryNumber>/<Descriptive_Text>**
    - **US-4507/Spojenie_s_databazou**

Commit message

- Pre Task:
    - **[<ProjectName>] Task <TaskNumber>: <Descriptive Text>**
    - **[VsExtension] Task 5005: Nacitanie pouzivatelov z databazy**
- Pre Fix ku konkrétnemu task-u:
    - **[<ProjectName>] Fix <TaskNumber>: <Descriptive Text>**
    - **[VsExtension] Fix 5005: Oprava SQL querry**
- Pre Fix generálne:
    - **[<ProjectName>] Fix: <Descriptive Text>**
    - **[VsExtension] Fix: Zmena textu v menu**