Programowanie aplikacji w Java

Maciej Gowin

Zjazd 4 - dzień 2

Linki

Opis

https://maciejgowin.github.io/wsb-java/

Kod źródłowy przykładów oraz zadań

https://github.com/MaciejGowin/wsb-programowanie-aplikacji-java

Język Java: pliki jar

Dystrybucja skompilowanego kodu w postaci wielu plików `.class` byłaby uciążliwa. W Java możliwe jest archiwizowanie plików w formacie opartym na formacie `zip`. Pozwala to na łatwe współdzielenie kodu.

Java wprowadza format `jar` (ang. Java archive).

Język Java: pliki jar

Do stworzenia pliku `jar` używamy polecenia:

```
jar --create --file my-jar-name.jar Main.class Another1.class Another2.class ...
```

```
jar cf my-jar-name.jar Main.class Another1.class Another2.class ...
```

gdzie:

- `--create` opcja tworzenia archiwum
- `--file` specyfikacja nazwy pliku
- `my-jar-name.jar` nazwa archiwum
- `Main.class Another1.class Another2.class ...` pliki do archiwizacji

Język Java: pliki jar

Po utworzeniu pliku `jar` możemy uruchomić program, używając polecenia:

```
java -cp my-jar-name.jar Main
```

Istotna jest definicja `-cp` (lub `-classpath`). W przypadku braku definicji tej opcji Java domyślnie ustawia ścieżkę przeszukiwań na obecny katalog `.`. W związku z tym, że nasze klasy umieszczone zostały w archiwum, musimy poinstruować JVM do jego przeszukania.

Dodatkowo `Main` to klasa posiadającą metodę `main`. Używamy jej w analogiczny sposób jak w przypadku uruchomienia z domyślną ścieżką przeszukiwania.

Język Java: pliki jar

Po rozpakowaniu archiwum możemy podejrzeć jego zawartość:

`jar --extract --file main.jar`

`jar xf main.jar`

```
META-INF
|- MANIFEST.MF
Main.class
Another1.class
Another2.class
...
```

Java automatycznie dodaje plik `MANIFEST.MF`, który posiada rozszerzone opisy oraz instrukcje związane z jego zachowaniem. W podstawowej formie:

```
Manifest-Version: 1.0
Created-By: 11.0.8 (N/A)
```

Język Java: pliki jar

Jeżeli podczas budowania pliku zdefiniujemy klasę startową możemy przekształcić plik `jar` w plik uruchomieniowy.

```
jar --create --file my-jar-name.jar --main-class Main Main.class Another1.class Another2.class ...
```

```
jar --create --file my-jar-name.jar --manifest MANIFEST.MF Another1.class Another2.class ...
```

Pozwoli to na uruchomienie programu bez zadawania klasy z metodą `main`.

```
java -jar my-jar-name.jar
```

Język Java: fat jar

W przypadku, w którym nasz kod używa zewnętrznych bibliotek, możemy je dodać do ścieżki wyszukiwań lub też umieścić je w wynikowym archiwum.

W taki sposób powstaje tzw. `uber jar` lub `fat jar` zawierający wszystkie zależności.

Programowanie: przykład 48

> LineCounter.java

```
public class LineCounter {

    public static void main(String[] args) {
        if (args.length != 1) {
            System.out.println("Specify filename");
            return;
        }

        FileUtils.countLines(args[0]).ifPresent(count -> System.out.printf("Lines: %d%n", count));
    }
}
```

Programowanie: przykład 48

> FileUtils.java

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.util.Optional;

import static java.util.Optional.empty;
import static java.util.Optional.of;

public class FileUtils {

    public static Optional<Long> countLines(final String filename) {
        try {
            return of(Files.readAllLines(Path.of(filename)).stream().filter(FileUtils::isNotBlank).count());
        } catch (IOException ex) {
            System.out.println("Failed to load file: " + ex.getMessage());
            return empty();
        }
    }

    public static boolean isNotBlank(final String value) {
        return value != null && value.replaceAll("\\s","").length() > 0;
    }
}
```

Programowanie: przykład 48

Compile

```
javac LineCounter.java
```

Create jar

```
jar --create --file line-counter.jar LineCounter.class FileUtils.class
```

```
jar cf line-counter.jar LineCounter.class FileUtils.class
```

Run with jar-based classpath

```
java -cp line-counter.jar LineCounter
```

Programowanie: przykład 48

Run as jar

```
java -jar line-counter.jar
```

> no main manifest attribute, in test.jar

Create Jar with main class definition

```
jar --create --file line-counter.jar --manifest MANIFEST.MF
LineCounter.class FileUtils.class
```

```
jar --create --file line-counter.jar --main-class LineCounter
LineCounter.class FileUtils.class
```

Run as jar

```
java -jar line-counter.jar
```

Budowanie aplikacji: automatyzacja procesu

Proces budowania może być uciążliwy. W szczególności, gdy nasze oprogramowanie zależne jest od wielu zewnętrznych bibliotek oraz wymagane są dodatkowe kroki niekoniecznie związane z samą kompilacją.

Dostępny jest szereg narzędzi, które automatyzują ten proces:

- Apache Ant
- Apache Maven
- Gradle

Budowanie aplikacji: Apache Ant

Korzysta z opisu procesu oraz kroków zdefiniowanych za pomocą pliku w formacie XML (o domyślnej nazwie `build.xml`). Wywołanie konkretnych kroków odbywa się przy pomocy polecenia `ant`, które operuje na zdefiniowanym wcześniej pliku.

```
ant compile
ant jar
ant run
```

Głównym narzędziem jest program konsolowy (ang. CLI, command-line interface) o nazwie `ant`.

Narzędzie `ant` dostępne jest pod adresem https://ant.apache.org/bindownload.cgi.

Budowanie aplikacji: Apache Ant

> build.xml

```
<?xml version="1.0"?>
<project name="Main" default="compile">
  <target name="clean" description="Deletes previous compile files">
    <delete dir="classes"/>
  </target>
  <target name="compile" description="Compiles application">
    <mkdir dir="classes" />
    <javac srcdir="." destdir="classes"/>
  </target>
  <target name="jar" depends="compile" description="Creates application jar file">
    <jar destfile="line-counter.jar">
      <fileset dir="classes" includes="**/*.class"/>
      <manifest>
        <attribute name="Main-Class" value="LineCounter"/>
      </manifest>
    </jar>
  </target>
</project>
```

Budowanie aplikacji: Apache Maven

Korzysta z opisu przy pomocy plików w formacie XML o nazwie POM (ang. Project Object Model). Opis skupia się na tym, co budujemy, a nie na tym, jak jest to wykonywane.

Budowanie odbywa się na podstawie **faz (ang. phase).** Fazy cyklu życia procesu budowania są z góry ustalone.

Głównym narzędziem jest program konsolowy (ang. CLI, command-line interface) o nazwie `mvn`.

Narzędzie `mvn` dostępne jest pod adresem https://maven.apache.org/download.cgi.

Apache Maven: fazy

Faza	Opis
clean	czyszczenie projektu z wcześniej wygenerowanych plików
validate	sprawdzenie czy wszystkie informacje potrzebne do procesu budowania są dostępne
compile	kompilacja kodu źródłowego
test-compile	kompilacja kodu źródłowego testów
test	wykonanie testów jednostkowych
package	budowanie paczki dystrybucyjnej (jar, war)

Apache Maven: fazy

Faza	Opis
integration-test	wykonanie testów integracyjnych
install	instalacja pakietu w lokalnym repozytorium
deploy	instalacja pakietu z zdalnym repozytorium

Apache Maven: fazy

Uruchomienie procesu następuje poprzez wybranie danej fazy.

```
mvn <phase>
mvn package
```

> Automatycznie wykonywane są wszystkie fazy poprzedzające.

Apache Maven: cele

Każda faza składa się z **celów (ang. goal),** które wykonywane są w kolejności. Cele są dostarczane poprzez mechanizm **wtyczek**, które je implementują. Każdy z celów musi definiować fazę, do której zostanie dodany.

Wtyczki mogą zostać dodane do procesu explicite lub też automatycznie. Maven dostarcza domyślną konfigurację celów na podstawie definicji `packaging`.

W najczęstszej konfiguracji możemy spotkać się z ustawieniem `packaging` na `jar`.

Apache Maven: cele

process-resources

resources:resources
org.apache.maven.plugins:maven-resources-plugin:2.6:resources

compile

compiler:compile
org.apache.maven.plugins:maven-compiler-plugin:2.5.1:compile

process-test-resources

resources:testResources
org.apache.maven.plugins:maven-resources-plugin:2.6:testResources

Apache Maven: cele

test-compile

compiler:testCompile
org.apache.maven.plugins:maven-compiler-plugin:2.5.1:testCompile

test

surefire:test
org.apache.maven.plugins:maven-surefire-plugin:2.12.4:test

package

jar:jar
org.apache.maven.plugins:maven-jar-plugin:2.4:jar

Apache Maven: cele

install

install:install
org.apache.maven.plugins:maven-install-plugin:2.4:install

deploy

deploy:deploy
org.apache.maven.plugins:maven-deploy-plugin:2.7:deploy

Apache Maven: wtyczki

Możemy zauważyć, że każda ze wspomnianych wtyczek zdefiniowana jest przez identyfikatory przedzielone `:`. Dla przykładu:

```
org.apache.maven.plugins : maven-deploy-plugin : 2.7
```

Gdzie:

`org.apache.maven.plugins` - definiuje grupę.
`maven-deploy-plugin` - definiuje artefakt.
`2.7` - definiuje wersję.

Apache Maven: wtyczki

Każda ze wtyczek może definiować więcej niż jeden cel niekoniecznie z tej samej fazy.

Istnieje możliwość wykonania danego celu z pominięciem fazy.

```
mvn <plugin>:<goal>
mvn compliler:compile
```

Apache Maven: główne fazy

Najczęściej będziemy używać polecenia łączącego 2 fazy:

```
mvn clean install
```

Pozwoli nam on na wyczyszczenie projektu z poprzednio wygenerowanych plików oraz pełne zbudowanie, połączone z uruchomieniem testów oraz instalacją w lokalnym repozytorium.

Programowanie: zadanie 25

Wygeneruj pierwszy projekt ze strukturą Maven. Użyj komendy (wpisanej w jednej linii):

```
mvn archetype:generate
    -DgroupId=pl.wsb.programowaniejava
    -DartifactId=example
    -DarchetypeArtifactId=maven-archetype-quickstart
    -DarchetypeVersion=1.4
    -DinteractiveMode=false
```

Przejdź do utworzonego katalogu `test` oraz uruchom proces:

```
cd example
mvn clean install
```

Uruchom program:

```
java -cp target/example-1.0-SNAPSHOT.jar pl.wsb.programowaniejava.App
```

Apache Maven: lokalne repozytorium .m2

Podczas budowania aplikacji na komputerze, na którym program Maven został użyty po raz pierwszy, zauważyć można wiele pobrań plików.

```
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-clean-plugin/3.1.0/maven-clean-plugin-3.1.0.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-clean-plugin/3.1.0/maven-clean-plugin-3.1.0.jar (30 kB at 244 kB/s)
```

Maven automatycznie pobiera zależności użyte w projekcie oraz instaluje je w lokalnych repozytorium artefaktów. Mieści się ono w katalogu `.m2/repository` w katalogu użytkownika. Pobieranie nowych artefaktów odbywa się jednorazowo.

Apache Maven: lokalne repozytorium .m2

Katalog `.m2` nie tylko przechowuje lokalne repozytorium. Znajdują się w nim też pliki konfiguracyjne Maven, w tym informacje o lokalizacji repozytoriów zewnętrznych, z których pobierane są zależności.

> Lokalizacja tego katalogu równi się od platformy.

Platforma	Domyślna lokalizacja
Windows	`C:\Users\<NAZWA_UZYTKOWNIKA>\.m2`
Linux	`/home/<NAZWA_UZYTKOWNIKA>/.m2`
Mac	`/Users/<NAZWA_UZYTKOWNIKA>/.m2`

Apache Maven: lokalne repozytorium .m2

Podczas budowania poinstruowaliśmy Maven, aby zainstalował nasz artefakt aplikacji w lokalnym repozytorium. W wyniku tego wynikowy plik `jar` zostanie dodany do katalogu `.m2/repository`.

```
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ test ---
[INFO] Installing /Users/gowinm/dev/external-projects/wsb-programowanie-aplikacji-java/0-zadanie-tworzenie-maven/test/target/test-1.0-SNAPSHOT.jar to /Users/gowinm/.m2/repository/pl/wsb/programowaniejava/test/1.0-SNAPSHOT/test-1.0-SNAPSHOT.jar
```

Artefakt został przekopiowany pod ścieżkę:

```
Users/gowinm/.m2/repository/pl/wsb/programowaniejava/test/1.0-SNAPSHOT/test-1.0-SNAPSHOT.jar
```

Gdzie:

- `pl/wsb/programowaniejava` to część ścieżki powstała z identyfikatora grupy (`groupId`),
- `test` to część ścieżki powstała z identyfikatora artefaktu (`artefactId`),
- `1.0-SNAPSHOT` to część ścieżki powstała z wersji artefaktu (`version`).

Apache Maven: struktura

Maven definiuje podstawową strukturę projektu.

```
test
|-- pom.xml
|-- src
|   |-- main
|   |   `-- java
|   |       `-- pl
|   |           `-- wsb
|   |               `-- programowaniejava
|   |                   `-- App.java
|   `-- test
|       `-- java
|           `-- pl
|               `-- wsb
|                   `-- programowaniejava
|                       `-- AppTest.java
`-- target
```

Apache Maven: struktura

Maven definiuje podstawową strukturę projektu.

- `pom.xml` zawiera definicję projektu oraz procesu budowania.
- `src/main/java` zawiera pliki z kodem źródłowym aplikacji.
- `src/test/java` zawiera pliki kodem źródłowym testów aplikacji.
- `target` zawiera pliki wygenerowane w procesie budowania.

Apache Maven: pom.xml

Informacje o projekcie.

```
<project>
  <groupId>pl.wsb.programowaniejava</groupId>
  <artifactId>test</artifactId>
  <version>1.0-SNAPSHOT</version>

  <name>test</name>
  <url>http://www.example.com</url>
</project>
```

Istotnymi elementami definiującymi projekt są identyfikator grupy (`groupId`), identyfikator artefaktu (`artifactId`) oraz wersja (`version`).

> Element `packaging` nie został zdefiniowany przez co przyjął domyślną wartość `jar` .

Apache Maven: identyfikatory

Identyfikator	Znaczenie
groupId	Unikatowy identyfikator dla projektu. Powinien nawiązywać do konwencji nazewniczych związanych z pakietami Java. Zalecane jest używanie odwróconych nazw domen np. `pl.wsb`.
artifactId	Nazwa wygenerowanego pliku Jar bez rozszerzenia. Nazwa dla konkretnego elementu, będącego częścią projektu.
version	Wersja przeważnie w formacie `x.x.x`, np. `0.1.2`. Dla projektów w fazie rozwojowym używamy rozszerzenia `-SNAPSHOT`, pozwalającego na definicję wersji tymczasowych.

Apache Maven: zmienne

Informacje o zmiennych dla naszego projektu. Element definiuje zmienne specyficzne dla projektu oraz pozwala na ustawienie wartości dla zmiennych oczekiwanych przez wtyczki.

```
<project>
  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.source>1.7</maven.compiler.source>
    <maven.compiler.target>1.7</maven.compiler.target>
  </properties>
</project>
```

> Zmienna `maven.compiler.source` jest zdefiniowana przez wtyczkę kompilatora oraz opisuje wersję Java.

Apache Maven: budowanie i wtyczki

Informacje o procesie budowania oraz wtyczkach użytych do jego wykonania.

```
<project>
  <build>
    <pluginManagement>
      <plugins>
        <plugin>
          <artifactId>maven-clean-plugin</artifactId>
          <version>3.1.0</version>
        </plugin>
      </plugins>
    </pluginManagement>
  </build>
</project>
```

Element `pluginManagement` służy do specyfikacji używanych wtyczek bez konieczności ich ładowania. W naszym przypadku Maven użył domyślnej konfiguracji dla `packaging` ustawionego na `jar` co spowodowało załadowanie `maven-clean-plugin`. Nadpisujemy jedynie wersję wtyczki.

Apache Maven: budowanie i wtyczki

Aby załadować wtyczkę bezpośrednio, użyjemy elementu `plugins` będącego dzieckiem `build`.

```
<project>
  <build>
    <pluginManagement>
      <plugins>
      </plugins>
    </pluginManagement>
    <plugins>
    </plugins>
  </build>
</project>
```

Apache Maven: zależności

Informacje o zależnościach projektu. Definiuje biblioteki używane przez kod źródłowy programu oraz kod źródłowy testów.

```
<project>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.11</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

> Dla zależności definiujemy również element `dependencyManagement`, którego rola jest podobna do tego znanego z `pluginManagement`.

Programowanie: przykład 49

Tworzenie projektu LineCounter przy pomocy Maven.

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>pl.wsb.programowaniejava</groupId>
    <artifactId>line-counter</artifactId>
    <version>1.0-SNAPSHOT</version>

    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>11</maven.compiler.source>
        <maven.compiler.target>11</maven.compiler.target>
    </properties>
</project>
```

Programowanie: przykład 49

Uruchomienie programu.

```
java -cp target/line-counter-1.0-SNAPSHOT.jar pl.wsb.programowaniejava.LineCounter sample
```

Programowanie: przykład 50

Użycie zewnętrznej biblioteki StringUtils z Apache Commons Lang.

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>pl.wsb.programowaniejava</groupId>
    <artifactId>line-counter-extended</artifactId>
    <version>1.0-SNAPSHOT</version>

    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>11</maven.compiler.source>
        <maven.compiler.target>11</maven.compiler.target>
    </properties>

    <dependencies>
        <dependency>
            <groupId>org.apache.commons</groupId>
            <artifactId>commons-lang3</artifactId>
            <version>3.12.0</version>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-assembly-plugin</artifactId>
                <version>3.1.1</version>

                <configuration>
                    <descriptorRefs>
                        <descriptorRef>jar-with-dependencies</descriptorRef>
                    </descriptorRefs>
                </configuration>

                <executions>
                    <execution>
                        <id>make-assembly</id>
                        <phase>package</phase>
                        <goals>
                            <goal>single</goal>
                        </goals>
                    </execution>
                </executions>
            </plugin>
        </plugins>
    </build>
</project>
```

Programowanie: przykład 50

Uruchomienie programu.

```
java -cp target/line-counter-extended-1.0-SNAPSHOT.jar pl.wsb.programowaniejava.LineCounter sample
```

```
mvn exec:java -Dexec.mainClass=pl.wsb.programowaniejava.LineCounter -Dexec.args=sample
```

```
java -cp target/line-counter-extended-1.0-SNAPSHOT-jar-with-dependencies.jar pl.wsb.programowaniejava.LineCounter sample
```

Budowanie aplikacji: Apache Maven Central repository

Wszystkie artefakty są domyślnie ściągane z naszego lokalnego repozytorium. Jeżeli artefakt nie jest dostępny, Maven będzie starał się go pobrać z repozytorium zewnętrznego.

Domyślnie Maven używa repozytorium zewnętrznego dostępnego pod adresem: `https://repo.maven.apache.org/maven2/.`

W celu łatwiejszego wyszukiwania artefaktów możemy posłużyć się stroną: `https://mvnrepository.com/`

Istnieje możliwość zdefiniowania własnych repozytoriów zewnętrznych np. wspólnych dla danej firmy. Możliwe jest to z poziomu `pom.xml` lub też `.m2/settings.xml`

Kontrola wersji

Podczas prac nad oprogramowaniem kod źródłowy ulega ciągłym zmianom. Dodatkowo proces tworzenia oprogramowania nie jest liniowy. Często nad jego rozwojem pracuje więcej niż jedna osoba.

Aby usprawnić zarządzanie zmianami, wprowadzamy systemy kontroli wersji wspierające procesy tworzenia.

Dostępnych jest szereg systemów, takich jak:

- CVS
- SVN
- Mercurial
- Git

Git: charakterystyka

Git jest rozproszonym systemem kontroli wersji. Może zostać scharakteryzowany przez swoje główne cechy.

Silne wsparcie dla nieliniowego procesu tworzenia oprogramowania

Git wspiera model tworzenia rozgałęzień oraz ich łączenia. Każdy programista może pracować nad własnymi lokalnymi zmianami, a następnie w łatwy sposób łączyć je ze zmianami innych.

Rozproszone tworzenie oprogramowania

Każdy programista posiada własną kopię repozytorium pozwalającą na pracę offline. Kopia ta zawiera pełną historię zmian. Lokalne repozytoria mogą wymieniać zmiany pomiędzy sobą.

Git: charakterystyka

Kompatybilność z istniejącymi protokołami

Git używa istniejących protokołów do wymiany danych takich jak: HTTP oraz FTP.

Wydajna obsługa dużych projektów

Głównym założeniem jest wydajność dla dużych projektów.

Git: charakterystyka

Autentykacja historii zmian

Dla zmian opublikowanych jakiekolwiek modyfikacje pozostawiają ślady, które w jasny sposób pokazują niepożądane zachowanie.

System narzędziowy

Git składa się z zestawu narzędzi, które pozwalają na zarządzanie wersją.

Git: charakterystyka

Strategie łączenia zmian

Git zawiera szereg algorytmów odpowiedzialnych za łączenie zmian. W przypadku konfliktów, gdy automatyczne łączenie zmian nie jest możliwe, wymagana jest ręczna finalizacja procesu.

Przechowywanie pełnego obrazu zmian

Git nie zapamiętuje zmian między kolejnymi rewizjami, lecz kompletne obrazy.

Git: narzędzie

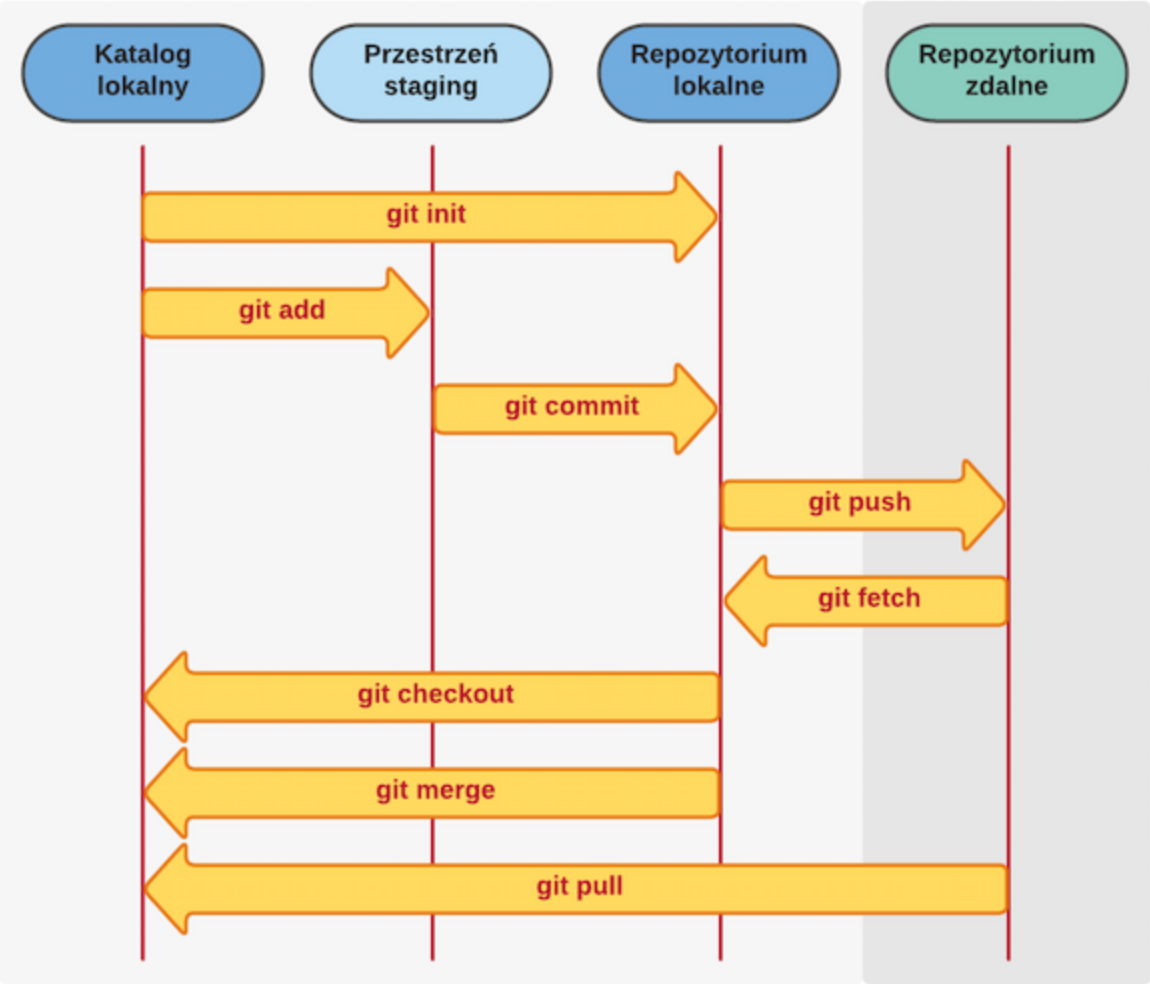
Praca z Git opera się na wykonywaniu komend, które pozwalają na zarządzanie zmianami, ich dodawanie, łączenie oraz publikację.

Głównym narzędziem jest program konsolowy (ang. CLI, command-line interface) o nazwie `git`, aczkolwiek istnieją też wersje graficzne. Niektóre z nich dostarczane z IDE.

Rozwiązanie konsolowe pozwala na łatwiejsze zrozumienie nazewnictwa oraz podstawowych koncepcji jego działania.

Narzędzie `git` dostępne jest pod adresem https://git-scm.com/downloads.

Git: podstawowy przepływ



Git: pojęcia

Katalog lokalny/obszar roboczy

Katalog, zawierający pliki, na których dokonujemy zmian.

Przestrzeń staging

Obszar, do którego dodawane są zmiany z obszaru roboczego, które zostaną dodane do repozytorium w postaci rewizji. Niejako bufor zmian.

Git: pojęcia

Repozytorium

Przestrzeń, w której znajdują się wszystkie pliki, pełna historia zmian oraz inne dane związane z `git`. Definiujemy repozytorium lokalne (znajdujące się na danej maszynie roboczej) oraz zdalne (zewnętrzne względem danej maszyny roboczej).

Rewizja

Rewizja (ang. commit) to pojedynczy wpis w repozytorium zawierający wszystkie pliki wraz z konkretnymi zamianami. Posiada przypisany unikatowy identyfikator.

Git: pojęcia

Gałąź

Gałąź (ang. branch) oddzielna przestrzeń, w której znajduje się kod posiadająca swój własny stan katalogu roboczego, przestrzeni staging oraz historię rewizji. Każde repozytorium posiada przynajmniej jedną gałąź. Domyślna gałąź to `master`.

> Niektóre systemy definiują domyślną gałąź jako `main`.

Scalanie

Scalanie (ang. merge) to proces łączenia zmian z dwóch różnych gałęzi. Najprostszą jego formą jest typ `Fast Forward`, w którym zmiany są liniowe.

Git: pojęcia

HEAD

Wskaźnik na ostatnią rewizję aktywnej gałęzi. Innymi słowy, to miejsce, w którym znajdujemy się w danym momencie.

origin

Domyślna nazwa dla repozytorium zewnętrznego.

Git: przypadki użycia

Tworzenie katalogu przechowującego kod źródłowy projektu.

```
gowinm:~> mkdir line-counter
```

Inicjalizacja lokalnego repozytorium dla projektu.

```
gowinm:~> cd line-counter
gowinm:~/line-counter> git init
Initialized empty Git repository in /Users/gowinm/line-counter/.git/
```

Git: katalog .git

Git automatycznie utworzył katalog `.git` przechowujący historię zmian lokalnego repozytorium oraz dane konfiguracyjne.

Dane konfiguracyjne repozytorium umieszczone są w pliku `.git/config`.

```
[core]
        repositoryformatversion = 0
        filemode = true
        bare = false
        logallrefupdates = true
        ignorecase = true
        precomposeunicode = true
```

Git: plik .gitconfig

Globalna konfiguracja jest definiowana w pliku `.gitconfig` w katalogu domowym użytkownika. Jego lokalizacja jest zależna od platformy.

```
[user]
        name = Maciej Gowin
        email = gowinm@ryanair.com
[core]
        autocrlf = false
[push]
        default = simple
```

Git: przypadki użycia

Sprawdzenie zmian w lokalnym repozytorium.

```
gowinm:~/line-counter> git log
fatal: your current branch 'master' does not have any commits yet
```

Git: przypadki użycia

Sprawdzenie zmian w przestrzeni staging.

```
gowinm:~/line-counter> git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
```

Git: przypadki użycia

Nowy, pusty plik `LineCounter.java`. Sprawdzenie zmian w przestrzeni staging.

```
gowinm:~/line-counter> touch LineCounter.java
```

```
gowinm:~/line-counter> git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        LineCounter.java

nothing added to commit but untracked files present (use "git add" to track)
```

Git: przypadki użycia

Dodanie pliku do przestrzeni staging.

```
gowinm:~/line-counter> git add LineCounter.java
```

```
gowinm:~/line-counter> git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   LineCounter.java
```

Git: przypadki użycia

Utworzenie pierwszej rewizji na podstawie zmian w przestrzeni staging.

```
gowinm:~/line-counter> git commit -m "First change"
[master (root-commit) 2a675e4] First change
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 LineCounter.java
```

```
gowinm:~/line-counter> git status
On branch master
nothing to commit, working tree clean
```

Git: przypadki użycia

Historia zmian w repozytorium.

```
gowinm:~/line-counter> git log
commit 2a675e48ead1ef2bd8bc538a2749832544be7640 (HEAD -> master)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:21:50 2021 +0100

    First change
```

Git: przypadki użycia

Zmiana zawartości pliku `LineCounter.java` oraz dodanie nowego, pustego pliku `sample`.

```
gowinm:~/line-counter> vi LineCounter.java
gowinm:~/line-counter> touch sample
```

> LineCounter.java

```
public class LineCounter {
    public static void main(String[] args) {
        System.out.println("Line Counter");
    }
}
```

Git: przypadki użycia

Sprawdzenie zmian.

```
gowinm:~/line-counter> git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   LineCounter.java

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        sample

no changes added to commit (use "git add" and/or "git commit -a")
```

Git: przypadki użycia

Dodanie zmian do przestrzeni staging oraz sprawdzenie jej zawartości. Utworzenie nowej rewizji.

```
gowinm:~/line-counter> git add LineCounter.java
gowinm:~/line-counter> git add sample
```

```
gowinm:~/line-counter> git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   LineCounter.java
        new file:   sample
```

```
gowinm:~/line-counter> git commit -m "Second change"
[master e800c8e] Second change
 2 files changed, 5 insertions(+)
 create mode 100644 sample
```

Git: przypadki użycia

Zmiana zawartości pliku oraz sprawdzenie statusu zmian.

```
gowinm:~/line-counter> vi sample
```

```
gowinm:~/line-counter> git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   sample

no changes added to commit (use "git add" and/or "git commit -a")
```

Git: przypadki użycia

Sprawdzenie zmian.

```
gowinm:~/line-counter> git diff
diff --git a/sample b/sample
index e69de29..d95f3ad 100644
--- a/sample
+++ b/sample
@@ -0,0 +1 @@
+content
```

Git: przypadki użycia

Dodanie pliku do przestrzeni staging oraz automatyczne utworzenie rewizji.

```
gowinm:~/line-counter> git commit -am "Third change"
[master cf92ad8] Third change
 1 file changed, 1 insertion(+)
```

Git: przypadki użycia

Przegląd rewizji.

```
gowinm:~/line-counter> git log
commit cf92ad802d57c2a7f62610c4ea2157af729d4a81 (HEAD -> master)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:37:01 2021 +0100

    Third change

commit e800c8e0bf4ba823e2a6da0103d197554e160deb
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:32:52 2021 +0100

    Second change

commit 2a675e48ead1ef2bd8bc538a2749832544be7640
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:21:50 2021 +0100

    First change
```

Git: przypadki użycia

Sprawdzenie dostępnych gałęzi oraz utworzenie nowej gałęzi `feature`.

```
gowinm:~/line-counter> git branch
* master
```

```
gowinm:~/line-counter> git branch feature
```

```
gowinm:~/line-counter> git branch
  feature
* master
```

Git: przypadki użycia

Sprawdzenie rewizji.

```
gowinm:~/line-counter> git log
commit cf92ad802d57c2a7f62610c4ea2157af729d4a81 (HEAD -> master, feature)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:37:01 2021 +0100

    Third change

commit e800c8e0bf4ba823e2a6da0103d197554e160deb
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:32:52 2021 +0100

    Second change

commit 2a675e48ead1ef2bd8bc538a2749832544be7640
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:21:50 2021 +0100

    First change
```

Git: przypadki użycia

Przełączenie się na gałąź `feature` oraz sprawdzenie aktywnej gałęzi.

```
gowinm:~/line-counter> git checkout feature
Switched to branch 'feature'
```

```
gowinm:~/line-counter> git branch
* feature
  master
```

Git: przypadki użycia

Sprawdzenie rewizji.

```
gowinm:~/line-counter> git log
commit cf92ad802d57c2a7f62610c4ea2157af729d4a81 (HEAD -> feature, master)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:37:01 2021 +0100

    Third change

commit e800c8e0bf4ba823e2a6da0103d197554e160deb
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:32:52 2021 +0100

    Second change

commit 2a675e48ead1ef2bd8bc538a2749832544be7640
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:21:50 2021 +0100

    First change
```

Git: przypadki użycia

Dodanie zmiany na gałęzi `feature`.

```
gowinm:~/line-counter> vi sample
```

```
gowinm:~/line-counter> git commit -am "Feature first change"
[feature ef14f54] Feature first change
 1 file changed, 1 insertion(+)
```

Git: przypadki użycia

Sprawdzenie ostatnich 3 rewizji.

```
gowinm:~/line-counter> git log -n 3
commit ef14f5478fb1c5b26dbcda355c0b48214964e57 (HEAD -> feature)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 22:19:59 2021 +0100

    Feature first change

commit cf92ad802d57c2a7f62610c4ea2157af729d4a81 (master)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:37:01 2021 +0100

    Third change

commit e800c8e0bf4ba823e2a6da0103d197554e160deb
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:32:52 2021 +0100

    Second change
```

Git: przypadki użycia

Przełączenie się na gałąź `master` oraz sprawdzenie ostatnich 2 rewizji.

```
gowinm:~/line-counter> git checkout master
Switched to branch 'master'
```

```
gowinm:~/line-counter> git log -n 2
commit cf92ad802d57c2a7f62610c4ea2157af729d4a81 (HEAD -> master)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:37:01 2021 +0100

    Third change

commit e800c8e0bf4ba823e2a6da0103d197554e160deb
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:32:52 2021 +0100

    Second change
```

Git: przypadki użycia

Scalenie zmian z gałęzi `feature` do gałęzi `master`. Sprawdzenie stanu rewizji.

```
gowinm:~/line-counter> git merge feature
Updating cf92ad8..ef14f54
Fast-forward
 sample | 1 +
 1 file changed, 1 insertion(+)
```

```
gowinm:~/line-counter> git log -n 2
commit ef14f5478fb1c5b26dbcda355c0b48214964e57 (HEAD -> master, feature)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 22:19:59 2021 +0100

    Feature first change

commit cf92ad802d57c2a7f62610c4ea2157af729d4a81
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 21:37:01 2021 +0100

    Third change
```

Git: przypadki użycia

Dodanie zmian do pliku `sample` na gałęzi `master`.

```
gowinm:~/line-counter> vi sample
```

```
gowinm:~/line-counter> git commit -am "Fourth change"
[master 75e6de4] Fourth change
 1 file changed, 1 insertion(+)
gowinm:~/line-counter> git log -n 2
commit 75e6de4963baf1de596b119a6d23f9f38a958865 (HEAD -> master)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 22:46:51 2021 +0100

    Fourth change

commit ef14f5478fb1c5b26dbcda355c0b482149664e57 (feature)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 22:19:59 2021 +0100

    Feature first change
```

Git: przypadki użycia

Dodanie zmian do pliku `LineCounter.java` na gałęzi `feature`.

```
gowinm:~/line-counter> git checkout feature
Switched to branch 'feature'
```

```
gowinm:~/line-counter> git branch
* feature
  master
```

```
gowinm:~/line-counter> vi LineCounter.java
```

```
gowinm:~/line-counter> git commit -am "Feature second change"
[feature f7d651d] Feature second change
 1 file changed, 5 insertions(+)
```

Git: przypadki użycia

Sprawdzenie rewizji na gałęzi `feature`.

```
gowinm:~/line-counter> git log -n 2
commit f7d651d66d1565a9e2541fee7de18eeb3bea20fa (HEAD -> feature)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:00:52 2021 +0100

    Feature second change

commit ef14f5478fb1c5b26dbcda355c0b482149664e57
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 22:19:59 2021 +0100

    Feature first change
```

Git: przypadki użycia

Przełączenie się na gałąź `master` oraz scalenie zmian z gałęzi `feature`.

```
gowinm:~/line-counter> git checkout master
Switched to branch 'master'
```

```
gowinm:~/line-counter> git merge feature
Merge made by the 'recursive' strategy.
 LineCounter.java | 5 +++++
 1 file changed, 5 insertions(+)
```

```
Merge branch 'feature'

# Please enter a commit message to explain why this merge is necessary,
# especially if it merges an updated upstream into a topic branch.
#
# Lines starting with '#' will be ignored, and an empty message aborts
# the commit.
```

Git: przypadki użycia

Sprawdzenie stanu rewizji gałęzi `master`.

```
gowinm:~/line-counter> git log -n 3
commit 3be9d76885d10da31292db9a6a860f23b969dd70 (HEAD -> master)
Merge: 75e6de4 f7d651d
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:03:37 2021 +0100

    Merge branch 'feature'

commit f7d651d66d1565a9e2541fee7de18eeb3bea20fa (feature)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:00:52 2021 +0100

    Feature second change

commit 75e6de4963baf1de596b119a6d23f9f38a958865
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 22:46:51 2021 +0100

    Fourth change
```

Git: GitHub

GitHub (https://github.com) jest platformą hostingową do rozwoju oprogramowania oraz kontroli wersji przy użyciu `git`. Oferuje funkcjonalności wbudowane w `git` oraz szereg własnych. Pozwala na tworzenie zdalnych repozytoriów przez użytkowników. Repozytoria te mogą być prywatne (widoczne przez zdefiniowaną grupę użytkowników) lub też publiczne (widoczne przez każdego).

Podobnym rozwiązaniem jest **Bitbucket** (https://bitbucket.org).

Istnieją też rozwiązania firmowe pozwalające na definiowanie własnych serwerów z platformami do rozwoju oprogramowania oraz tworzenia scentralizowanych repozytoriów zdalnych.

Git: przypadki użycia

Dodanie zdalnego repozytorium pod identyfikatorem `origin`. Połączenie lokalnej gałęzi ze zdalną gałęzią pod tą samą nazwą.

```
gowinm:~/line-counter> git remote add origin https://github.com/MaciejGowin/line-counter.git
```

```
gowinm:~/line-counter> git push -u origin master
Enumerating objects: 20, done.
Counting objects: 100% (20/20), done.
Delta compression using up to 8 threads
Compressing objects: 100% (15/15), done.
Writing objects: 100% (20/20), 1.88 KiB | 1.88 MiB/s, done.
Total 20 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/MaciejGowin/line-counter.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

Git: przypadki użycia

Po dodaniu zdalnego repozytorium `.git/config` zostało rozszerzone o:

```
[remote "origin"]
        url = https://github.com/MaciejGowin/line-counter.git
        fetch = +refs/heads/*:refs/remotes/origin/*
```

Po ustawieniu zdalnej gałęzi `.git/config` zostało rozszerzone o:

```
[branch "master"]
        remote = origin
        merge = refs/heads/master
```

Git: przypadki użycia

Sprawdzenie zdalnych gałęzi.

```
gowinm:~/line-counter> git branch -r
  origin/master
```

Git: przypadki użycia

Sprawdzenie stanu rewizji.

```
gowinm:~/line-counter> git log -n 3
commit 3be9d76885d10da31292db9a6a860f23b969dd70 (HEAD -> master, origin/master)
Merge: 75e6de4 f7d651d
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:03:37 2021 +0100

    Merge branch 'feature'

commit f7d651d66d1565a9e2541fee7de18eeb3bea20fa (feature)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:00:52 2021 +0100

    Feature second change

commit 75e6de4963baf1de596b119a6d23f9f38a958865
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 22:46:51 2021 +0100

    Fourth change
```

Git: przypadki użycia

Dodanie zmian do pliku `sample` oraz utworzenie nowej rewizji.

```
gowinm:~/line-counter> vi sample
```

```
gowinm:~/line-counter> git commit -am "Fifth change"
[master 1771ecf] Fifth change
 1 file changed, 1 insertion(+)
```

Git: przypadki użycia

Sprawdzenie rewizji.

```
gowinm:~/line-counter> git log -n 2
commit 1771ecf8489bf59cec32dad6e7a679b1633d90d3 (HEAD -> master)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:40:54 2021 +0100

    Fifth change

commit 3be9d76885d10da31292db9a6a860f23b969dd70 (origin/master)
Merge: 75e6de4 f7d651d
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:03:37 2021 +0100

    Merge branch 'feature'
```

Git: przypadki użycia

Wysłanie zmian do zdalnego repozytorium. Sprawdzenie stanu.

```
gowinm:~/line-counter> git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 313 bytes | 313.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/MaciejGowin/line-counter.git
   3be9d76..1771ecf  master -> master
```

```
gowinm:~/line-counter> git log -n 1
commit 1771ecf8489bf59cec32dad6e7a679b1633d90d3 (HEAD -> master, origin/master)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:40:54 2021 +0100

    Fifth change
```

Git: przypadki użycia

Imitacja pracy na drugim lokalnym repozytorium. Pobranie obecnego stanu repozytorium zdalnego.

```
gowinm:~> mkdir line-counter-another
```

```
gowinm:~> cd line-counter-another
```

```
gowinm:~/line-counter-another> git clone https://github.com/MaciejGowin/line-counter.git .
Cloning into '.'...
remote: Enumerating objects: 23, done.
remote: Counting objects: 100% (23/23), done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 23 (delta 2), reused 23 (delta 2), pack-reused 0
Unpacking objects: 100% (23/23), done.
```

Git: przypadki użycia

Sprawdzenie rewizji. Dodanie zmiany.

```
gowinm:~/line-counter> git log -n 2
commit 1771ecf8489bf59cec32dad6e7a679b1633d90d3 (HEAD -> master, origin/master)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:40:54 2021 +0100

    Fifth change

commit 3be9d76885d10da31292db9a6a860f23b969dd70
Merge: 75e6de4 f7d651d
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:03:37 2021 +0100

    Merge branch 'feature'
```

Git: przypadki użycia

Wysłanie zmian do zdalnego repozytorium.

```
gowinm:~/line-counter-another> vi sample
```

```
gowinm:~/line-counter-another> git commit -am "Remote first change"
[master 383996a] Remote first change
 1 file changed, 1 insertion(+)
```

```
gowinm:~/line-counter-another> git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 325 bytes | 325.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/MaciejGowin/line-counter.git
   1771ecf..383996a  master -> master
```

Git: przypadki użycia

Sprawdzenie stanu.

```
gowinm:~/line-counter-another> git log -n 1
commit 383996a9df19b3daa26b7abc6c6721efdd73b1e8 (HEAD -> master, origin/master, origin/HEAD)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Wed Dec 8 00:06:12 2021 +0100

    Remote first change
```

Git: przypadki użycia

Przejście do oryginalnego repozytorium lokalnego. Porównanie zmian ze zmianami zewnętrznymi.

```
gowinm:~/line-counter> git log master..origin/master
```

```
gowinm:~/line-counter> git log -n 2
commit 1771ecf8489bf59cec32dad6e7a679b1633d90d3 (HEAD -> master, origin/master)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:40:54 2021 +0100

    Fifth change

commit 3be9d76885d10da31292db9a6a860f23b969dd70
Merge: 75e6de4 f7d651d
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:03:37 2021 +0100

    Merge branch 'feature'
```

Git: przypadki użycia

Pobranie zmian ze zdalnych gałęzi. Porównanie zmian ze zmianami zewnętrznymi.

```
gowinm:~/line-counter> git fetch
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/MaciejGowin/line-counter
   1771ecf..383996a  master     -> origin/master
```

```
gowinm:~/line-counter> git log master..origin/master
commit 383996a9df19b3daa26b7abc6c6721efdd73b1e8 (origin/master)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Wed Dec 8 00:06:12 2021 +0100

    Remote first change
```

Git: przypadki użycia

Sprawdzenie zmian.

```
gowinm:~/line-counter> git log -n 2
commit 1771ecf8489bf59cec32dad6e7a679b1633d90d3 (HEAD -> master)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:40:54 2021 +0100

    Fifth change

commit 3be9d76885d10da31292db9a6a860f23b969dd70
Merge: 75e6de4 f7d651d
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:03:37 2021 +0100

    Merge branch 'feature'
```

Git: przypadki użycia

Scalenie zewnętrznych zmian ze zdalnego repozytorium.

```
gowinm:~/line-counter> git merge origin/master
Updating 1771ecf..383996a
Fast-forward
 sample | 1 +
 1 file changed, 1 insertion(+)
```

```
gowinm:~/line-counter> git log -n 2
commit 383996a9df19b3daa26b7abc6c6721efdd73b1e8 (HEAD -> master, origin/master)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Wed Dec 8 00:06:12 2021 +0100

    Remote first change

commit 1771ecf8489bf59cec32dad6e7a679b1633d90d3
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Tue Dec 7 23:40:54 2021 +0100

    Fifth change
```

Git: przypadki użycia

Dodanie oraz opublikowanie kolejnych zmian na drugim lokalnym repozytorium.

```
gowinm:~/line-counter-another> vi sample
```

```
gowinm:~/line-counter-another> git commit -am "Remote second change"
[master 96bb69e] Remote second change
 1 file changed, 1 insertion(+)
```

```
gowinm:~/line-counter-another> git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 330 bytes | 330.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/MaciejGowin/line-counter.git
   383996a..96bb69e  master -> master
```

Git: przypadki użycia

Jednorazowe pobranie oraz scalenie zmian z zewnętrznego repozytorium.

```
gowinm:~/line-counter> git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/MaciejGowin/line-counter
   383996a..96bb69e  master     -> origin/master
Updating 383996a..96bb69e
Fast-forward
 sample | 1 +
 1 file changed, 1 insertion(+)
```

```
gowinm:~/line-counter> git log -n 2
commit 96bb69e0d6df284d3685bef8f46ba186262dca16 (HEAD -> master, origin/master)
Author: Maciej Gowin <gowinm@ryanair.com>
Date:   Wed Dec 8 00:16:41 2021 +0100

    Remote second change
```

Git: przypadki użycia

Kompilacja programu. Wynikowe pliki `class` są widoczne podczas sprawdzenia statusu.

```
gowinm:~/line-counter> javac LineCounter.java
```

```
gowinm:~/line-counter> git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        LineCounter.class

nothing added to commit but untracked files present (use "git add" to track)
```

Git: przypadki użycia

Pominięcie plików o rozszerzeniu `class` przez Git poprzez dodanie ich do `.gitignore`.

```
gowinm:~/line-counter> echo "*.class" > .gitignore
gowinm:~/line-counter> git add .gitignore
gowinm:~/line-counter> git commit -m "Define gitignore"
[master 3635755] Define gitignore
 1 file changed, 1 insertion(+)
 create mode 100644 .gitignore
```

```
gowinm:~/line-counter> git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 330 bytes | 330.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/MaciejGowin/line-counter.git
   96bb69e..3635755  master -> master
```

Git: przypadki użycia

Sprawdzenie statusu zmian.

```
gowinm:~/line-counter> ls
LineCounter.class       LineCounter.java        sample
```

```
gowinm:~/line-counter> git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
```

Git: przypadki użycia

Nadpisanie autora.

```
gowinm:~/line-counter> git config user.name "Maciej Henryk Gowin"
gowinm:~/line-counter> cat .git/config
[core]
        repositoryformatversion = 0
        filemode = true
        bare = false
        logallrefupdates = true
        ignorecase = true
        precomposeunicode = true
[remote "origin"]
        url = https://github.com/MaciejGowin/line-counter.git
        fetch = +refs/heads/*:refs/remotes/origin/*
[branch "master"]
        remote = origin
        merge = refs/heads/master
[user]
        name = Maciej Henryk Gowin
```

Git: podsumowanie komend

```
git init
git clone <REMOTE_URL>
git add <FILE>
git commit -m <MESSAGE>
git commit -am <MESSAGE>
git status
git diff
git log
git log -n <LIMIT>
git log <BRANCH1>..<BRANCH2>
git branch
git branch -r
git branch <BRANCH>
git checkout <BRANCH>
git merge <BRANCH>
git remote add <REMOTE_ID> <REMOTE_URL>
git push
git push -u <REMOTE_ID> <BRANCH>
git fetch
git pull
```