Platformy programistyczne: .NET i Java

WYKŁAD 1: WPROWADZENIE

Kto, co, jak i kiedy

Kto?

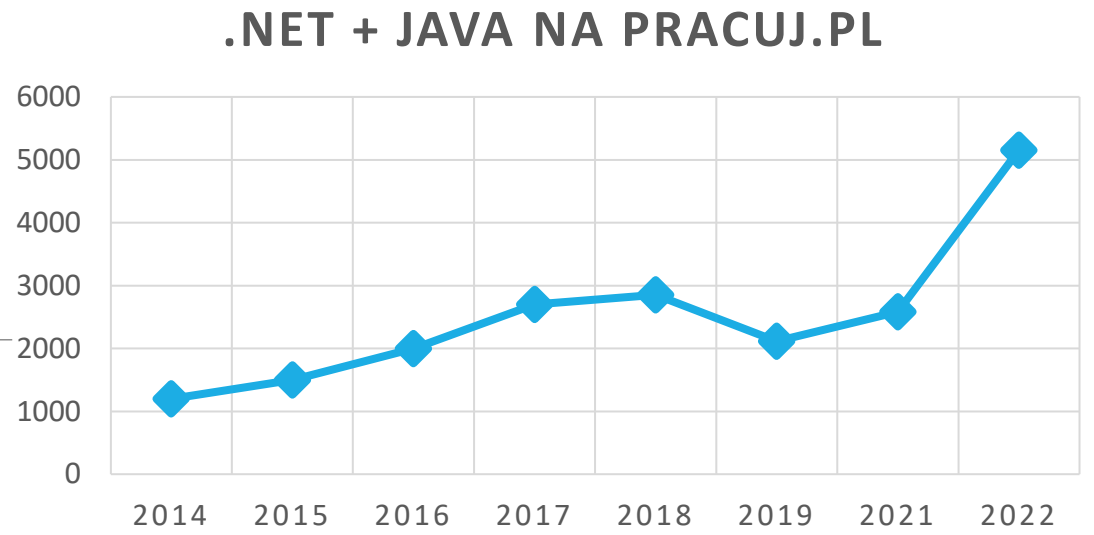
- dr inż. Mariusz Uchroński
- p. 215, C-3
- mariusz.uchronski@pwr.edu.pl

- dr inż. Bartosz Jabłoński
- s. P0.2, C-16
- bartosz.jablonski@pwr.edu.pl
- http://jablonski.wroclaw.pl

O co chodzi?

- .NET i Java są najpopularniejszymi platformami ogólnego przeznaczenia z dużym naciskiem na zastosowania sieciowe
- Są wykorzystywane w bardzo różnych dziedzinach i obszarach:
  - aplikacje desktopowe,
  - aplikacje webowe,
  - rozwiązania klient/serwer,
  - systemy wbudowane,
  - aplikacje mobilne,
  - aplikacje w chmurze,
  - ...

Co Wam to da?



- Szersze spojrzenie na świat :)
- (Darmowy!) dostęp do aktualnej wiedzy z dziedziny
- Nowe możliwości zatrudnienia (ponad 5000 aktywnych ofert dla programistów .NET lub Java na portalu Pracuj.pl na dzień 2022.02.24)
- Dobre pieniądze w przyszłości
  (doświadczony programista Java/.NET zarabia teraz 12 000 – 27 000 zł)
- Możliwość wykorzystania tej wiedzy w innych dziedzinach (niekoniecznie jako programiści)
- Możliwość zaliczenia tego kursu…

Jak? - Wykład

- Wykład 1: Zajęcia wprowadzające
- Wykład 2: Wprowadzenie do platformy .NET (BJ)
- Wykład 3: Język C#, .NET w aplikacjach webowych (BJ)
- Wykład 4: Dynamic, kolekcje, LINQ, odlatujemy w chmury (AWS, Azure, …) (BJ)
- Wykład 5: Wprowadzenie do Javy (MU)
- Wykład 6: Java – wyjątki, klasy abstrakcyjne, interfejsy, wątki, Swing (MU)
- Wykład 7: Java – programowanie generyczne, kolekcje, gniazda sieciowe, bazy danych (MU)
- Wykład 8: (najprawdopodobniej) Kolokwium

Jak zaliczyć? – Wykład

- Wersja **optymalna** dla pracujących regularnie
  - Wykazałem się na laboratorium
  - Napisałem kilka niezapowiedzianych kartkówek na wykładzie
  - Jestem zwolniony z kolokwium ☺
- Wersja **dla hard-corowców**
  - Wybrałem wersję minimum na laboratorium
  - Nie napisałem za dużo kartkówek na wykładzie
  - Piszę kolokwium i dostaję pozytywną ocenę ☺

Zaliczenie

- F1 — Ocena sposobu wykonania zadania (uwzględniająca jakość napisanego kodu oraz zakresu zaimplementowanych funkcji częściowo w trakcie zajęć, a częściowo po ich zakończeniu), ocena poziomu nabytych umiejętności (na podstawie odpowiedzi na pytania związane z wykonanym zadaniem) w trakcie zajęć laboratoryjnych
- F2 — Odpowiedzi ustne lub pisemne z pytań zadawanych w trakcie wykładu Kolokwium końcowe

Ocena końcowa:

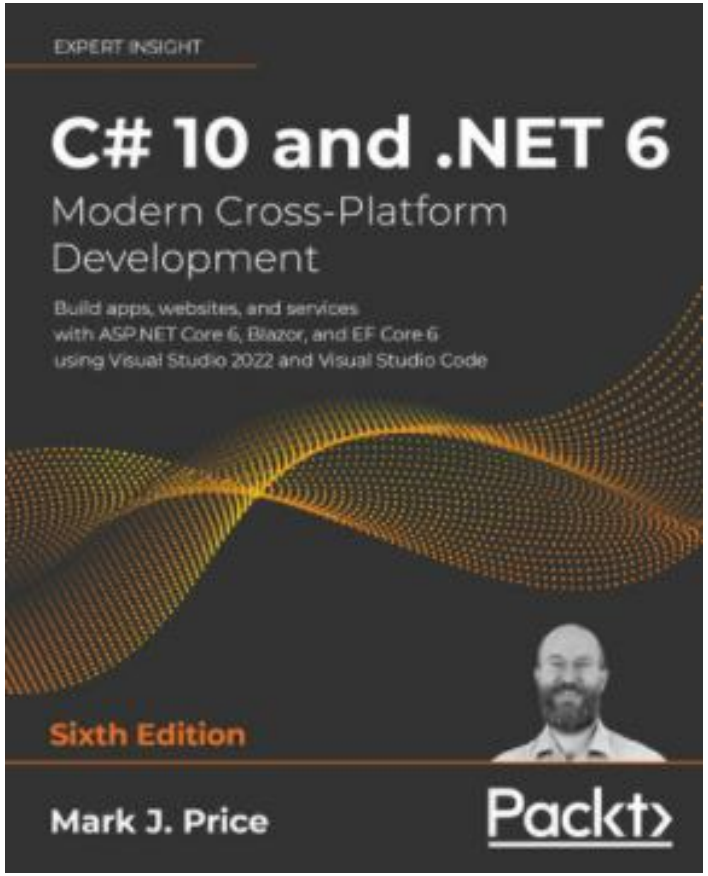
$$P = 0{,}5 * F1 + 0{,}5 * F2$$

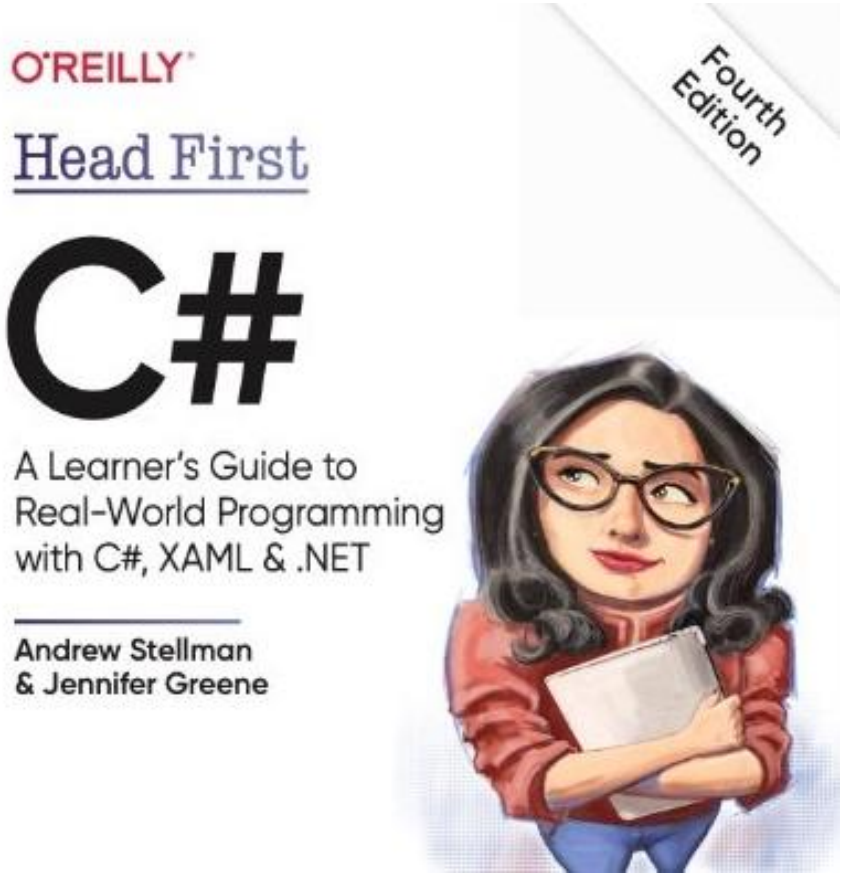
Koniecznie jest uzyskanie oceny pozytywnej z każdej formy realizowanych w ramach przedmiotu (laboratorium oraz wykład)

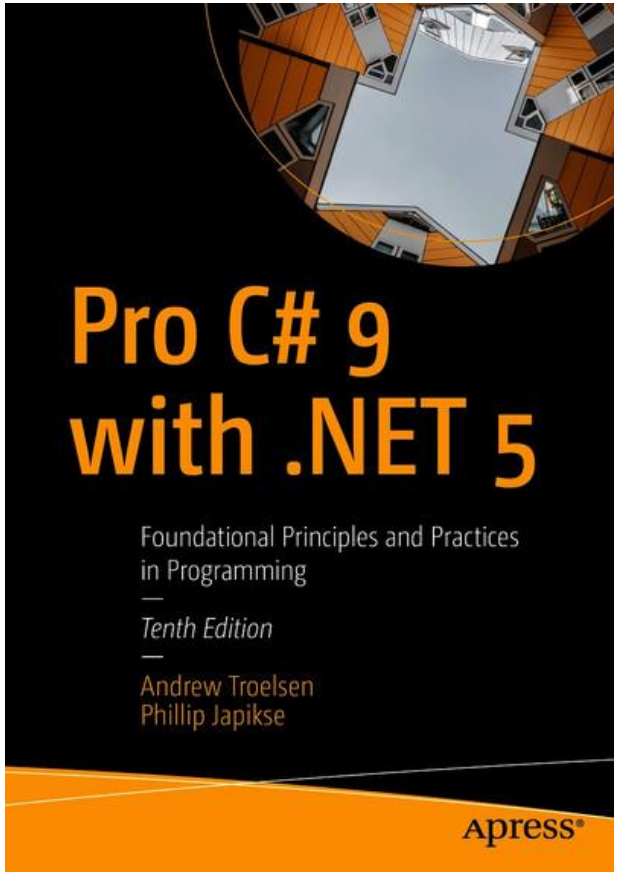
Na wykładzie świat się nie kończy

https://web.facebook.com/ekadotnet/

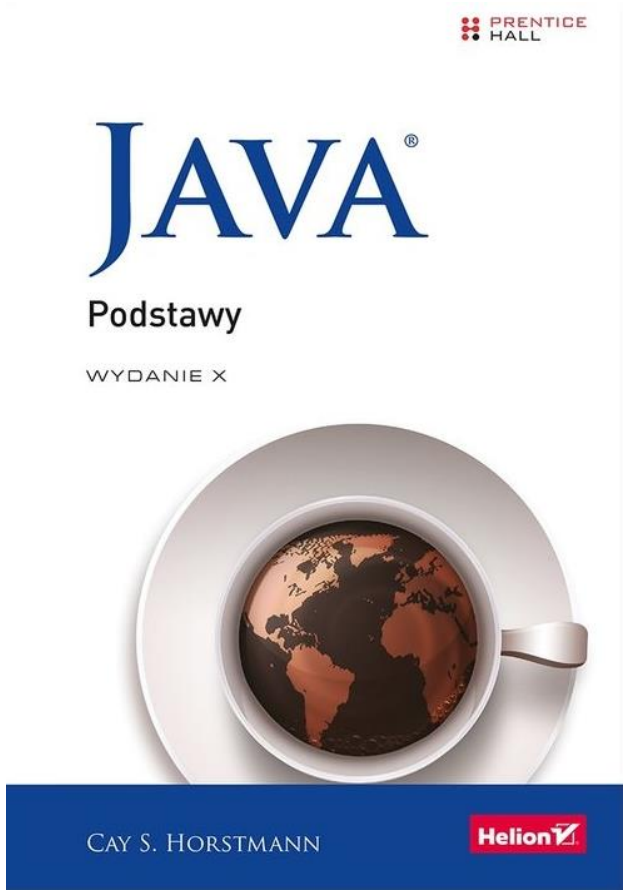
Literatura - .NET



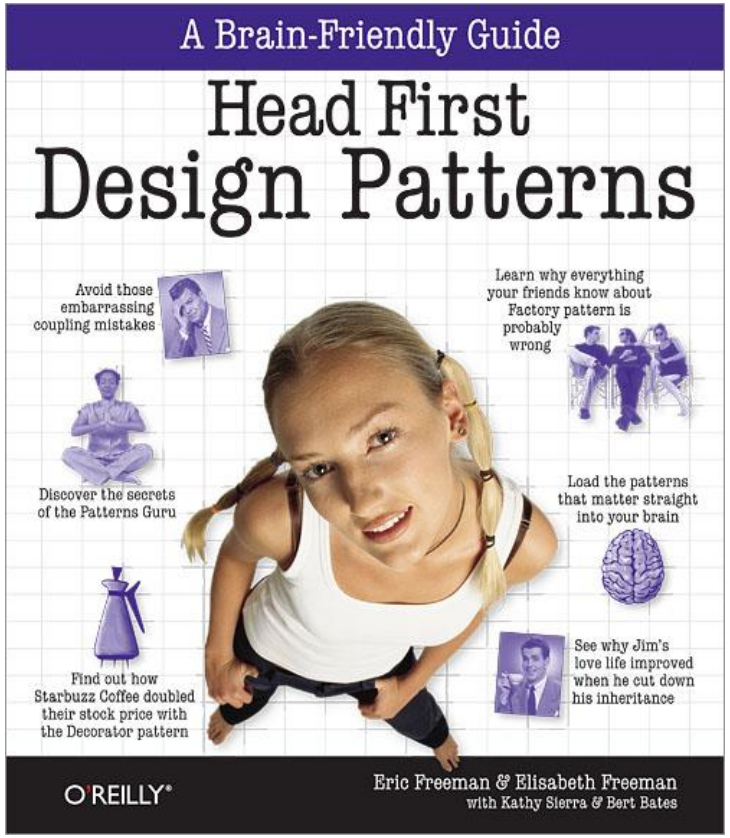


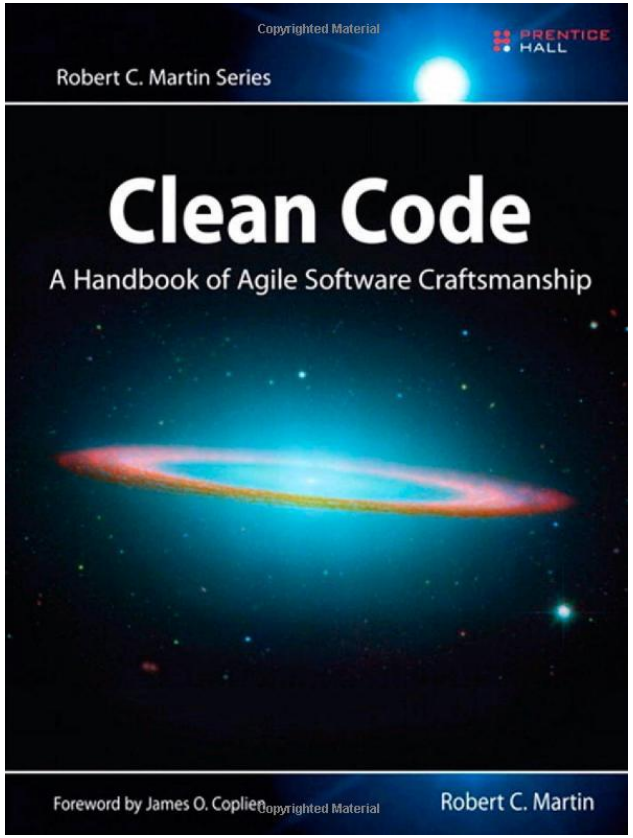


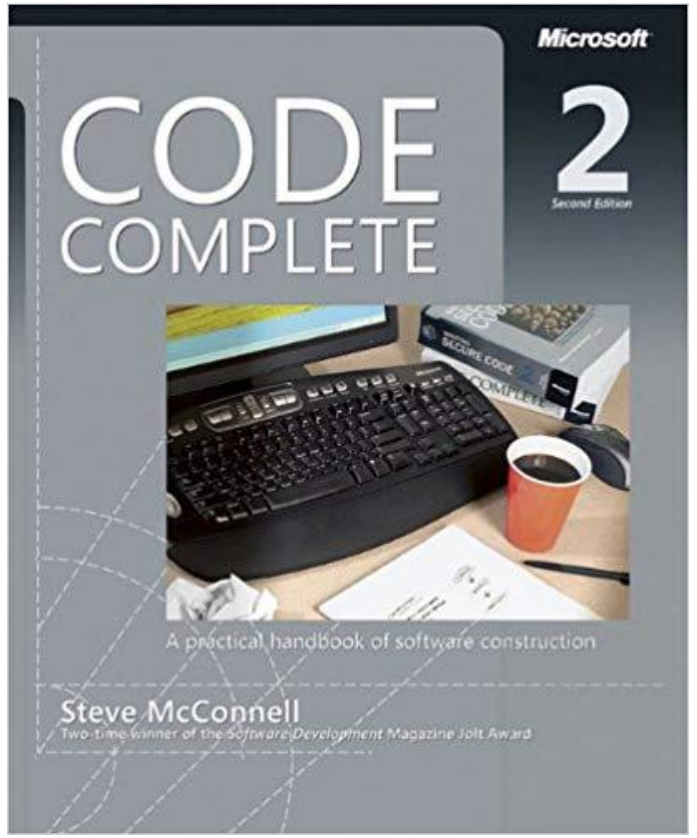
Literatura - Java



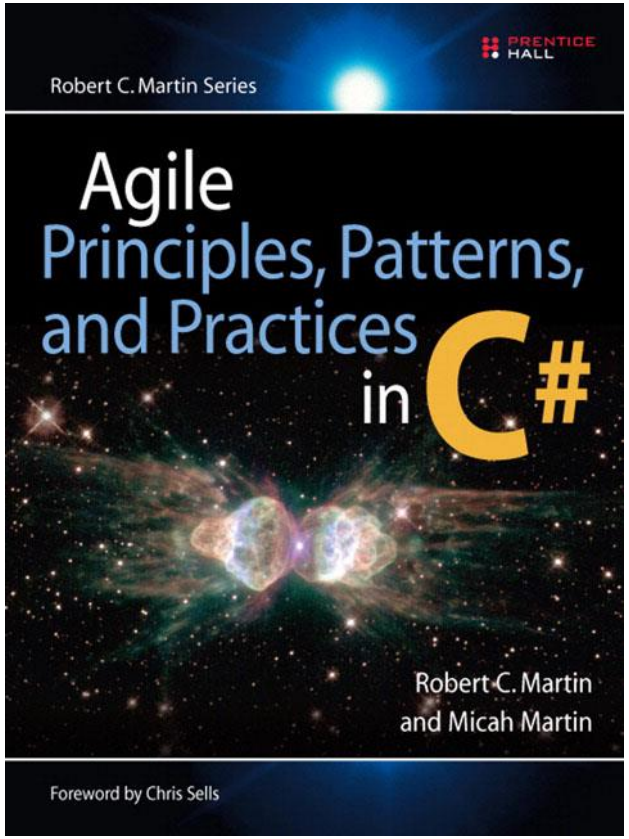
Literatura wspólna

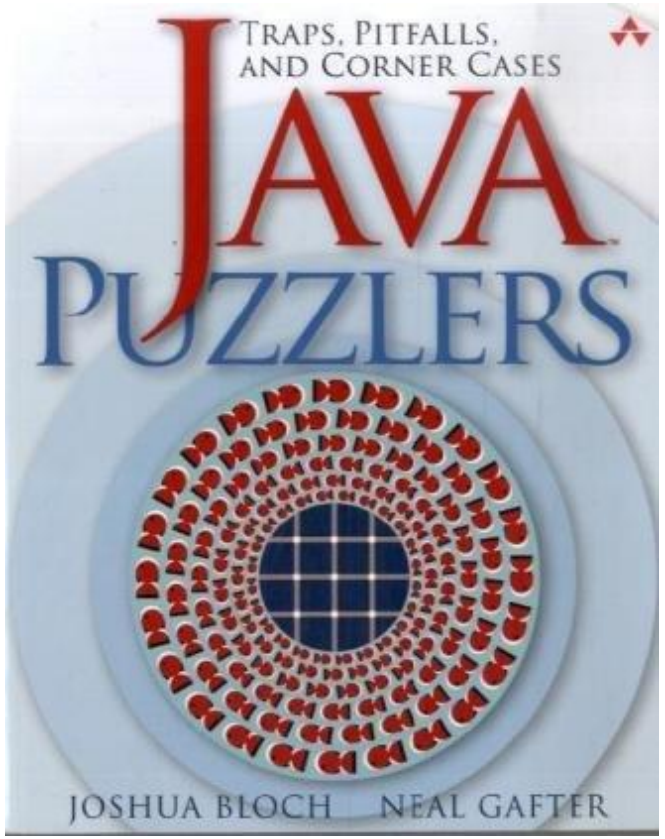


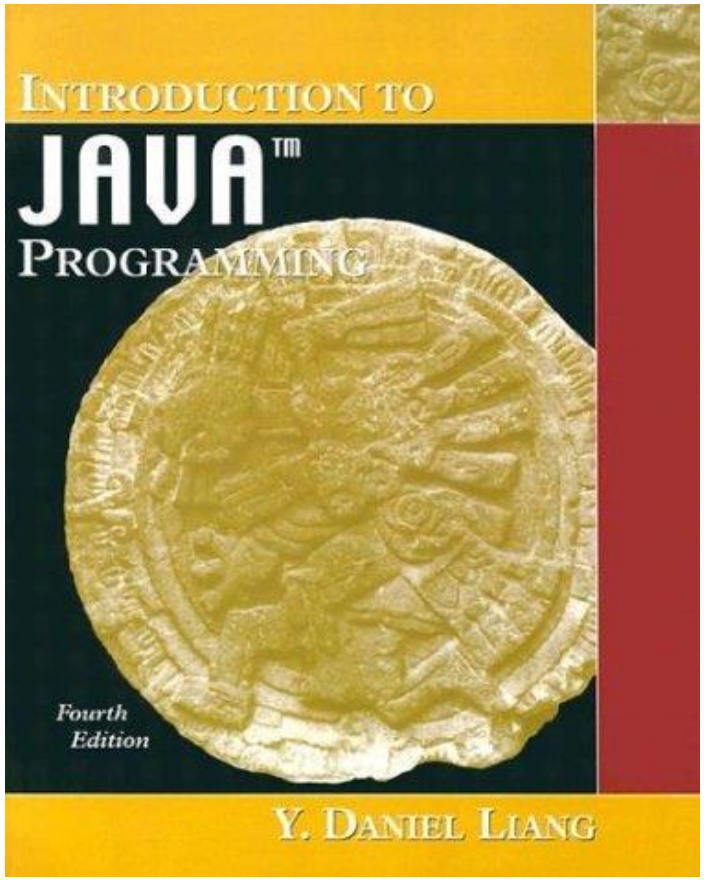




Literatura uzupełniająca







Efekty kształcenia

Z zakresu wiedzy:

- PEK_W01 – Zna specyfikę programowania w języku Java i w językach platformy .Net
- PEK_W02 – Zna możliwości zintegrowanych środowisk programowania dla platformy Java i .Net
- PEK_W03 – Zna różnice i podobieństwa między platformami .Net i Java oraz ich potencjał

Z zakresu kompetencji społecznych:

- PEK_K01 – ma świadomość wpływu jakości tworzonego kodu na możliwość jego dalszego rozwoju przez innych programistów.
- PEK_K02 – rozumie konieczność samodzielnego dokształcania się, szczególnie w obliczu ciągłej ewolucji technologii informatycznych i zmian słownika branżowego, używanego w komunikacji pomiędzy specjalistami.

Efekty kształcenia

Z zakresu umiejętności:

- PEK_U01 – Umie napisać prostą aplikację konsolową na platformie Java i .Net
- PEK_U02 – Umie napisać prostą aplikację okienkową na platformie Java i .Net
- PEK_U03 – Umie napisać prostą aplikację sieciową na platformie Java i .Net
- PEK_U04 – Umie zaprojektować i wykorzystać struktury danych dla platformy Java i platformy .Net
- PEK_U05 – Umie przygotować i przeprowadzić wdrożenie własnej aplikacji
- PEK_U06 – Umie wykorzystywać narzędzia typu repozytorium kodu Git

Narzędzia

Java

- Aktualna wersja: 8 LTS/11 LTS
- Producent: Oracle Corporation, do 2010 Sun Microsystems
- JRE – Java Runtime Environment – służy do uruchamiania programów napisanych w Javie
- JDK – Java Development Kit – jest zestawem narzędzi programisty (zawiera m.in. Kompilator)
- Programy kompilowane są do kodu bajtowego i wykonywane przez maszynę wirtualną (Java Virtual Machine)
- Środowiska programistyczne: Netbeans, IntelliJ IDEA, Eclipse

Eclipse IDE

- Darmowe zintegrowane środowisko programistyczne (IDE) środowisko programistyczne do tworzenia programów w Javie (i nie tylko).
- Projekt udostępniony na zasadach otwartego oprogramowania przez Eclipse Foundation.
- Platforma oferuje obsługę wtyczek umożliwiających korzystanie z systemów kontroli wersji, narzędzi budowania, tworzenie GUI, współpracę z serwerami aplikacji I baz danych, modelowanie za pomocą UML i wiele, wiele innych.
- Niezbędne linki:
  - Java Development Kit (JDK) 11.0.2 (LTS)
  - Eclipse Standard 4.10.0

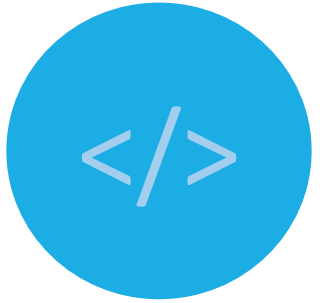
Visual Studio 2022

Visual Studio IDE

Universal Windows Platform
.NET Desktop
Windows Desktop
Web Development
Azure Development
Office/Sharepoint Development
Node.JS Development
Data Storage and Processing
Mobile Development
Game Development
VS Extension Development
Community/Proffesional/Enterprise

Visual Studio for Mac

A mobile-first, cloud-first IDE.
Made for the Mac.

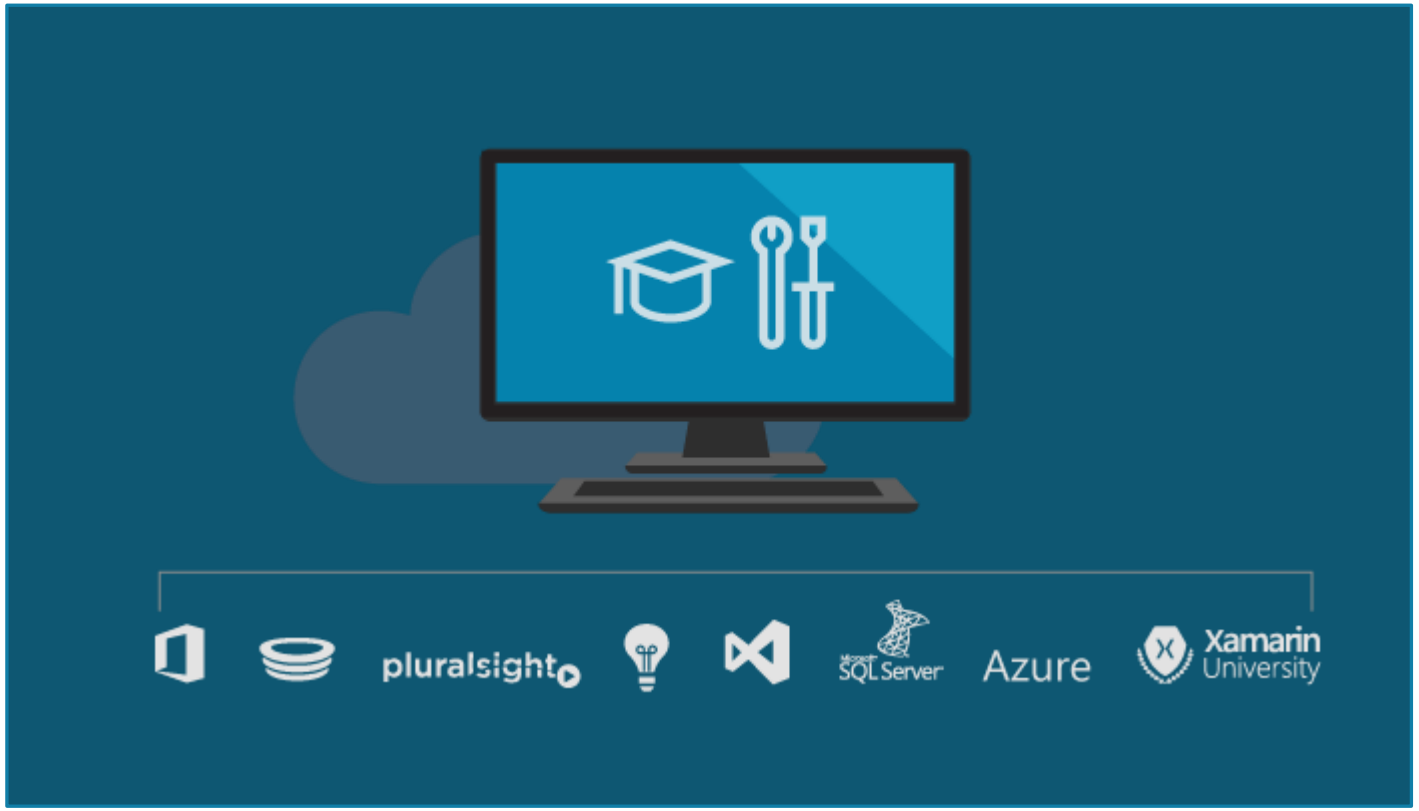


Visual Studio Code

Code editing. Redefined.

http://www.visualstudio.com/

Visual Studio Dev Essentials



https://www.visualstudio.com/dev-essentials/

Uczymy się, uczymy…

O'REILLY®
Safari

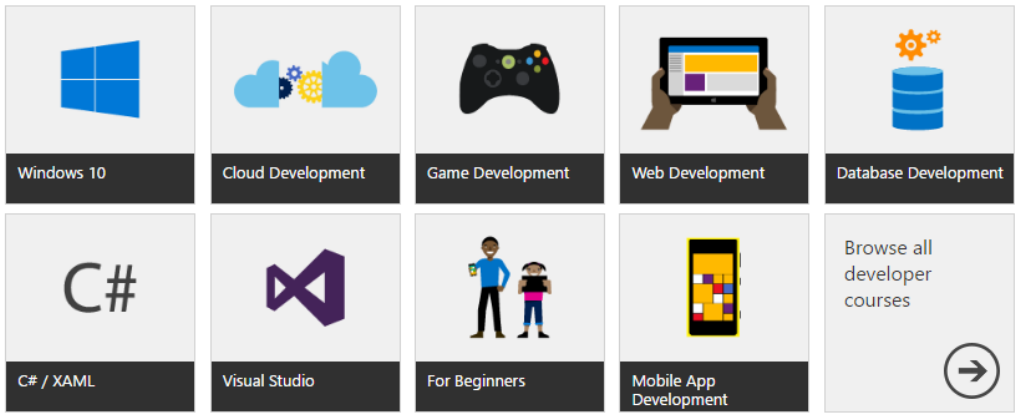
https://safaribooksonline.com

PLURALSIGHT

https://www.pluralsight.com/

MVA
Microsoft Virtual Academy

Free Microsoft training delivered by experts



https://mva.microsoft.com/

Kod, jeszcze więcej kodu…

https://bitbucket.org/

https://azure.microsoft.com/en-us/services/devops/

https://github.com/

Co potrzebujesz?

- Dostępu do literatury
- Komputer z zainstalowanymi środowiskami (Visual Studio 2022, jdk8 + Eclipse IDE)
- Konto na np.. **Azure DevOps** założone na adres poczty PWr
- Dobrych chęci, żeby się nauczyć czegoś nowego
- Trochę czasu, żeby przyjść na zajęcia
- Dużo czasu, żeby ćwiczyć indywidualnie

Git jest git

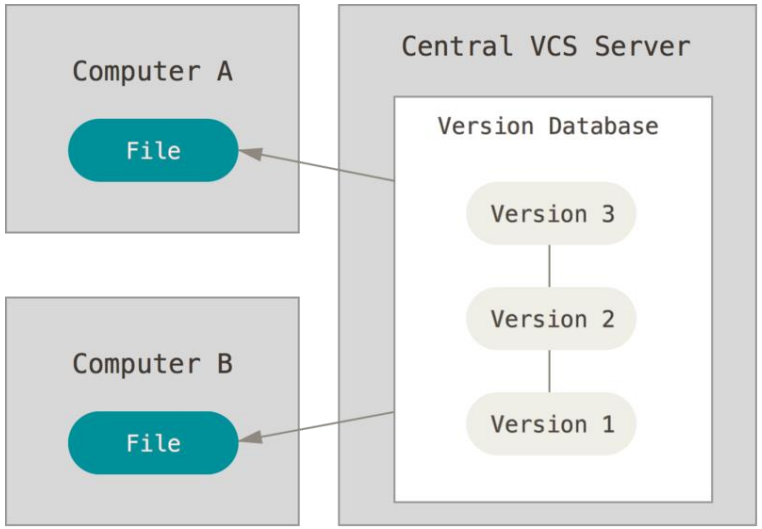
System kontroli wersji

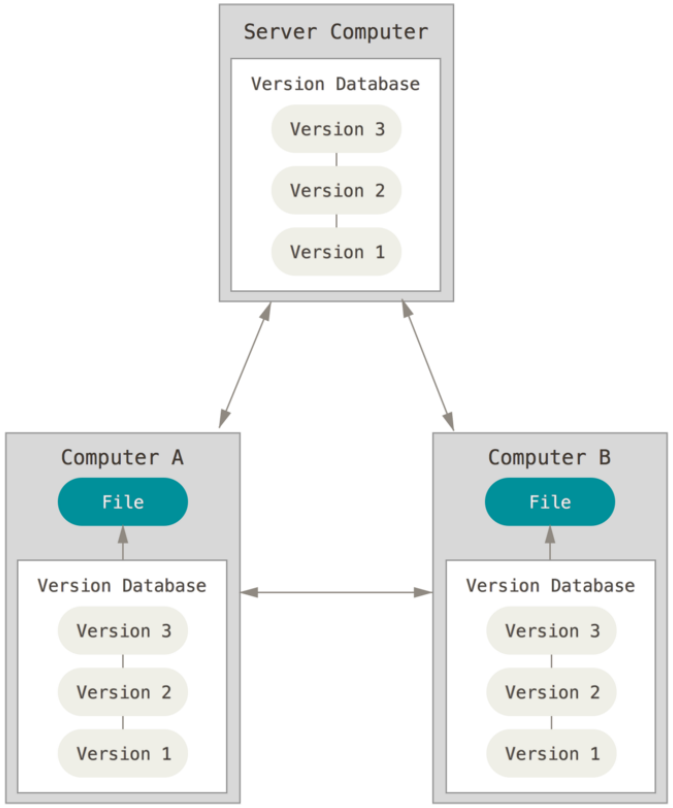
- Oprogramowanie służące do śledzenia zmian w plikach (w szczególności w kodzie źródłowym)
- Pomaga zachować i przeglądać wprowadzane zmiany (a także porównywać wersje między sobą)
- Wspomaga łączenie zmian wprowadzanych przez różne osoby w zespole (a także w przez różne zespoły)
- Przykładowe systemy kontroli wersji: GIT, Subversion (SVN), Team Foundation Server (TFS)

Instalacja i materiały

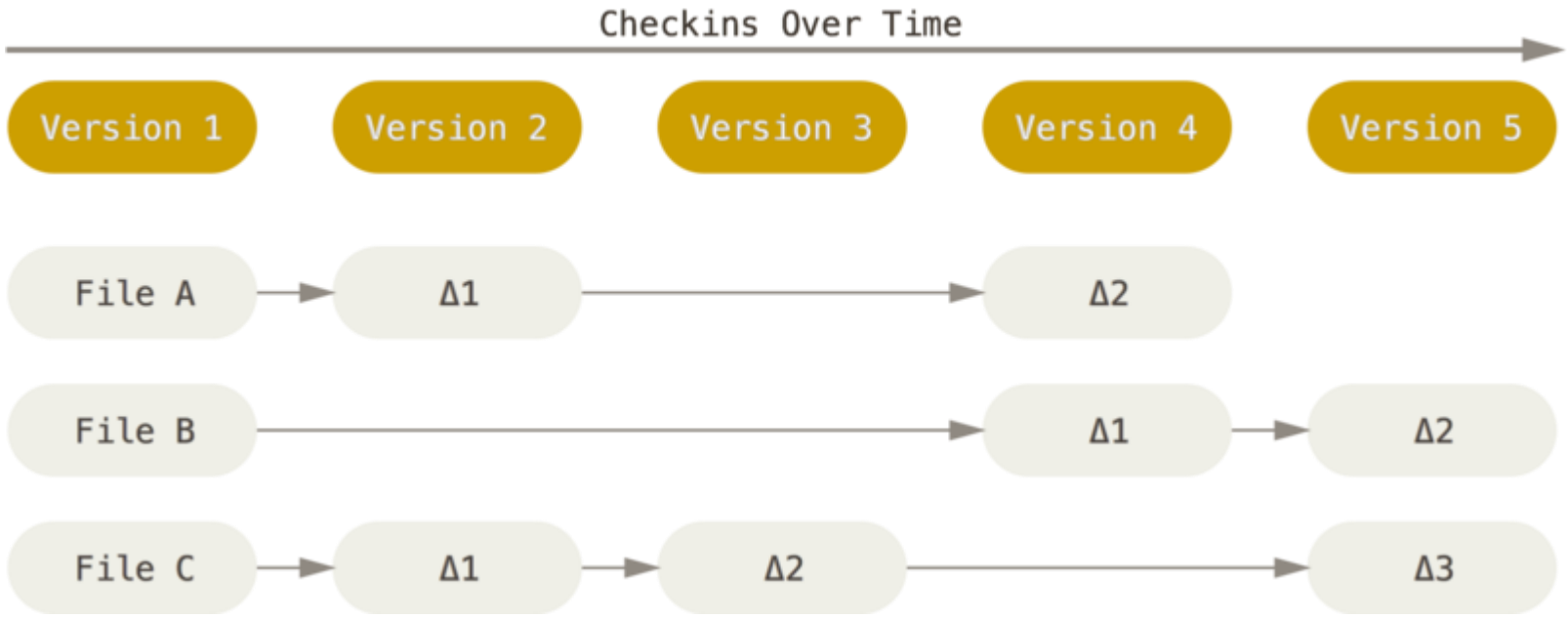
- Klient wbudowany w Visual Studio (i nie tylko) / SourceTree https://www.sourcetreeapp.com/
- Klient samodzielny: http://git-scm.com/downloads
- Bitbucket: https://bitbucket.org/
- Do poczytania: http://git-scm.com/book/en/v2
- Do poćwiczenia: http://pcottle.github.io/learnGitBranching/
- Do trzymania pod ręką: http://rogerdudler.github.io/git-guide/

Git – system rozproszony

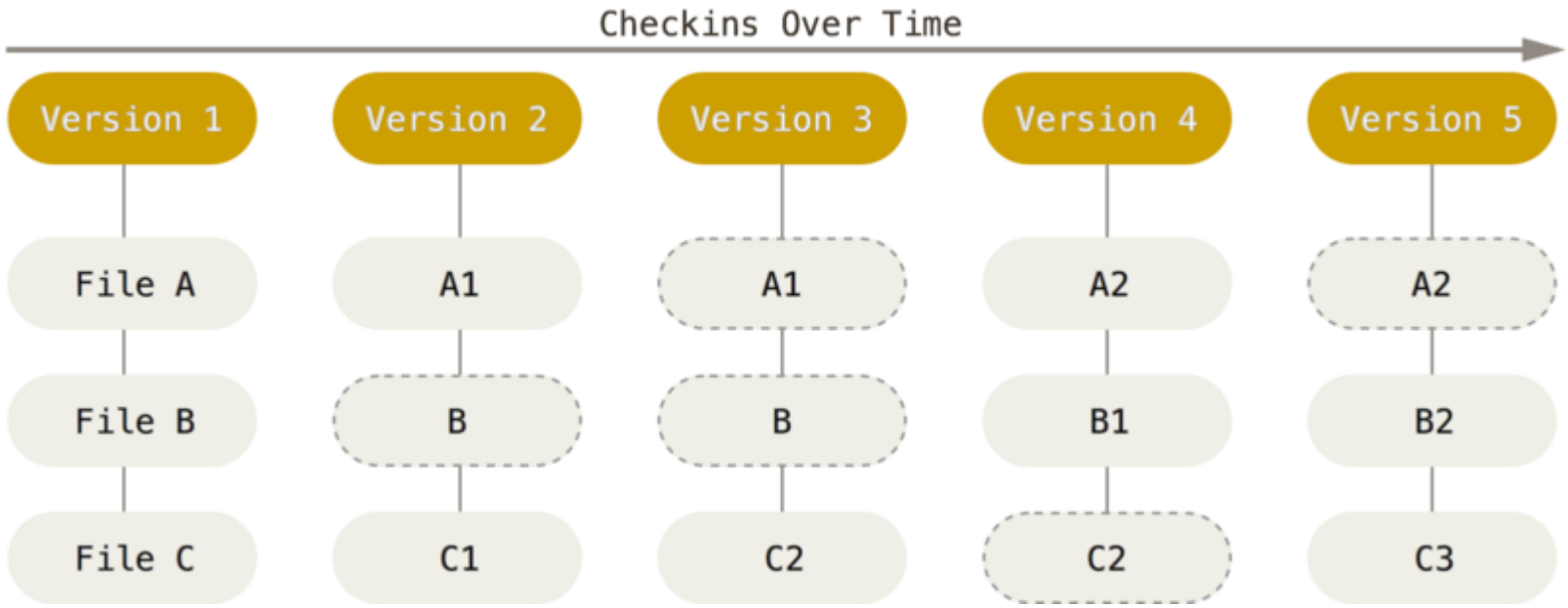




Zapisywanie zmian w dawnych czasach

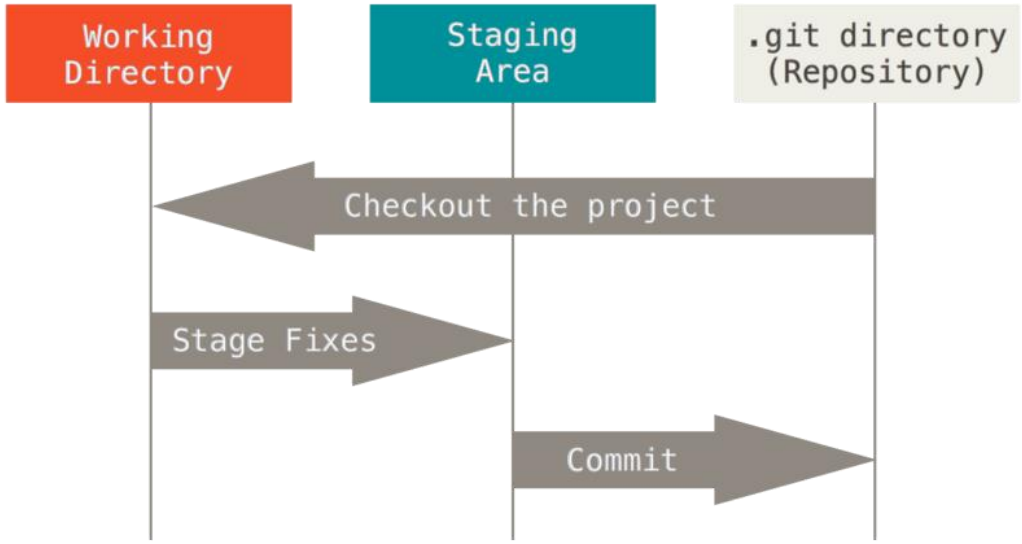


Historia zmian w Git

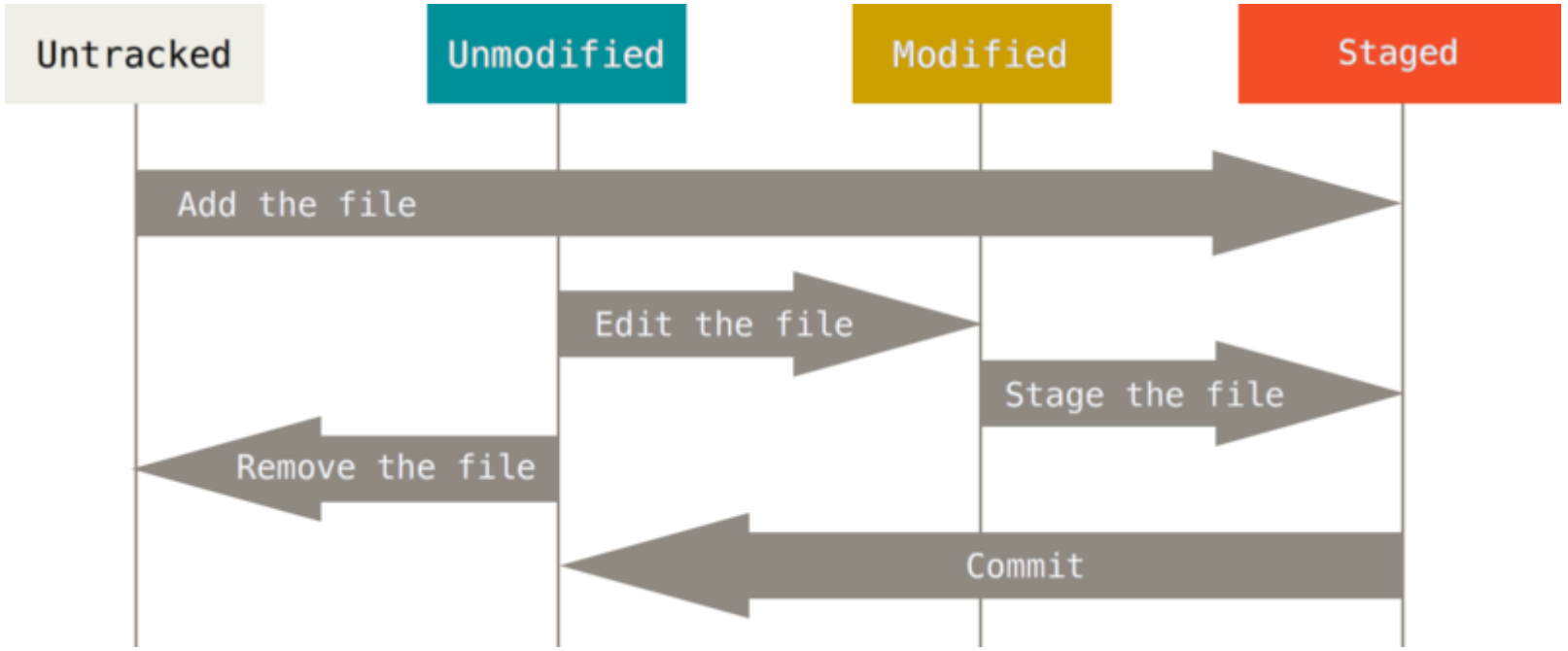


Git - charakterystyka

- Prawie każda operacja jest lokalna -> szybkość działania
- mechanizm oparty na sumach kontrolnych SHA-1 -> spójność
- Git w większości przypadków tylko dodaje dane
- 3 sekcje w Git-cie



Git – cykl życia pliku



Mega skrócony zestaw komend

- Inicjalizacja nowego repo

```
git init
```

- Klonowanie istniejącego

```
git clone [url]
```

- Aktualny status

```
git status
```

- Dodawanie zmian do stage

```
git add <filename>
```

- Zakomitowanie zmian

```
git commit –m ”opis”
```

- Wrzucenie zmian na serwer

```
git push origin <nazwa>
```

- Stworzenie branch-a

```
git branch <nazwa>
```

- Przełączanie branch-a

```
git checkout <nazwa>
```