Technologia Programowania 2017/2018 – Lista 2 (lab)
Termin: do 27 października

PMD, SpotBugs, Checkstyle, UMLet, Maven

Zadanie 1 — **PMD** to narzędzie pomagające wykryć złe praktyki programistyczne. Kierując się instrukcjami ze tej strony zainstaluj je w Eclipse[1]. Dla programu z Zadania 3 z Listy 1 wybierz *PMD → Check Code*. W oknie *Violations Overview* wybierz element zawierający błędy. W oknie *Violations Outline* wybierz *Show details*. W oknie *Window → Preferences → PMD → Rule Configuration* możesz przeczytać opisy błędów wraz z przykładami i zdecydować czy mają być zgłaszane. Sprawdź jakie zbiory błędów (ang. rule set) są zdefiniowane. **(10 p.)**

Zadanie 2 — **SpotBugs** to następca narzędzia FindBugs, służącego do wykrywania potencjalnych błędów programistycznych w kodzie Java[2]. Plug-in SpotBugs możesz zainstalować w Eclipse wybierając w menu *"Help→Eclipse Marketplace"* i wyszukać wpisując *"spotbugs"*. Dla programu z Zadania 3 z Listy 1 wybierz *"SpotBugs→SpotBugs"* i zmień perspektywę na "SpotBugs". Przejrzyj błędy i ich opisy, sprawdź, czy są jakieś błędy niewykryte przez PMD. Konfiguracje możesz przeprowadzić w oknie *Window → Preferences → Java→ SpotBugs* . Możesz również przejrzeć opis wykrywanych błędów oraz krótki tutorial. **(10 p.)**

Zadanie 3 — **Checkstyle** to narzędzie pomagając wykryć błędy w stylu (np. odstępy, nawiasy, konwencje nazw). Możesz je zainstalować wybierając w menu *"Help→Eclipse Marketplace"* i wyszukać wpisując *"Checkstyle Plugin"*. Dla programu z Zadania 3 z Listy 1 wybierz *Checkstyle → Check Code with Checkstyle*. Wybierz *Checkstyle→Clear Checkstyle violations* aby usunąć informacje o błędach. W oknie *Window → Preferences → Checkstyle* możesz ustalić jakie błędy mają być zgłaszane. Skopiuj zestaw domyślnych zasad (np. Google Checks), znajdź zasadę, która mówi, że w kodzie nie powinno być znaków tabulacji, przeczytaj uzasadnienie, wyłącz tę zasadę i zweryfikuj, że związane z nią błędy nie są już znajdowane. **(10 p.)**

Zadanie 4 — **UMLet** to narzędzie umożliwiające w miarę wygodne rysowanie diagramów UML. Pobierz je i skopiuj plik *com.umlet.plugin*.jar* do katalogu Eclipse'a o nazwie *dropins*, a następnie uruchom Eclipse. Aby stworzyć nowy diagram UML wybierz projekt, a następnie z menu *File→New→Other→UMLet diagram*. Przeczytaj FAQ a następnie stwórz diagram klas dla programu z Zadania 3 z Listy 1. **(10 p.)**

Zadanie 5 — **Maven** to narzędzie automatyzujące budowę oprogramowania, ułatwiające zarządzanie zależnościami i wspomagające proces ciągłej integracji. W Eclipse narzędzie Maven jest dostępne domyślnie[3], ale sugeruję by przynajmniej początkowo używać konsoli[4]. Przejrzyj samouczek (i w razie potrzeby dodatkowe materiały, np. te w języku polskim). Odpowiedz na pytania prowadzącego dotyczące tworzenia projektów (archetypy), dodawania zależności, wykonywania celów (ang. goals). Wyjaśnij czym jest *phase, plug-in, goal*. Wskaż na swoim komputerze katalog z pobranymi przez Maven zależności z dowolnego pliku pom.xml. **(20 p.)**

J.L.

[1] Te same lub analogiczne do omawianych na tej liście narzędzi istnieją dla IntelliJ.

[2] Zobacz krótkie porównanie PMD, FindBugs oraz Checkstyle.

[3] W Eclipse dla wybranego projektu Java wybierz *Configure → Convert to Maven project*.

[4] Konsolową wersję Maven możesz pobrać tutaj. Działa również w systemie Windows – upewnij się, że skrypt mvn w katalogu bin jest dostępny na ścieżce systemowej.