PAŃSTWOWA WYŻSZA SZKOŁA ZAWODOWA IM. ST. STASZICA W PILE

Opracowanie autorskie: *Inż.* ***Mirosław Ochodek***
Konsultacja metodyczna: *Prof. PWSZ dr inż.* ***Bolesław Ochodek***

PROGRAMOWANIE OBIEKTOWE

Laboratorium numer 1
Przygotowanie stanowiska programistycznego

Piła 2004

1. Pobranie i instalacja Java SDK

Istnieje wiele implementacji maszyny wirtualnej **Javy** przeznaczonych dla różnych platform systemowych. Podczas zajęć laboratoryjnych używana będzie implementacja dostarczona przez firmę **Sun Microsystems** w wersji dla systemu **Microsoft Windows**.

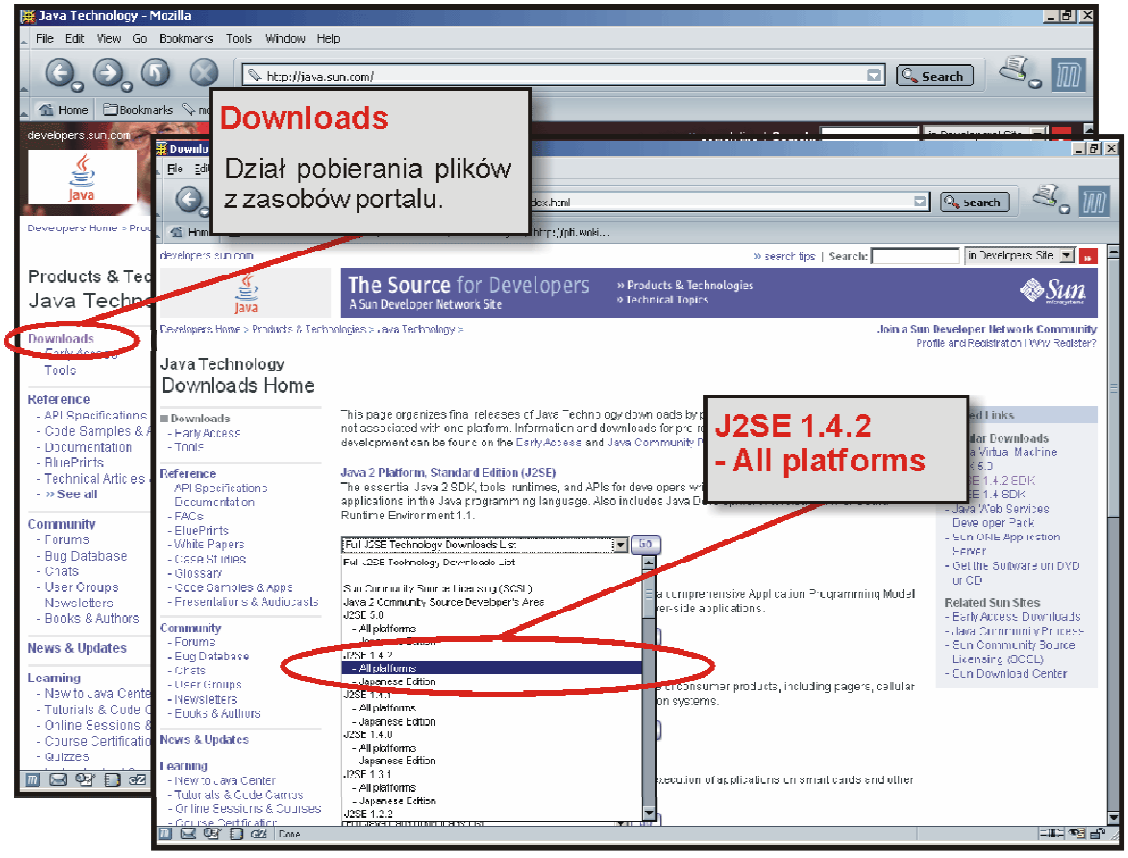
Jako, że instalowanie oprogramowania blisko związanego z systemem dozwolone jest wyłącznie administratorom systemu, niektóre fragmenty procesu instalacji zostaną pokazane na rysunkach i omówione teoretycznie.

1.1. Pobranie instalacji Javy SDK ze strony firmy Sun Microsystems

W celu pobrania wersji instalacyjnej Javy, należy odwiedzić stronę:

http://www.java.sun.com

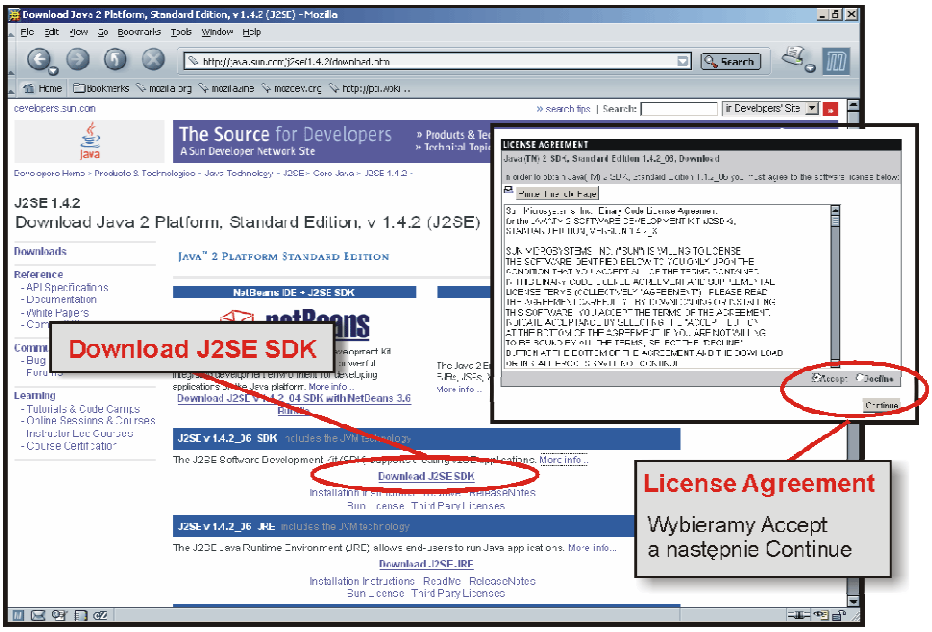
Następnie należy udać się do działu ***Downloads***. Kolejnym etapem jest wybranie z listy rozwijalnej interesującą nas wersje. Zaznaczamy **J2SE 1.4.2 – All platforms**, czyli **Java 2 Standard Edition**, wszystkie platformy. Java w wersji Standard Edition jest przeznaczona do użytku bezpłatnie (pod pewnymi ograniczeniami). Po dokonaniu wyboru należy przejść do następnej strony za pomocą przycisku **GO**, umieszczonego po prawej stronie listy rozwijalnej. Dwa pierwsze kroki pokazano na rysunku 1.



Rysunek 1. Wybór działu downloads i wersji Javy

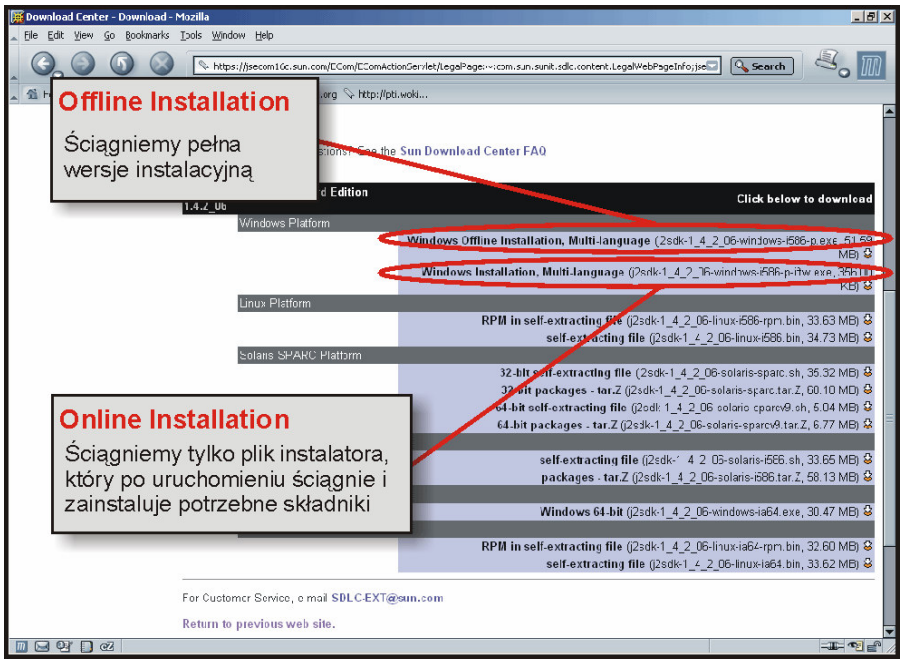
Kolejne etapy pobierania wersji instalacyjnej przedstawiono na rysunku 2. Na pierwszej stronie wybieramy ***Download J2SE SDK***. Jako wynik otrzymamy stronę z warunkami licencji

użytkowania **Java Standard Edtition**. Po zapoznaniu się z umową wybieramy ***Accept*** i przechodzimy dalej wybierając przycisk ***Continue***.



Rysunek 2. Zapoznanie się z umową licencyjną

Ostatnim etapem jest wybranie odpowiedniej wersji dla naszej platformy systemowej. Na rysunku 3 przedstawiono stronę wyboru wersji. Dla systemów Microsoft Windows można wybrać wersję do instalacji Offline i Online. Pierwsza jest w postaci dużego pliku (około 50 MB) i jest to kompletna wersja instalacyjna. Natomiast druga wersja to mały plik instalatora, który po ściągnięciu i uruchomieniu pobierze ze strony wymagane do instalacji komponenty.



Rysunek 3. Wybór wersji dla platformy systemowej

1.2. Instalacja Javy SDK

Po uruchomieniu instalatora wystarczy przejść przez wszystkie okna by zainstalować w sposób standardowy Javę. Warto zapamiętać (lub ustawić według własnych preferencji) docelowy folder instalacji. Informacja ta będzie przydatna w ustawianiu zmiennych środowiskowych.

Jeśli instalacja przebiegła pomyślnie dysponujemy już własną wersją Javy i teoretycznie jesteśmy gotowi do pracy. Jednak, jeśli nie korzystalibyśmy z żadnego zintegrowanego środowiska programistycznego tylko pracowalibyśmy przy użyciu notatnika i konsoli, warto ułatwić sobie pracę rozszerzając zmienną systemową **PATH** o ścieżkę do folderu **bin** naszej instalacji.

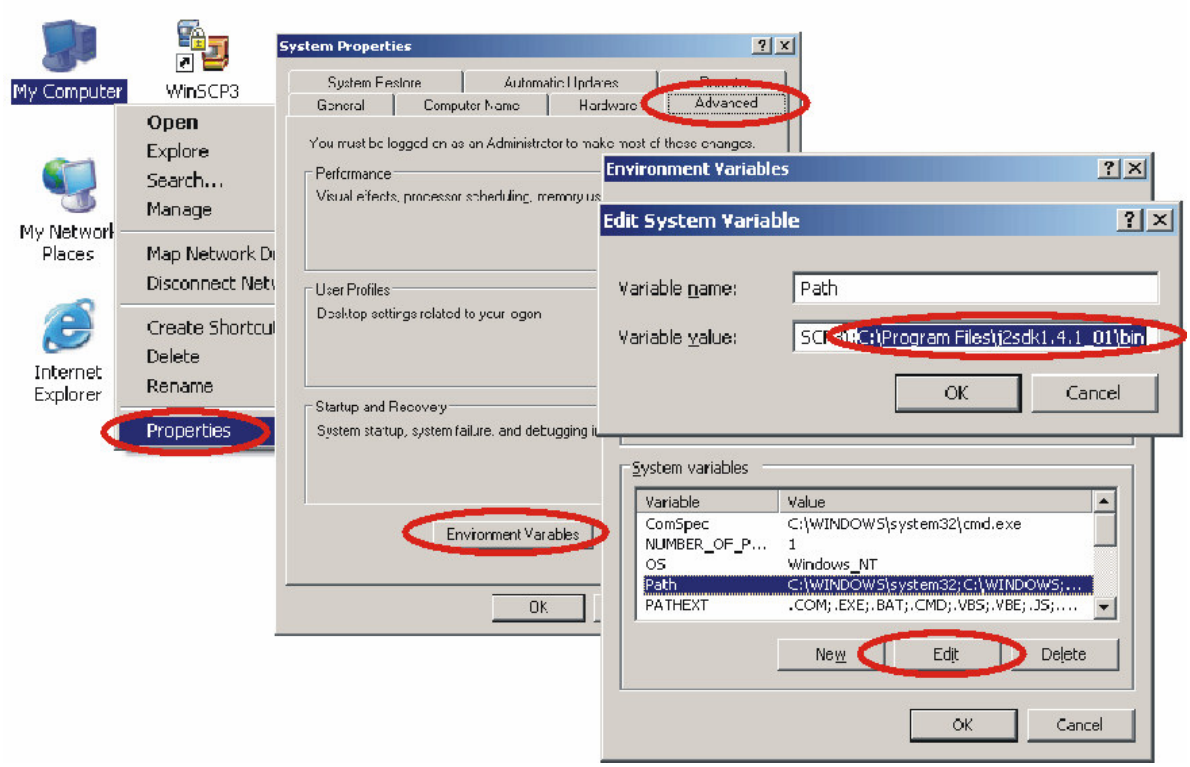
Dodawanie ścieżki do katalogu **bin** zainstalowanej na naszym komputerze różni się w zależności od wersji systemu Windows. Jeśli posiadamy wersje 95,98 lub Me należy do pliku ***autoexec.bat*** dopisać następującą linie:

SET PATH=%PATH%;Dysk:\Ścieżka_do_folderu_Javy\bin

Jeśli na przykład zainstalowaliśmy Javę w folderze **C:\Program Files\j2sdk1.4.1_01** to wpis do pliku wyglądałby następująco:

SET PATH=%PATH%;C:\Program Files\j2sdk1.4.1_01\bin

Jeżeli dysponujemy systemem Windows NT,2000,XP to możemy dodać zmienną systemową wpisując wspomnianą już ścieżkę w okno systemowe pokazane na rysunku 4.



Rysunek 4. Ustawianie ścieżki w systemie Windows NT,2000,XP

2. Praca z językiem Java przy użyciu konsoli i notatnika

Po zainstalowaniu Javy można przystąpić do prac programistycznych. Do napisania prostego programu wystarczy zwykły edytor tekstu, a do uruchomienia konsola systemowa. Pierwszym napisanym programem w Javie będzie prosty program wypisujący komunikat „Witaj Świecie".

2.1. Tworzenie prostego programu przy użyciu notatnika

Stworzymy w notatniku prosty program. Nie przejmuj się, jeśli nie rozumiesz dokładnie każdej linijki kodu. Program będzie posiadał jedna klasę – **WitajSwiecie**. Klasa ta zawiera metodę **main**, która zostanie wywołana zaraz po uruchomieniu programu. Gotowy program mógłby wyglądać tak:

```
public class WitajSwiecie {
    public static void main(String[] args) {
      System.out.println("Witaj Swiecie!");
    }
}
```

Plik należy zapisać pod nazwą **WitajSwiecie.java**, ponieważ pliki w języku Java, muszą nazywać się tak jak klasa, której definicja znajduje się w środku i mają rozszerzenie **.java**.

2.2. Kompilacja pliku źródłowego przy użyciu konsoli systemowej

Po utworzeniu pliku należy otworzyć konsole systemową i dokonać kompilacji utworzonej klasy do postaci **bajtkodu**. Do kompilacji plików źródłowych służy program javac, czyli Java Compiler. Kompilacji stworzonej klasy możemy dokonać przy użyciu polecenia:

```
C:\>javac scieżka_do_folderu\Nazwa_klasy.java
```

Zatem zakładając, że plik źródłowy umieszczono na dysku D, w głównym folderze, komenda wyglądałaby następująco:

```
C:\>javac D:\WitajSwiecie.java
```

Podczas kompilacji wyświetlone mogą zostać informacje odnośnie błędów składniowych lub ostrzeżenia. Na przykład, jeśli w deklaracji klasy zamiast słowa **public** wpisalibyśmy **pblic**, kompilator wygeneruje następujący komunikat:

```
C:\>javac D:\WitajSwiecie.java
D:\WitajSwiecie.java:1: `class` or `interface` expected
pblic class WitajSwiecie {
^
1 error
```

Jeśli nie wyświetlono żadnego komunikatu kompilacja przebiegła prawidłowo i w katalogu oprócz pliku źródłowego powinien znajdować się plik skompilowanej klasy **WitajSwiecie.class**.

2.3. Uruchomienie programu przy użyciu konsoli systemowej

Do uruchomienia skompilowanego programu służy program java. W celu uruchomienia skompilowanej klasy **WitajSwiecie** wpisujemy polecenie:

```
C:\>java -classpath scieżka_do_folderu Nazwa_klasy
```

Przełącznik –classpath informuje, w którym folderze (folderach) mają być poszukiwane skompilowane pliki klas (pliki z rozszerzeniem **.class**). Zatem jeśli plik **WitajSwiecie.class** znajduje się w głównym folderze dysku D, to polecenie uruchamiające program wyglądałoby następująco:

```
C:\>java -classpath D: WitajSwiecie
```

Należy zwrócić uwagę na brak rozszerzenia po nazwie klasy. W tym miejscu podajemy tylko nazwę klasy a maszyna wirtualna będzie poszukiwać pliku **.class** we wskazanych przez classpath lokalizacjach.

Jako wynik uruchomienia programu **WitajSwiecie** w oknie konsoli pojawi się wynik:

```
C:\>java -classpath D: WitajSwiecie
Witaj Swiecie!
C:\>
```

Znak zachęty informuje nas, że program zakończył działanie.

3. Pobranie i instalacja środowiska Eclipse

Jak łatwo zauważyć tworzenie programów przy pomocy notatnika i uruchamianie ich w konsoli systemowej jest mało wygodne. Przy każdej, nawet najmniejszej zmianie, trzeba każdy plik klasy z rekompilować i uruchomić program. Dodatkową niedogodnością jest fakty, iż o popełnionych błędach składniowych dowiadujemy się po próbie kompilacji.

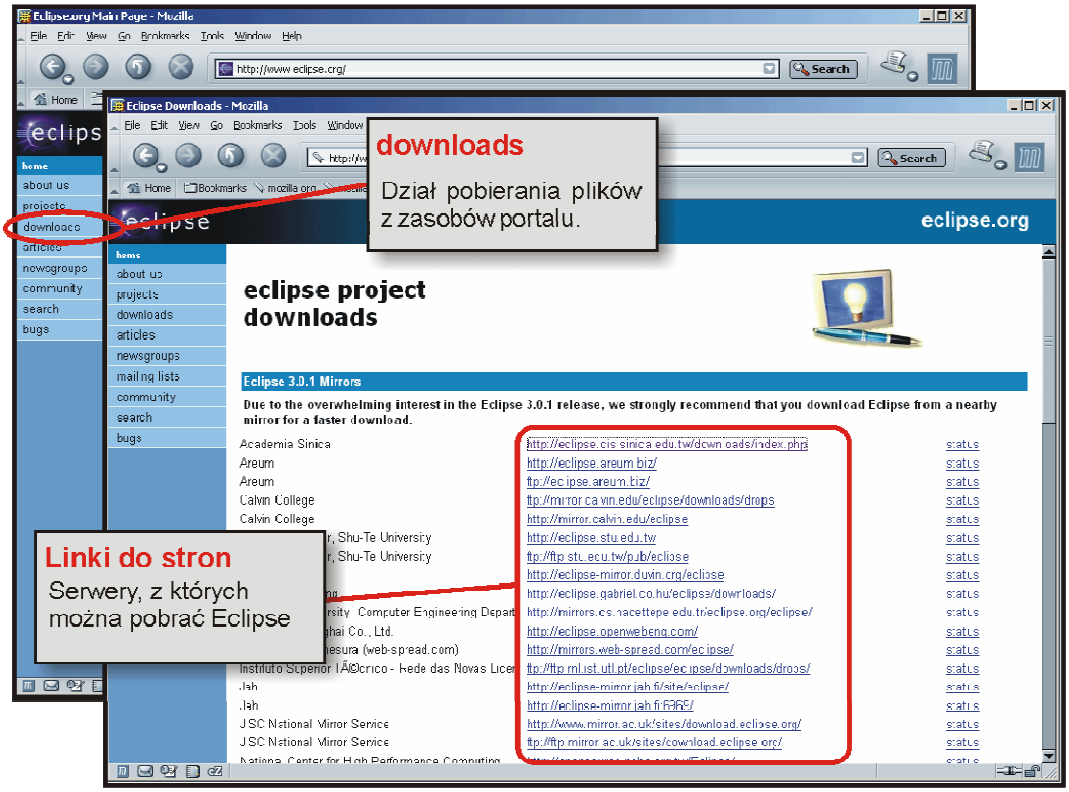
Rozwiązaniem, które w znaczący sposób podnosi efektywność tworzenia aplikacji są zintegrowane środowiska programistyczne (IDE). Na rynku jest wiele takich rozwiązań. Jednym z najlepszych (nie tylko do tworzenia programów w Javie) jest środowisko opracowane przez specjalistów z IBM o nazwie Eclipse. Środowisko to posiada bardzo zaawansowane funkcje, które w innych dostępnych IDE są wciąż rzadko spotykane. Dodatkowo autorzy wymyślili bardzo dobrze działający system wtyczek, które umożliwiają budowanie własnych dodatków (na przykład do tworzenia aplikacji w innych językach programowania).

3.1. Pobranie wersji instalacyjnej środowiska Eclipse

Środowisko **Eclipse** jest programem udostępnianym darmowo. Wersje instalacyjną można pobrać ze strony:

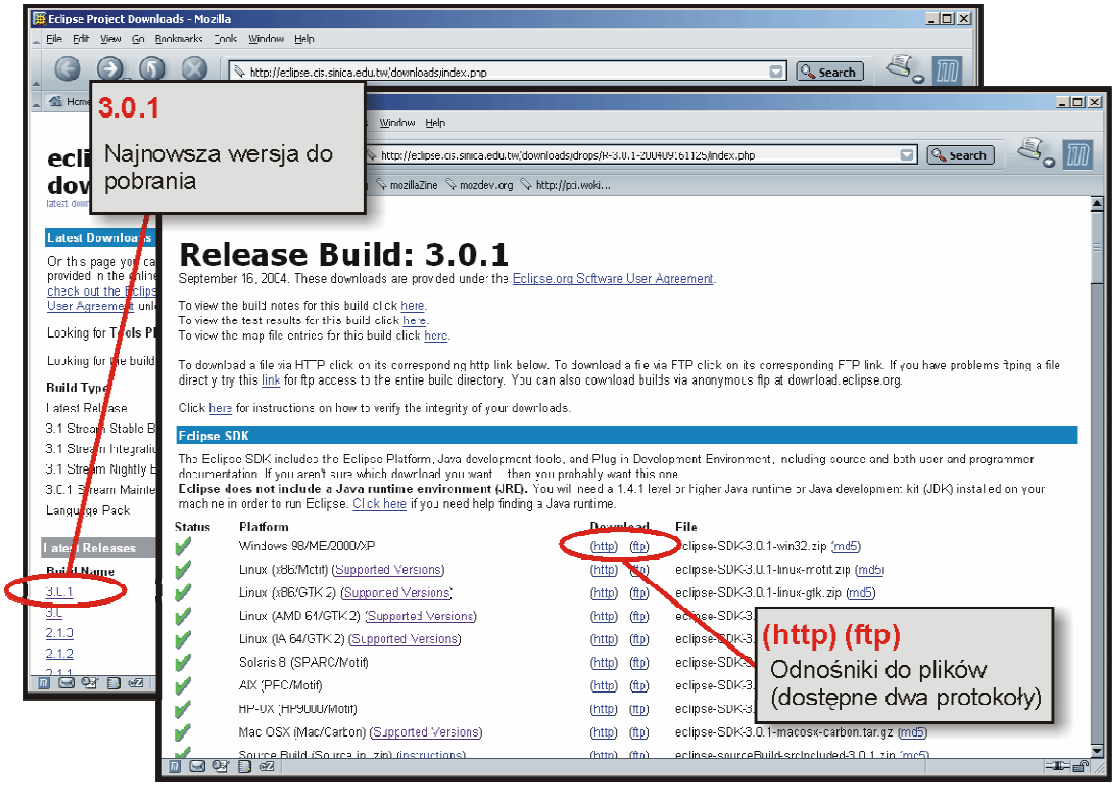
http://www.eclipse.org

Ze strony powitalnej wybieramy dział ***downloads***. Z następnej strony należy wybrać jeden z serwerów, który zawiera wersje instalacyjne środowiska. Najkorzystniej jest wybrać serwer zlokalizowany możliwie blisko miejsca zamieszkania, wówczas czas pobierania plików będzie prawdopodobnie najkrótszy. Przejście przez pierwsze dwie strony pokazano na rysunku 5.



Rysunek 5. Wybieranie serwera do pobrania instalacji Eclipse

Kolejnym etapem przedstawionym na rysunku 6, jest wybranie wersji środowiska. Sugerujemy wybieranie najnowszej dostępnej, pełnej wersji (***latest release***). Może się zdarzyć, że nowsze wersje środowiska będą wymagały także nowszych wersji maszyny wirtualnej Javy! W kolejnym oknie dostępny jest odnośnik do interesującej nas wersji systemowej.



Rysunek 6. Pobranie wersji instalacyjnej Eclipse

3.2. Instalacja środowiska Eclipse

Do uruchomienia środowiska niepotrzebna jest specjalnie przeprowadzona instalacja. Wystarczy rozpakować pobrane archiwum zip i uruchomić program **eclipse.exe**.

4. Konfiguracja środowiska Eclipse

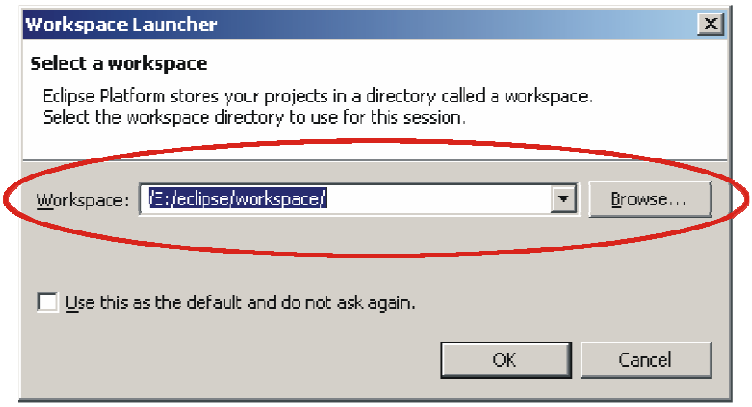
Środowisko Eclipse jest bardzo złożoną aplikacją, wiec nie sposób jest opisać tu znacznej większości funkcji i możliwości ustawień. Poniższe porady są bardzo elementarne, ale ich celem jest dokonanie podstawowych ustawień umożliwiających pracę.

4.1. Ustawienie przestrzeni roboczej (Workspace)

Podczas uruchamiania środowiska Eclipse zostaniemy zapytani o lokalizacje przestrzenie roboczej (***Workspace***).

Przestrzeń robocza to folder, w którym automatycznie będą zapisywane nasze projekty. Każdy użytkownik może mieć odrębny workspace. Czasem okazuje się, że nawet w przypadku użytkowania środowiska przez jednego programistę, warto zdefiniować kilka przestrzeni. Projekty umieszczalibyśmy w przestrzeniach w zależności od ich przeznaczenia (np. Studia, Praca itd.). Jeśli używamy tylko jednej przestrzeni możemy zaznaczyć ***Use this as default and do not ask again***, aby zdefiniowana przestrzeń była wybierana domyślnie.

Na rysunku 7 przedstawiono przykładowe okienku wyboru przestrzeni roboczej (oczywiście nazwa folderu może być inna niż workspace).



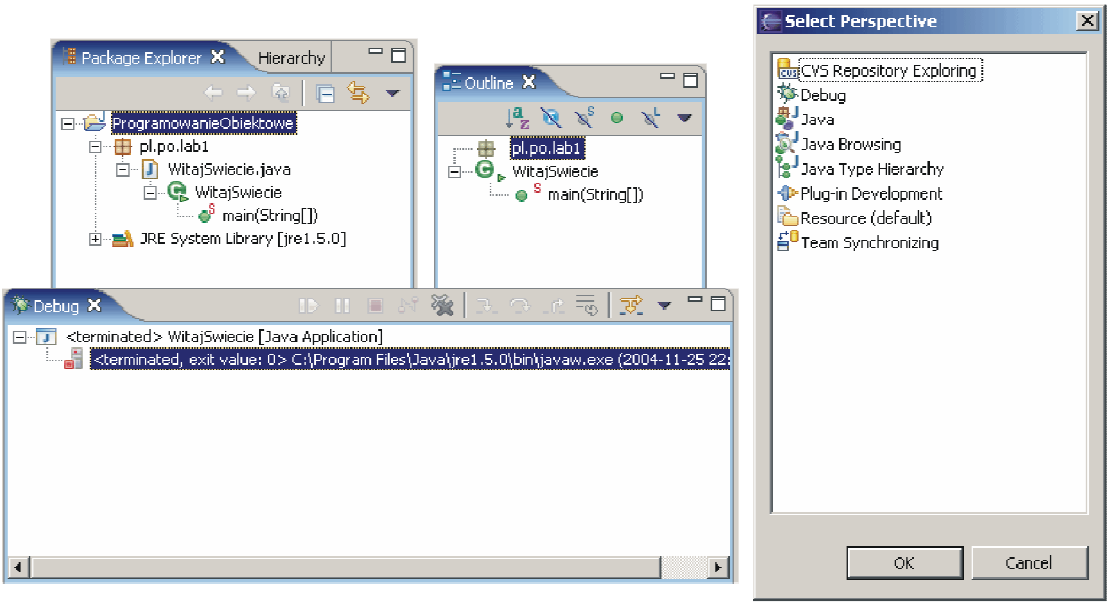
Rysunek 7. Okno wyboru przestrzeni roboczej

Jeżeli w trakcie pracy chcemy zmienić przestrzeń roboczą należy wybrać z menu ***File*** opcje ***Open Workspace***, w efekcie pojawi się okno pokazane na rysunku 7.

4.2. Perspektywy i widoki w ustawieniach układu okna środowiska

Organizacja okna środowiska oparty jest na dwóch elementach. Najmniejszą jednostką jest widok. Widoki to pojedyncze okienka zawierające informacje dotyczące jednego zagadnienia. Przykładowe widoki, wraz z podpisami pokazano na rysunku 8. Niektóre z nich zostaną opisane trochę bardziej szczegółowo w dalszej części.

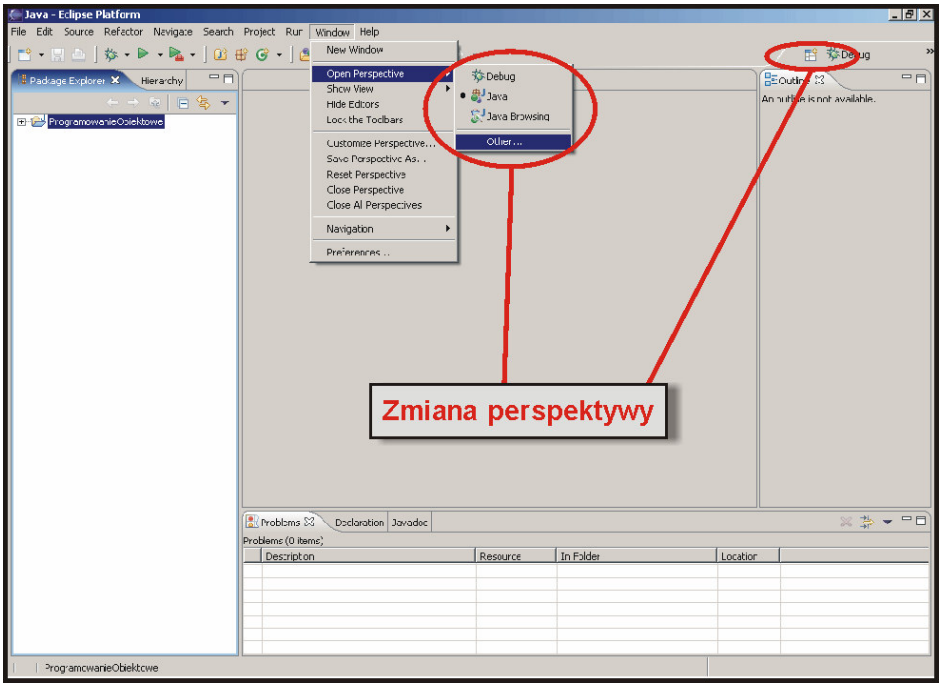
Ustawienia wszystkich widoków w obrębie okna, ich stan (np. czy są widoczne) nazywamy perspektywami. Standardowo dostępne jest kilka perspektyw, każda zawiera widoki dobrane w zależności od przeznaczenia. Dostępne perspektywy przedstawiono także na rysunku 8. Oczywiście użytkownik może tworzyć własne perspektywy, dzięki czemu istnieje możliwość pełnego dostosowania środowiska pracy do swoich potrzeb.



Rysunek 8. Przykładowe widoki i wykaz standardowych perspektyw

Do kodowania najbardziej wygodna wydaje się być perspektywa Java, natomiast do śledzenia wykonania programu niezbędna jest perspektywa ***Debug***.

Zmiany aktualnie wybranej perspektywy możemy dokonać z menu **Window** wybierając ***Open Perspective*** lub przy użyciu przycisku znajdującego się po na pasku narzędzi. Obie metody zostały pokazane na rysunku 9.



Rysunek 9. Zmiana perspektywy

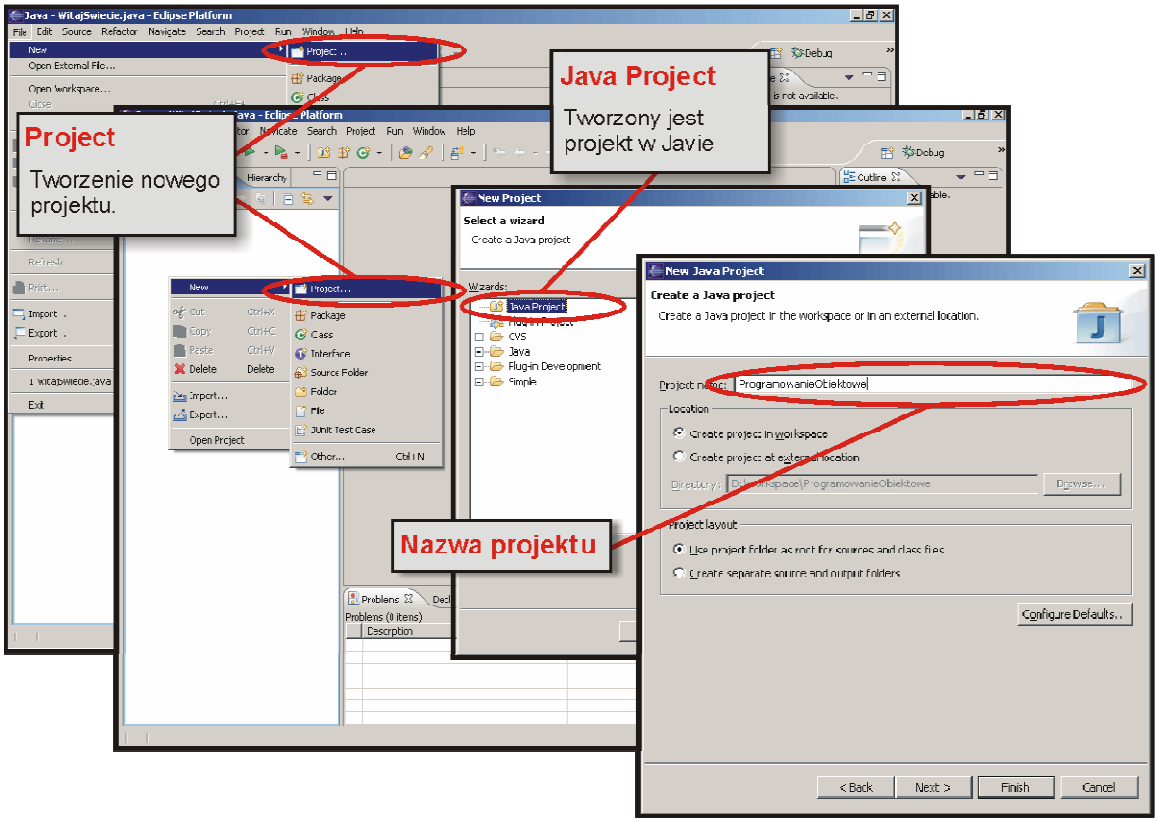
5. Podstawowe funkcje Eclipse w tworzeniu programów

Spróbujmy wykorzystać środowisko programistyczne **Eclipse** tworząc program identyczny do przedstawionego w rozdziale 2. W trakcie tworzenia programu, pokazane zostaną podstawowe funkcje udostępniane przez środowisko.

5.1. Tworzenie projektu

Najwyższym poziomem logicznym w strukturze zarządzania kodem jest Projekt (Project). Nie jest to element Javy, ale struktura wprowadzona w środowisku dla ułatwienia organizacji i zarządzania tworzonym oprogramowaniem.

Dla potrzeb ćwiczeń stwórzmy projekt **ProgramowanieObiektowe**, w którym będziemy przechowywać programy. Aby stworzyć projekt możemy wybrać z menu ***File*** opcje ***New*** *i* ***Project***, lub w widoku ***Package Explorer*** kliknąć prawym przyciskiem i z menu kontekstowego wybrać podobnie ***New*** i ***Project***. Oby dwie ścieżki postępowania pokazano na rysunku 10.



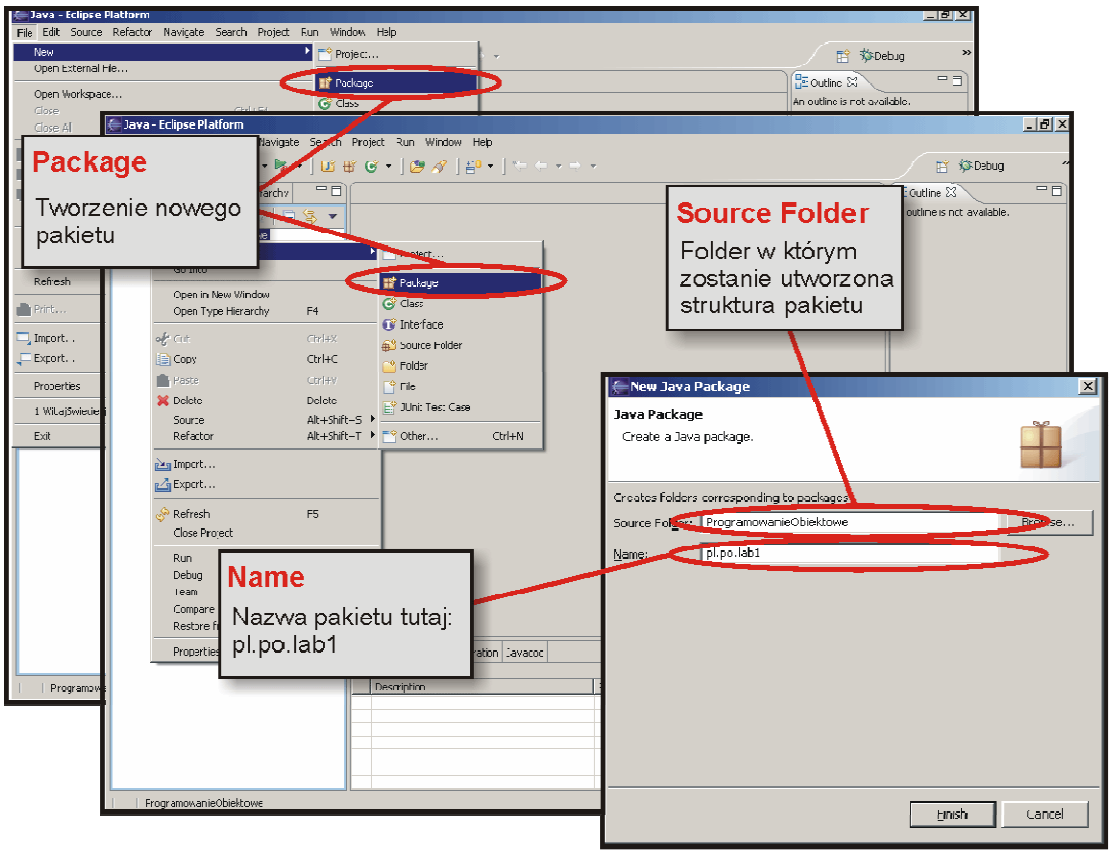
Rysunek 10. Tworzenie nowego projektu

5.2. Tworzenie pakietu

Pakiety w przeciwieństwie do projektów są elementem języka Java. Podstawową rolą grupowania kodu w pakiety jest modułowość. W obrębie jednego pakietu powinny znajdować się klasy ze sobą powiązane zależnościami logicznymi bądź ich przeznaczeniem.

Nazewnictwo pakietów przypomina adresowanie używane przy tworzeniu stron internetowych. Różnica polega na tym, że nazwę pakietu zapisujemy od tyłu. Na potrzeby programu **WitajSwiecie** stworzymy pakiet **pl.po.lab1**. Nazwa stworzonego pakietu pochodzi od nazwy kraju (**pl**), programowania obiektowego (**po**) oraz jest to pierwsze laboratorium (lab1). Stworzenie takiego pakietu zaowocuje utworzeniem podobnej struktury folderów. Pierwszym folderem będzie pl, jego podfolderem będzie po. Natomiast po zawierać będzie folder lab. 1. Dopiero w tym folderze znajdować się będą klasy programu.

Aby utworzyć pakiet zaznaczamy projekt **ProgramowanieObiektowe** w widoku **Package Explorer** i podobnie jak przy tworzeniu projektu, możemy kliknąć prawym przyciskiem na nazwie projektu i wybrać New a następnie Package lub tę samą opcje z menu ***File***. Tworzenie pakietu przedstawiono na rysunku 11.



Rysunek 11. Tworzenie pakietu

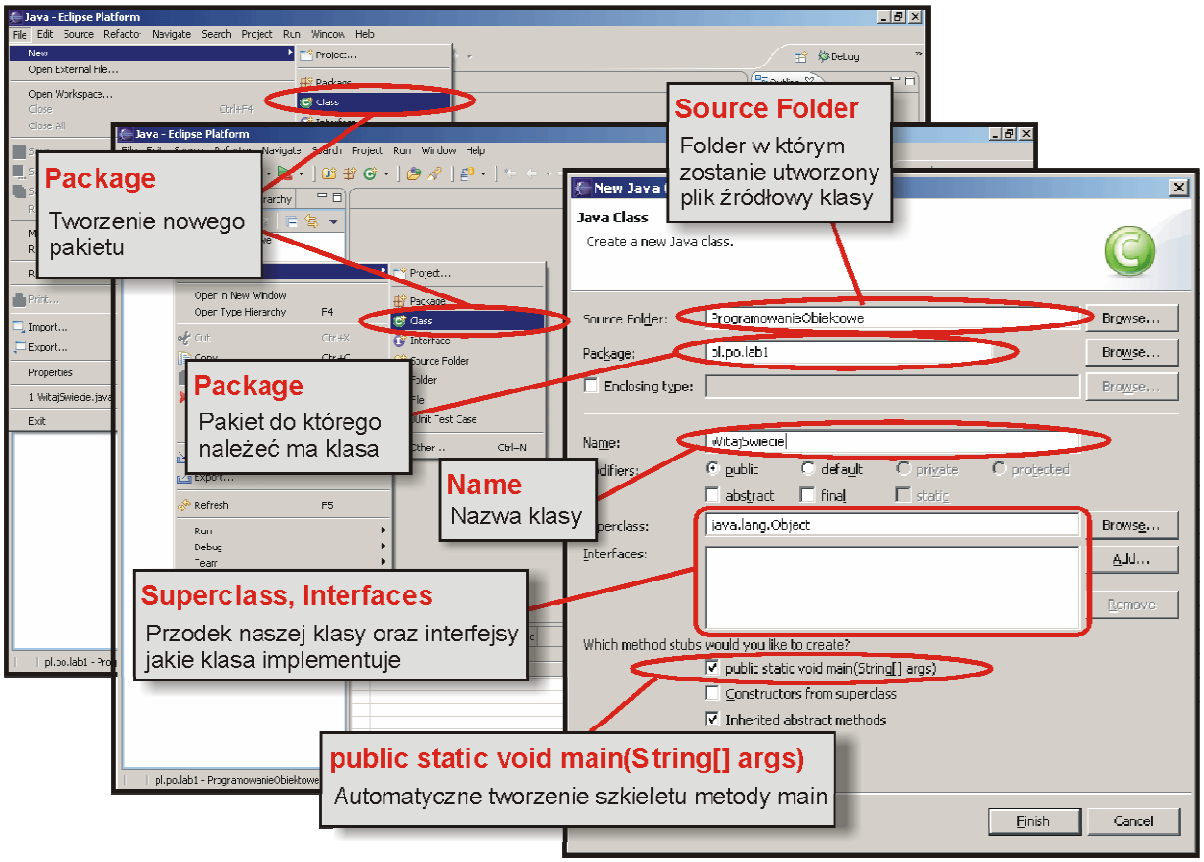
5.3. Tworzenie klas

Środowisko Eclipse umożliwia automatyczne stworzenie klasy, czyli utworzenie pliku, umieszczenie go w odpowiednim miejscu struktury folderów oraz automatyczne wygenerowanie szkieletu klasy. Aby utworzyć nową klasę należy w widoku **Package Explorer** zaznaczyć miejsce, w którym ma znajdować się klasa (w naszym przypadku jest to pakiet pl.po.lab1). Następnie z menu ***File*** wybrać opcje ***New***, a następnie ***Class*** lub dokonać tego samego klikając prawym przyciskiem myszy na nazwie pakietu i wybierając z menu kontekstowego.

Kolejnym etapem jest wypełnienie formularza. Możemy podać źródłowy folder i pakiet, nadać nazwę klasie, a także zdefiniować przodków jej przodków. Dodatkowo w dolnej części

okna możemy zaznaczyć opcje **public static void main(String[] args)**, co spowoduje automatyczne utworzenie metody wykonywanej podczas uruchamiana klasy jako program.

Rysunek 12 przedstawia sposób tworzenia klasy oraz objaśnia opcje formularza.



Rysunek 12. Tworzenie nowej klasy

Po wykonaniu powyższych czynności w pakiecie **pl.po.lab1** powinna pojawić się nowa klasa WitajSwiecie. Aby ją otworzyć w edytorze klikamy ją dwukrotnie. Po wykonaniu tej czynności w oknie edytora powinien znajdować się kod (pomijając komentarze):

```
public class WitajSwiecie {
      public static void main(String[] args) {
      }
}
```

W celu ukończenia tworzenia klasy dodajmy w metodzie **main** znaną już linijkę kodu wyświetlającą tekst powitalny:

```
System.out.println(”Witaj Swiecie!”);
```

Po wykonaniu powyższej czynności klasa gotowa jest już do skompilowania i uruchomienia. Żeby dokonać kompilacji i uruchomienia nie trzeba używać konsoli systemowej i poleceń javac i java. Eclipse automatycznie wykona wszystkie potrzebne polecenia za nas.

5.4. Wsparcie edytora w tworzeniu kodu

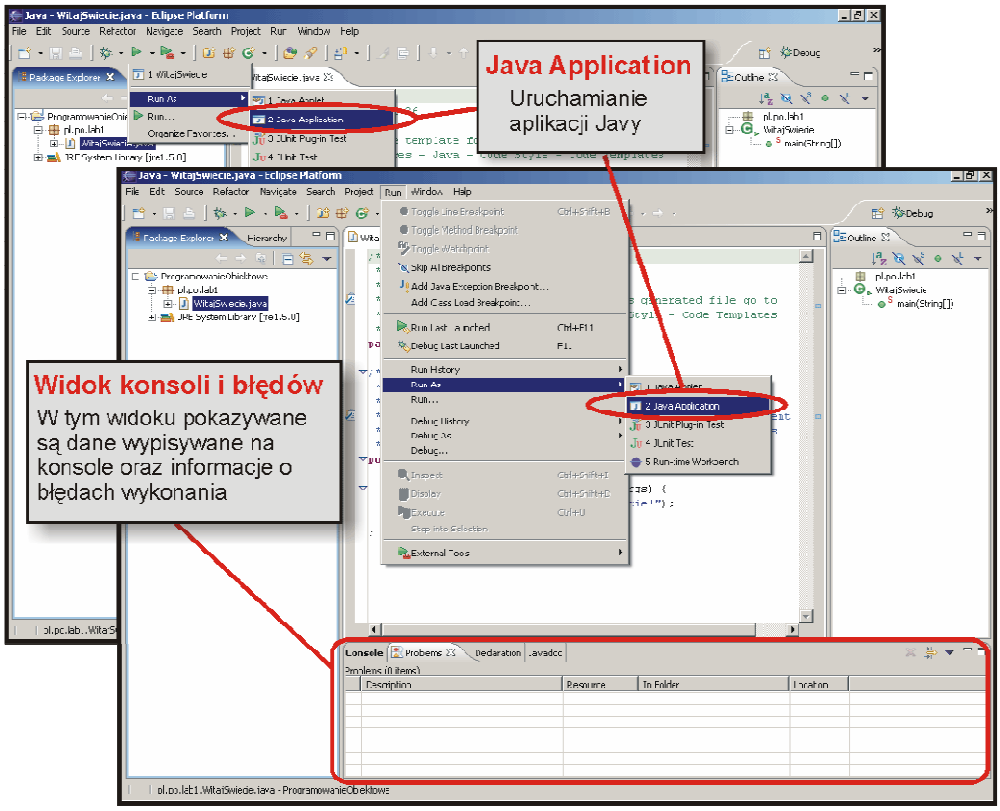
Środowisko Eclipse posiada wsparcie dla programistów podczas pisania kodu w edytorze. Na bieżąco kolorowana jest składnia (np. słowa kluczowe zostają pogrubione), a także na

czerwono podkreślane są błędy składniowe. Ikony w kształcie żarówki pokazujące się po prawej stronie edytora udostępniają podpowiedzi dla programisty. Na przykład, jeśli zapomnieliśmy zadeklarować zmienną, zostaniemy o tym poinformowani. Co więcej po kliknięciu na ikonę środowisko zaproponuje nam pewne rozwiązania zaistniałego problemu.

Inną przydatną własnością edytora jest możliwość przeglądania wszystkich dostępnych pól i metoda klasy, a także uzyskiwanie podpowiedzi na bieżąco. Naciskając kombinację klawiszy ***Ctrl+Spacja*** używając edytora, środowisko spróbuje przedstawić nam wszystkie znane mu alternatywy zakończenia frazy, którą wpisujemy.

5.5. Uruchomienie programu w środowisku Eclipse

Aby uruchomić program wybieramy w widoku Package Explorer plik źródłowy klasy zawierającej metodę main (w naszym przypadku WitajSwiecie.java). Następnie z menu ***Run*** wybieramy ***Run As*** i na końcu ***Java Application***. Podobny efekty możemy uzyskać używając przycisku na pasku narzędzi. Sposób uruchomienia programu pokazano na rysunku 13.



Rysunek 13. Uruchamianie programu w Javie

Oczywiście istnieje wiele innych możliwości uruchamiania programów w środowisku Eclipse, jednak ten wydaje się być najprostszym.

5.6. Eksportowanie klas i zasobów do pliku JAR

Wprawdzie środowisko Eclipse umożliwia łatwe uruchamianie programów, jednak trudno wymagać od użytkownika naszych programów by za każdym razem, gdy będzie chciał sko-

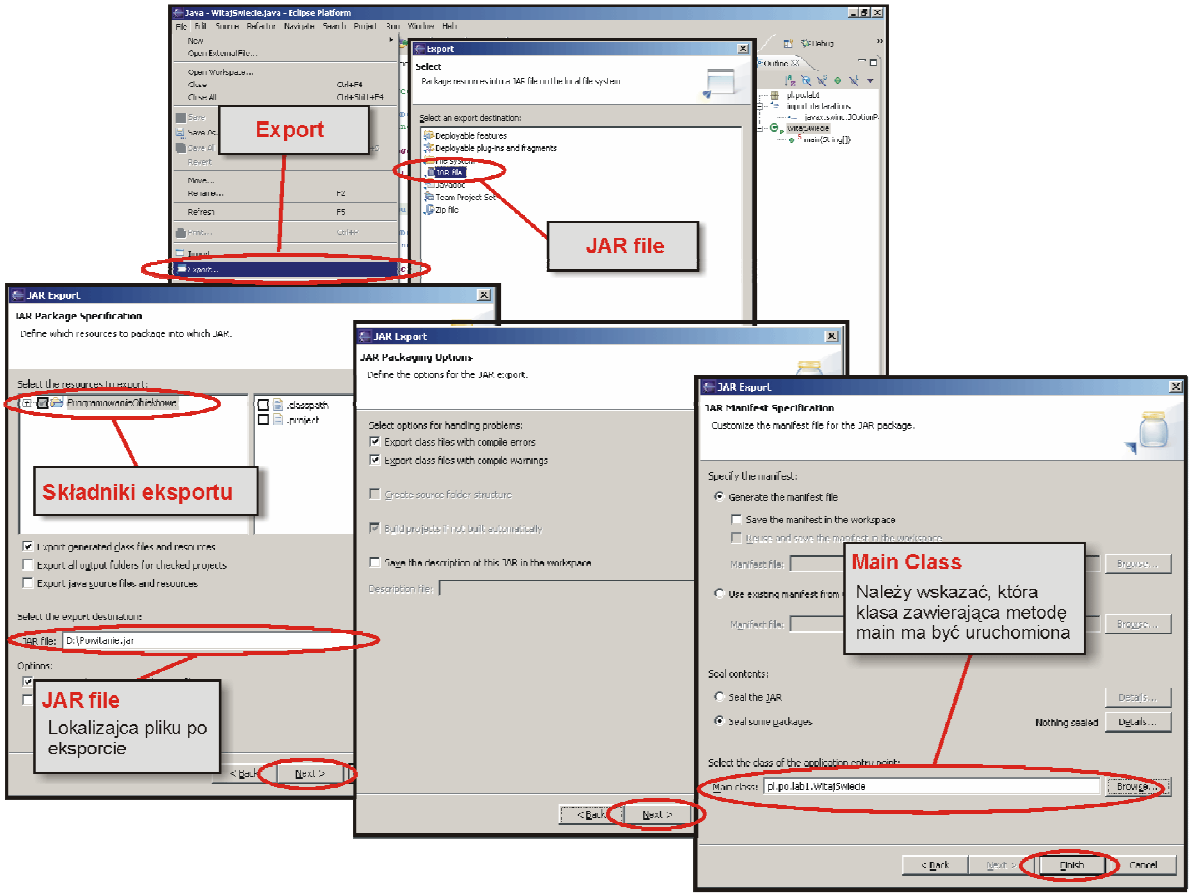
rzystać z aplikacji uruchamiał Eclipse. Jak już zauważyliśmy korzystanie z konsoli też nie należy do komfortowych warunków pracy.

W takiej sytuacji pomocny jest eksport programu do pliku JAR (Java Archive). Jest to plik archiwizowany oparty na kompresji zip. Podczas eksportu wszystkie klasy i zasoby umieszczane są w jednym pliku. Mamy też możliwość wskazania klasy startowej. Wówczas podczas uruchomienia takiego pliku najpierw zostaje on rozpakowany tymczasowo, a następnie uruchomiony.

Zanim umieścimy klase **WitajSwiecie** w pliku JAR, dokonajmy pewnej zmiany. Zamiast wyświetlania powitania na konsoli systemowej ukaże się osobne okno z wiadomością.

Zamieńmy zawartość metody main klasy WitajSwiecie na następującą:

```
JOptionPane.showMessageDialog(null,"Witaj Swiecie!");
```



Rysunek 14. Tworzenie archiwum JAR

Po wpisaniu fragmentu kodu po prawej stronie edytora pojawi się ikona żarówki. Po kliknięciu na niej dostępną będzie opcja „**Import 'JoptionPane' (javax.swing)**". Wybranie jej spowoduje dopisanie deklaracji importu „***import javax.swing.JOptionPane;***" na początku pliku źródłowego klasy. Polecenie import umożliwia korzystanie z klas zawartych w innych pakietach (w tym przypadku jest to pakiet ***javax.swing***).

Jeśli uruchomimy program zaobserwujemy, że w efekcie jego działania wyświetla się okno z tekstem „Witaj Swiecie!". Aby umieścić klasę w archiwum JAR, wybieramy z menu File opcje Export. Dokładną ścieżkę postępowania pokazano na rysunku 14.

Jeśli eksport przebiegł prawidłowo możemy przejść do folderu wskazanego przez nas podczas tworzenia archiwum i uruchomienie go. W efekcie powinniśmy otrzymać okienko z napisem „Witaj Swiecie”.