Platforma GitHub

1 Cel laboratoriów

Zapoznanie się z działaniem platformy GitHub. Założenie konta i nauka współdzielenia źródeł.

2 GitHub

„GitHub – hostingowy serwis internetowy przeznaczony dla projektów programistycznych wykorzystujących system kontroli wersji Git. Stworzony został przy wykorzystaniu frameworka Ruby on Rails i języka Erlang. Serwis działa od kwietnia 2008 roku[1]. W kwietniu 2011 ogłoszono, iż GitHub obsługuje 2 miliony repozytoriów[2]. Github udostępnia darmowy hosting programów open source oraz płatne prywatne repozytoria."[1]

Główne cele:

1. wymiana kodu i serwis publikowania kodu
2. website społeczny dla programistów

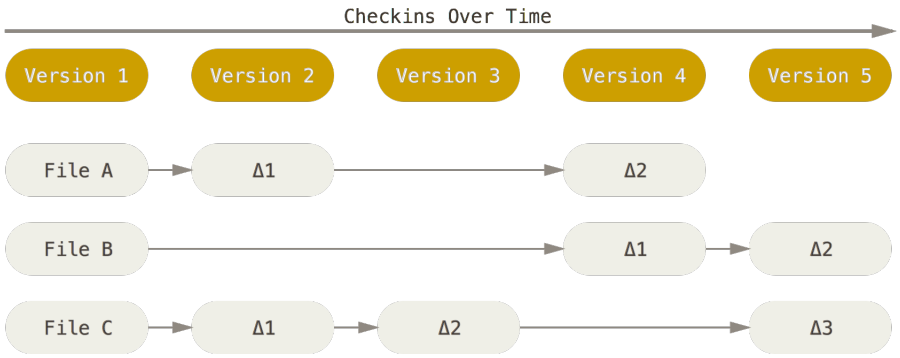
2.1 Git

Najważniejszym elementem jest Git - system kontroli wersji, którego pierwszym twórcą był Linus Torvalds. Git jak każdy system kontroli wersji zarządza i archiwizuje kolejne poprawki w projekcie. Jest głównie przeznaczony dla kodu, ale może być również używany w plikach takich jak dokumenty Worda.

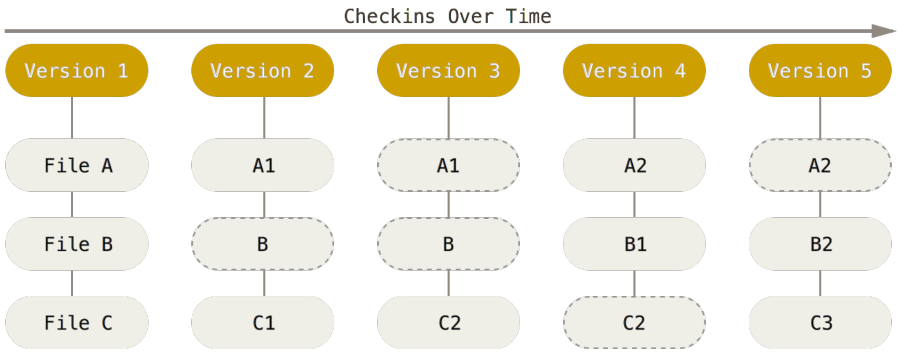
Niektóre z poprzedników Git, takie jak CVS i Subversion, posiadaj centralne „repozytorium" wszystkich plików związanych z projektem. Gdy programista wprowadza zmiany, zmiany te wprowadzane są bezpośrednio do centralnego repozytorium. W rozproszonych systemów kontroli wersji, takich jak Git, jeśli chcesz wprowadzić zmiany w projekcie można skopiować całość repozytorium do własnego systemu. Możesz dokonać zmian w lokalnej kopii, i poprzez „check in" dokonać zmian w centralnym serwerze. Taki system zachęca do tworzenia granularnych zmian, ponieważ system nie wymusza łączności z serwerem za każdym razem kiedy dokonuje się zmiany.

Różnice pomiędzy działaniem systemów kontroli wersji takich jak CVS, Subversion, Perforce, Bazaar wyjaśnia Rysunek 1 i 2. W Git za każdym razem, gdy wykonasz zatwierdzenie

[1] http://pl.wikipedia.org/wiki/GitHub



Rysunek 1: Kontrola wersji jako lista zmian w plikach źródłowych: źródło:http://git-scm.com/book/en/v2/Getting-Started-Git-Basics



Rysunek 2: Kontrola wersji w Git: źródło:http://git-scm.com/book/en/v2/Getting-Started-Git-Basics

zmian „commit" lub zapiszesz projekt Git zapamiętuje stan (jako migawkę) wszystkich plików w tym momencie. Jeżeli plik się nie zmienił, przechowuje się tylko link do pliku, który był już zapisany wcześniej.

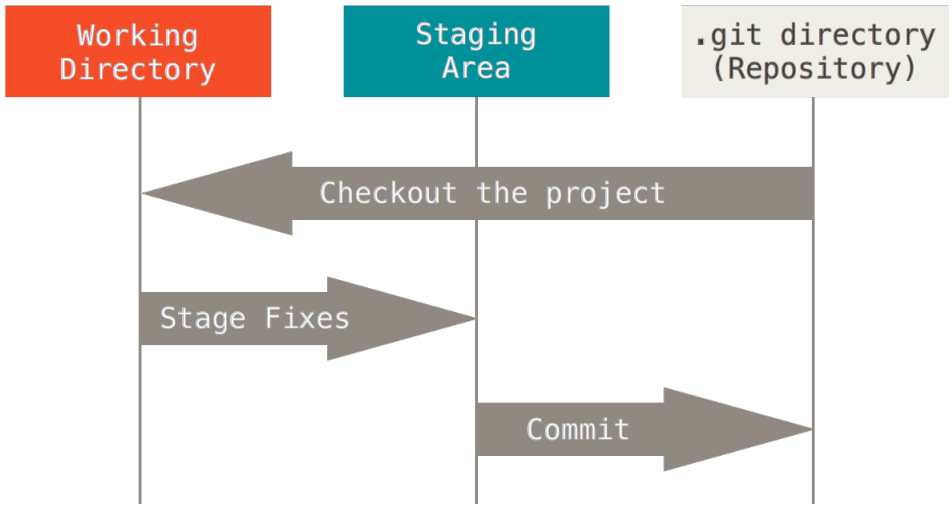
Główne cechy GIT to:

1. Prawie wszystkie operacje wykonywane są lokalnie — historia zmian jest przechowywana lokalnie, porównania plików z poprzednich wersji

2. Integralność — każda operacja jest związana z generowaniem sumy kontrolnej (SHA-1) na pliku. Suma ma format 40 znakowe ciągu heksadecymalnego np. $24b9da6552252987aa493b52f8696cd6d3b00373$

3. Dodawanie — Git głównie dodaje elementy, usuwanie jest wieloetapowym zadaniem i zabezpiecza przed utratą danych.

4. Trzy stany — pliki mogą się znajdować w jednym z trzech stanów: zmodyfikowany, staged (plik został zmodyfikowany i czeka by przenieść bazy zmian) i commited (pliki zapisane w bazie zmian) (rysunek 3).



Rysunek 3: Trzy stany pliku. źródło:http://git-scm.com/book/en/v2/Getting-Started-Git-Basics

2.2 GitHub

GitHub jest usługą hostingową dla repozytorium Git i dodaje wiele własnych elementów. Git jest narzędziem obsługiwanym z linii komend, GitHub zapewnia graficzny interfejs webowy. Zapewnia również kontrolę dostępu i kilka możliwości współpracy, takich jak wiki i podstawowe narzędzia zarządzania zadaniami dla każdego projektu.

Flagową funkcjonalnością GitHub jest „forking" - kopiowanie repozytorium z konta jednego użytkownika do drugiego. Umożliwia to wykorzystanie projektu, do którego nie masz dostępu do zapisu i możesz modyfikować go na swoim koncie. Jeśli dokonasz zmian, które chcesz udostępnić, możesz wysłać zgłoszenie zwane „pull request" do pierwotnego właściciela. Właściciel może następnie, za pomocą kliknięcia, scalić zmiany znalezionych w twoim repo z oryginalnym repo.

Dotychczas, jeżeli chciało się włączyć do projektu open source trzeba było ręcznie pobrać kod źródłowy projektu, wprowadź zmiany lokalnie, następnie utworzyć listę zmian zwanych „patch", a następnie wysłać mailem do opiekuna projektu. Opiekun musiał ocenić tę poprawkę, czasami wysłany przez zupełnie nieznanego kodera, i zdecydować, czy do zmiany nadają się do scalenia.

W środowisku sieciowym jakim jest GitHub ta procedura wygląda inaczej. Kiedy koder złoży „pull request", opiekun projektu może zobaczyć jego profil, który obejmuje wkład w projekty zapisane w GitHub. Jeśli poprawka zostanie przyjęta, można uzyskać kredyt od opiekuna, i to widać w profilu. GitHub jest jak CV, które pomaga określić dorobek programisty. Im więcej ludzi i projektów na GitHub, tym lepszy obraz daje to opiekunom projektów. Poprawki mogą być publicznie dyskutowane.

3 Zadania do wykonania

Celem zadań jest nauka wykorzystania Git i GitHub wraz z założeniem konta.

3.1 Git - instalacja i konfiguracja

1. Przejdź do strony: `http://git-scm.com/downloads` lub `https://git-for-windows.github.io`. Drugi adres zawiera narzędzia okienkowe do obsługi Git.
2. Zainstaluj Git
3. Otwórz Git Bash, który jest CLI (Command Line Interface). Przetestować w nim polecenia: pwd, clear, ls, ls -l, ls -a, ls -al, cd .., cd, mkdir, touch, cp, rm, mv, echo, date.
4. Każde zatwiedzenie zmian „commit" jest będzie oznaczone nazwą użytkownika i adresem e-mail. Dlatego wykonaj następujące polecenia

   ```
   $git config --global  user.name ,,Your name here''
   $git config --global  user.email ,,Your email here''
   ```

   Wykonanie polecenia po raz kolejny doprowadzi do zmiany tych danych.
5. Sprawdź zmiany

   ```
   $git config --list
   ```
6. Aby wyjść z basha wpisz:

   ```
   $exit
   ```

3.2 GitHub - założenie konta i konfiguracja

1. Przejdź do strony: `https://github.com/`
2. Wprowadź: username, email i password i kliknij „Sign up for GitHub"
3. Uwaga: użyj tego samego adresu, który został użyty do konfiguracji Git
4. Na następnym ekranie kliknij „Free plan„ and kliknij „Finish sign up"
5. Na ekranie powitalnym wiele pomocnych linków (sprawdź)
6. Kliknij na swoje imię w prawym górnym rogu, by zobaczyć swój profil. Tu znajdują się informacje o aktywności na koncie, kim jesteś i nad czym pracujesz, powoli to miejsce stanie się Twoim portfolio. Można swój profil uzupełnić (wg uznania).
7. Poeksploruj portal ;-) znajdź kolegów i ciekawe projekty.

3.3 Lekcja - nauka poleceń i działania zdalnie

Wykorzystaj adres: `http://try.github.io`

3.4 GitHub tworzenie repozytorium

Git jest na lokalnym komputerze, GitHub na serwerze (współdzielenie danych oraz kopia zapasowa Twojej pracy).

Dwie metody tworzenia repozytorium

1. własne z plików
2. fork innego projektu

Metoda pierwsza:

1. Idź do strony: (`https://github.com/yourUserNameHere/`) i kliknij na „Create a new repo" w prawym górym rogu lub
2. Idź do strony `https://github.com/new` zaloguj się.
3. Nadaj nazwę i wpisz krótki opis
4. Wybierz „Public" (private jest dostępny tylko dla użytku komercyjnego lub edukacji)
5. Zaznacz box przy „Initialize this repository with a README"
6. Kliknij „Create repository"

Stworzenie lokalnej kopii:

1. Otwórz Git Bash
2. Wykonaj polecenia

```
$ mkdir ~/test-repo
$ cd ~/test-repo
$ git init
$ git remote add origin https://github.com/yourUserNameHere/test-repo.git
```

Metoda druga:

1. Znajdź repo, które jest godne uwagi i kliknij na „Fork" (`https://help.github.com/articles/fork-a-repo`)
2. Lokalna kopię repozytorium (do katalogu bieżącego) wykonuje się poleceniem

```
$ git clone https://github.com/yourUserNameHere/repoNameHere.git
```

4 Literatura obowiązkowa

1. `https://git-scm.com/book/pl/v1/Pierwsze-kroki-Podstawy-Git`