POPRO – Laboratorium 2
Praca zespołowa. Wersjonowanie. Sprawozdania.

Łukasz Dąbała

1 Tematyka

Laboratorium drugie dotyczy pewnych aspektów pracy zespołowej, wersjonowania oprogramowania oraz sposobu przygotowania sprawozdań z niektórych zajęć.

2 Przygotowanie do zajęć

To ćwiczenie laboratoryjne wymaga zapoznania się z materiałami pomocniczymi. Gdybyś chciał zapoznać się z nimi na zajęciach – nie starczy Ci czasu na wykonanie zadań, które muszą być zrobione, a ich rezultaty przekazane prowadzącemu *przed* końcem zajęć. Zawsze przeczytaj całą instrukcję i oszacuj ilość czasu potrzebną do zrobienia wszystkich zadań. Przygotuj się do zajęć co najmniej oglądając dostępny materiał umieszczony w Internecie.

Przed zajęciami należy zapoznać się z literaturą dotyczącą narzędzi wersjonowania plików, np. CSV, Git, SVN, GitLab. Warto także obejrzeć cztery filmy w dokumentacji Gita (https://git-scm.com/doc) oraz przejrzeć książkę Pro Git (https://git-scm.com/book/en/v2).

3 Przebieg laboratorium

W trakcie laboratorium należy wykonać wszystkie zadania. Jeśli oczekiwany jest rezultat zadania – należy przedstawić go prowadzącemu.

3.1 Wersjonowanie plików

Pracując z systemem komputerowym zwykle tworzymy różnego rodzaju pliki. Są to teksty programów, dokumenty do wydrukowania, notatki, nagrywamy dźwięk czy obraz. Rzadko zdarza się, że pierwsza wersja naszego dzieła jest ostateczna. Wtedy poprawiamy to, co do tej pory zrobiliśmy. W końcu po wielu poprawkach uzyskujemy postać, którą uznajemy za ostateczną. Po pewnym czasie nawet jednak ta *wersja ostateczna* wymaga zmian z różnych powodów. Często okazuje

się, że chcemy odtworzyć ścieżkę uzyskania konkretnej postaci naszych plików. Jak dotrzeć do wcześniejszych *wersji* plików?

Z drugiej strony niektóre prace musimy wykonać we współpracy z innymi członkami *zespołu projektowego,* w którym uczestniczymy. Jak pracować nad jednym plikiem *jednocześnie?* W takim przypadku potrzeba dotarcia do wcześniejszych wersji plików jest dużo bardziej widoczna, choćby w celu prześledzenia toku pracy innych członków zespołu. Kolejnym rzeczywistym problemem jest wspólna i *zdalna* (tj. przez sieć) praca nad jednym plikiem czy zestawem plików. A już zupełnie oczywista jest konieczność tworzenia kopii zapasowych plików, na wypadek wszelkich problemów z pamięcią masową.

Odpowiedzią na te wszystkie problemy i wyzwania są systemy wersjonowania plików. Są one niezbędne w procesie wytwarzania oprogramowania, w którym dodatkowo pojawia się konieczność publikacji kolejnych, udoskonalonych wersji pakietów oprogramowania.

Obejrzyj film https://git-scm.com/video/what-is-version-control.

3.2 System kontroli wersji Git

Git, https://git-scm.com/, jest najpopularniejszym systemem wersjonowania plików, ze względu na elastyczność i spełnianie często rozbieżnych potrzeb członków zespołu projektowego. Jest to system, w którym pojęcie projektu równa się katalogowi w lokalnym systemie plików. Może także pracować ze zdalnym repozytorium – wtedy stanowi narzędzie pracy grupowej.

Używanie tego systemu jest polecane w każdej aktywności, w której powstają jakieś pliki, stanowiące rezultat pracy.

Obejrzyj filmy https://git-scm.com/video/what-is-git i https://git-scm.com/video/get-going.

3.3 Zadanie: ustaw parametry globalne

Do parametrów globalnych należy m.in. nazwa i email użytkownika. Należy wykonać z wykorzystaniem terminala:

```
git config --global user.name "Imię Nazwisko"
git config --global user.email "mail_użytkownika@jakaś.domena"
```

3.4 Zadanie: utwórz lokalne repozytorium

W oknie terminala utwórz pierwsze repozytorium:

```
git init lab2
```

Obejrzyj powstałą strukturę katalogów i plików, ale niczego nie modyfikuj. W katalogu projektu utwórz plik z programem typu *Hello world!* Dodaj jego źródło do repozytorium:

```
git add <nazwa pliku źródłowego>
```

Projekt „NERW 2 PW. Nauka – Edukacja – Rozwój – Współpraca" współfinansowany jest ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

Fundusze Europejskie Wiedza Edukacja Rozwój
Rzeczpospolita Polska
Politechnika Warszawska
Unia Europejska Europejski Fundusz Społeczny

Zwróć uwagę, że polecenie `add` niczego jeszcze nie dodaje do repozytorium, informuje ono tylko Gita, że chcesz włączyć wskazany plik w jego aktualnej postaci do bieżącej struktury wersjonowanych plików. Aby umieścić stan prac w repozytorium wykonaj operację `commit`, dopiero ona aktualizuje repozytorium:

```
git commit -m"Pierwsza wersja programu"
```

Operacja `commit` powinna być zawsze opatrzona jednozdaniowym komentarzem opisującym zgrubnie wprowadzone zmiany (opcja `-m`).

Często sprawdzaj stan repozytorium:

```
git status
```

W katalogu projektu utwórz plik `.gitignore`, który posłuży wskazaniu plików, których nie chcesz wersjonować. Każda linia w `.gitignore` określa plik lub grupę plików wyłączonych z wersjonowania, obowiązują symbole wieloznaczne `*` i `?`. Znajdź w Internecie przykłady wpisów w pliku `.gitignore`.

Zwróć uwagę, że pliki, których nazwa zaczyna się od kropki są traktowane jako pliki *ukryte* w uniksopodobnych systemach operacyjnych. Systemy typu Windows nie stosują tej zasady, ich pliki ukryte muszą mieć ustawiony odpowiedni atrybut. W obu rodzajach systemów jednak obowiązuje nazwa pliku `.gitignore`.

Jakie wpisy warto stworzyć w `.gitignore` gdy wykorzystywanym językiem jest C? Których plików nie warto przechowywać w systemie wersjonowania? Dlaczego?

Zapoznaj się z książką Pro Git, https://git-scm.com/book/en/v2. Niektóre jej rozdziały zostały przetłumaczone na język polski.

3.5 Wydziałowa instalacja GitLab

System Git może być także używany w chmurze. W takim przypadku istnieje interfejs użytkownika takiego systemu, w zależności od rozwiązania, dostępny w przeglądarce internetowej lub w specjalizowanej aplikacji.

Bardzo popularny jest system GitHub, https://github.com/, powstały jako pierwszy, poważny system wsparcia rozproszonego i wielodostępowego wersjonowania plików dla systemu Git. Wydział Elektroniki i Technik Informacyjnych oferuje swoim Studentom inny, podobny system o nazwie GitLab. Post https://usersnap.com/blog/gitlab-github/ przedstawia interesujące porównanie własności obu systemów.

Pod adresem https://gitlab-stud.elka.pw.edu.pl/ dostępna jest wydziałowa instalacja GitLab. Należy się do niej zalogować, korzystając z zakładki LDAP, w której używamy tej samej nazwy użytkownika i hasła, które obowiązują przy logowaniu do komputerów w laboratorium. Dostęp do wydziałowego GitLaba jest możliwy z zewnątrz Uczelni, dzięki czemu można go używać do wszelkich celów związanych z tokiem studiowania. Należy pamiętać o przestrzeganiu regulaminu korzystania z sieci Wydziału. W ramach przygotowań do zajęć warto zapoznać się z tym systemem wcześniej.

Projekt „NERW 2 PW. Nauka – Edukacja – Rozwój – Współpraca” współfinansowany jest ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

Fundusze Europejskie Wiedza Edukacja Rozwój
Rzeczpospolita Polska
Politechnika Warszawska
Unia Europejska Europejski Fundusz Społeczny

3.6 Zadanie: utwórz projekt w wydziałowym GitLabie i prześlij tam pliki ze swojego lokalnego repozytorium

Zaloguj się do Gitlaba wykorzystując swój login i hasło.

Wybierz `New project`, nazwij projekt w formacie: POPRO_20Z_Nazwisko_Imię i go utwórz. **UWAGA:** To będzie Twoje repozytorium przez całe laboratorium w tym semestrze. Każde kolejne zajęcia laboratoryjne powinny być tworzone w nowym katalogu tzn. lab i numer. W przypadku niepoprawnej nazwy repozytorium lub nazwy katalogu dotyczącego zajęć, prace nie będą sprawdzane.

Po utworzeniu projektu system pokazuje kilka wskazówek dotyczących integracji zdalnego repozytorium z lokalnym. Wykonaj polecenia wysłania lokalnego repozytorium do repozytorium zdalnego.

Dodaj do repozytorium obu prowadzących tzn. w zakładce *Members* - *Invite member* wpisz login prowadzących: *ldabala* oraz *kignasiak*. Wybierz rolę *developer* dla członka zespołu. Następnie kliknij przycisk *Invite.*

3.7 Zadanie: seria modyfikacji bazowego programu

1. Dodaj możliwość pobierania danych od użytkownika z wykorzystaniem parametrów wywołania programu. Należy przyjąć, że będziemy podawać imię użytkownika oraz jego pseudonim jako pierwszy i drugi parametr wywołania. Następnie wypisz na ekran "Hello world", imię oraz pseudonim użytkownika. Wykonaj commit oraz wyślij zmiany na serwer.

2. Należy zmodyfikować program tak, aby obliczał średnią liczb z podanych parametrów. Pamiętaj, że liczby do średniej będą od 3 parametru wywołania! Tak zmodyfikowaną wersję należy umieścić w osobnym branchu (nazywając branch np. z_change) w repozytorium.

3. Należało nagle zmodyfikować stabilną wersję programu (znajdującą się w branchu master). Zmień program tak, aby imię było wypisywane na końcu. Stwórz nowy branch i wyślij zmiany na serwer.

4. Należy zmodyfikować program tak, aby zwracał jedynie część całkowitą ze średniej. Taką wersję należy umieścić w osobnym branchu.

5. W tym momencie w repozytorium powinny być 3 branche. Należy nowe funkcjonalności wprowadzić do stabilnej wersji (branch master). W tym celu połącz ze sobą wszystkie branche (wykorzystując operację merge!). Program powinien wypisać: "Hello world" pseudonim imię oraz wartość całkowitą ze średniej.

4 Sprawozdania z zajęć

Niektóre laboratoria wymagają wykonania sprawozdania z przeprowadzonych prac. Najczęściej sprawozdanie ma formę dokumentu, który można wydrukować. W sprawozdaniu muszą znaleźć się następujące informacje:

Projekt „NERW 2 PW. Nauka – Edukacja – Rozwój – Współpraca" współfinansowany jest ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

Fundusze Europejskie Wiedza Edukacja Rozwój
Rzeczpospolita Polska
Politechnika Warszawska
Unia Europejska Europejski Fundusz Społeczny

- Imię, nazwisko, numer indeksu, numer grupy, email studenta/studentów wykonujących zadania.
- Data i godzina zajęć, których dotyczy sprawozdanie.
- Nazwa przedmiotu, tytuł/numer zadania/ćwiczenia.

Następnie należy opisać wykonane prace, zamieszczając wymagane elementy: rysunki, zestawienia tabelaryczne, zrzuty z ekranu, fragmenty kodu itp. Instrukcja do ćwiczenia/laboratorium dokładnie sprecyzuje konieczne elementy.

Sprawozdanie należy zakończyć wnioskami. Warto też podzielić się własnymi uwagami, które zdaniem wykonawców pomogą ocenić zrozumienie tematu i osiągnięcie celu zajęć.

4.1 Zadanie: sprawozdanie z tych ćwiczeń laboratoryjnych

W ramach sprawozdania z tych zajęć utwórz plik tekstowy, w którym opiszesz swoje wrażenia z pracy z narzędziem Git. Do czego jest ono przydatne? W jaki sposób można radzić sobie z konfliktami? Czy wykorzystanie narzędzi graficznych ułatwia pracę z Gitem?

Czy były jakieś problemy z wykonaniem modyfikacji programu? Z czego one wynikały? Czy wystąpił konflikt w trakcie operacji merge różnych branchy?

Literatura

[1] Git Version Control System – https://git-scm.com/

[2] Dokumentacja Git – https://git-scm.com/doc

[3] Książka Pro Git – https://git-scm.com/book/en/v2

[4] System GitHub – https://github.com/

[5] System GitLab – https://about.gitlab.com/

[6] Porównanie GitLab i GitHub – https://usersnap.com/blog/gitlab-github/

Projekt „NERW 2 PW. Nauka – Edukacja – Rozwój – Współpraca” współfinansowany jest ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

Fundusze Europejskie Wiedza Edukacja Rozwój | Rzeczpospolita Polska | Politechnika Warszawska | Unia Europejska Europejski Fundusz Społeczny