Daniel Górski
Korporacyjna Java
Wykład 3: Pull-request'y

Projekty wiele lat temu...

- Były znacznie mniejsze niż dzisiaj
- Poziom wejścia aby zostać programistą był dużo wyższy
- Zasoby były dużo droższe i one były głównym wąskim gardłem

Projekty dzisiaj

- Złożoność projektów wszerz wzrosła o rzędy wielkości
- Powstało mnóstwo narzędzi wspierających
- Zasoby sprzętowe są tanie w porównaniu do minionych dekad
- Pomimo obniżenia się poziomu wejścia w rolę programisty, obecnie wąskim gardłem są zasoby ludzkie
  - Możliwe, że obecnie zachodzi tutaj zmiana...

Projekty w przyszłości...

- Sztuczna inteligencja może zamieszać w branży
  - Zaczyna się to powoli dziać
- Możliwe dalsze jeszcze węższe specjalizacje
- Osoby mające dużą wiedzę w szerokich dziedzinach również będę potrzebne
- Możliwy mniejszy popyt na pracę nisko-wykwalifikowanych programistów
  - Wydaje się, że właśnie zaczynamy wchodzić w ten punkt zwrotny

Praca zespołowa kiedyś...

- Zespoły były znacznie mniejsze niż obecnie
- Zespoły były znacznie mniej rozproszone niż obecnie
- Synchronizacja zmian była łatwiejsza
- Bardziej płaska struktura: większa odpowiedzialność za projekt

Praca zespołowa obecnie

- Projekty stały się większe
- Zespoły stały się większe i rozproszone
- Próg wejścia aby zostać programistą obniżył się
- Projekty są dużo bardziej korporacyjne
  - Wielopoziomowa struktura hierarchiczna
  - Rozliczanie się z wykonanych zadań
  - Częstsze rotacje, choć dużo zależy od firmy
- Wydajne repozytorium jest koniecznością!

Git i mechanizm pull-request

- Tworzymy swój nowy branch z bieżącej wersji innego branch'a
- Wprowadzamy zmiany do swojego branch'a
- Gdy skończyliśmy pracę to tworzymy pull-request Naszego branch'a do wyjściowego
  - To jest moment na automatyczne wykonanie testów
  - Inne osoby z zespołu oglądają Nasze zmiany i zatwierdzają je
  - Po zebraniu odpowiedniej liczby akceptacji Nasz kod jest mergowany do wyjściowego branch'a
- Cały proces jest dobrze zautomatyzowany

Podstawowe przepływy

- Podstawowa ścieżka:
  - Branch master
  - Branch'e zadaniowe
  - Pull-request branch'y zadaniowych do master'a
- Klasyczna bardziej rozbudowana:
  - Branch'e master i stagging
  - Branch'e zadaniowe
  - Pull-request branch'y zadaniowych do stagging'a
  - Co jakiś czas duże testy i podmiana master'a
  - Awaryjne pull-request'y do master'a części stagging

Pull-request: podstawowe problemy

- Branch z którego wyszliśmy zmienił się do momentu gdy chcemy wystawić PR
  - Merge / Rebase ostatnich zmian
  - Sprawdzenie spójności
- Inny PR został zmergowany i Nasz ma teraz konflikty
  - Automatyczne powiadomienia i merge / rebase, spójność

Pull-request: podstawowe problemy

- Duża nowa funkcjonalność -> duży PR
  - Ciężko przejrzeć wiele zmian i inni programiści są niechętni do zaakceptowania
  - Duża szansa na pojawienie się konfliktów
  - Trudność w późniejszym roll-back'u i znalezieniu źródła błędu
- Z tego powodu warto duże zadania dzielić na mniejsze
  - O ile jest możliwe wykonanie sensownego podziału

Pull-request: nowe możliwości

- Korporacja ma standardowe narzędzie do wprowadzania zmian w kodzie
  - Dużym pokryciem kodu testami może zmniejszyć ryzyko błędów, pisanie dużej ilości testów jest kosztowne
  - Wprowadzeniem automatycznych reguł "clean-code" może ustandaryzować kod, zbyt rygorystyczne reguły mogą być przeciwskuteczne i wymagają nadmiernego nakładu pracy programistów

Pull-request: nowe możliwości

- Lider techniczny zespołu ma duże możliwości utrzymywania kodu wysokiej jakości
- Programiści mają możliwości sprawnego poruszania się po historii zmian w kodzie
- Pull-request jest też platformą wymiany informacji o stylu / dobrych praktykach...

Pull-request: nowe specjalizacje

- Devopsi
  - Ustawienie procesu budowy / testowania / oceny jakości kodu / automatycznego wdrażania
  - Konfiguracja automatycznie tworzonych środowisk testowych
- Osoby robiące review
  - Review pod kątem clean-code
  - Review architektoniczne (SOLID)
  - Review wydajnościowe

Ideał CI (Continuous Integration)

- Projekt ma bardzo dobre pokrycie testami
- Każdy push zmian do repozytorium odpala testy całego projektu / serwisu
- Jest oceniana jakość kodu za pomocą automatycznego narzędzia
- Zmergowane zmiany są automatycznie / półautomatycznie wdrażane na produkcję / środowisko testowe

Pracownia

- Stworzyć 4-osobowe zespoły i zgłosić mi ich skład wraz z unikalną nazwą
- Założyć zespołowe repozytorium
  - Nadać w ramach zespołu uprawnienia
  - Nadać mi uprawnienie do odczytu (zylo-pl)

Pracownia: projekt

- Projektem będzie aplikacja konsolowa wspierająca rozwiązywanie zadań geometrycznych
- Pierwszym zadaniem jest dodanie dwóch poleceń (bezparametrowych):
  - version: zwraca informację o aktualnej wersji aplikacji
  - help: zwraca informację o obsługiwanych poleceniach
- Językiem wspieranym ma być język angielski

Harmonogram

- Przed wykładem od 7:30 jestem dostępny na prezentację programu z wymaganiami z zeszłego tygodnia (od 3-ciej pracowni wymagania, czyli od 4-tego wykładu jestem wcześniej)
- 8:15 – 9:45: Wykład
- 9:45 – 10:00: Przerwa
- 10:00 – 11:30: Pracownia: jestem dostępny na prezentację programu + bieżące pytania techniczne