Wprowadzenie do StdDraw

czyli interfejs graficzny szybko, łatwo i przyjemnie

Co to jest StdDraw?

`StdDraw` jest biblioteką stworzoną przez Roberta Sedgewicka and Kewina Wayne'a z Uniwersytetu w Princeton na potrzeby prowadzonych przez nich zajęć dla studentów z podstaw programowania.

https://introcs.cs.princeton.edu/java/15inout/

Tworzenie programów w języku Java z interfejsem graficznym jest zadaniem nietrywialnym i dosyć czasochłonnym i z pewnością nie jest najlepszym zajęciem dla początkujących. Stąd jak się domyślam powstał pomysł stworzenia biblioteki, dzięki której można w prosty sposób tworzyć wizualizacje danych lub wyników obliczeń w języku Java wymagające interfejsu graficznego.

`StdDraw` w sensie fizycznym jest kodem źródłowym klasy `StdDraw.java` zawierającym właściwości i metody dzieki którym zarówno łatwo wygenerujemy interfejs graficzny, jak i wygenerujemy za jego pomocą żądane wizualizacje.

Właściwości i metody te są statycznymi właściwościami klasy, co oznacza, że nie ma konieczności tworzenia nowych obiektów klasy `StdDraw`, wystarczy odwołanie się do konkretnych, opisanych w dokumentacji właściwości i metod, a reszta zostanie wygenerowana przez kod biblioteki.

(może tutaj dać przykłady jakiś autentycznych wodotrysków?)

Pobieranie i konfiguracja StdDraw

Kod źródłowy biblioteki dostępny jest przez internet pod adresem podanym poniżej.

https://introcs.cs.princeton.edu/java/stdlib/StdDraw.java

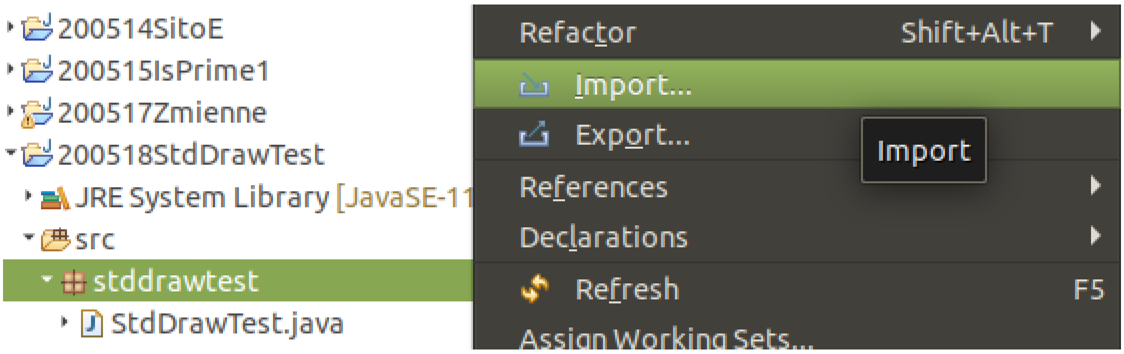
Natomiast dokumentacja pod adresem:

https://introcs.cs.princeton.edu/java/stdlib/javadoc/StdDraw.html .

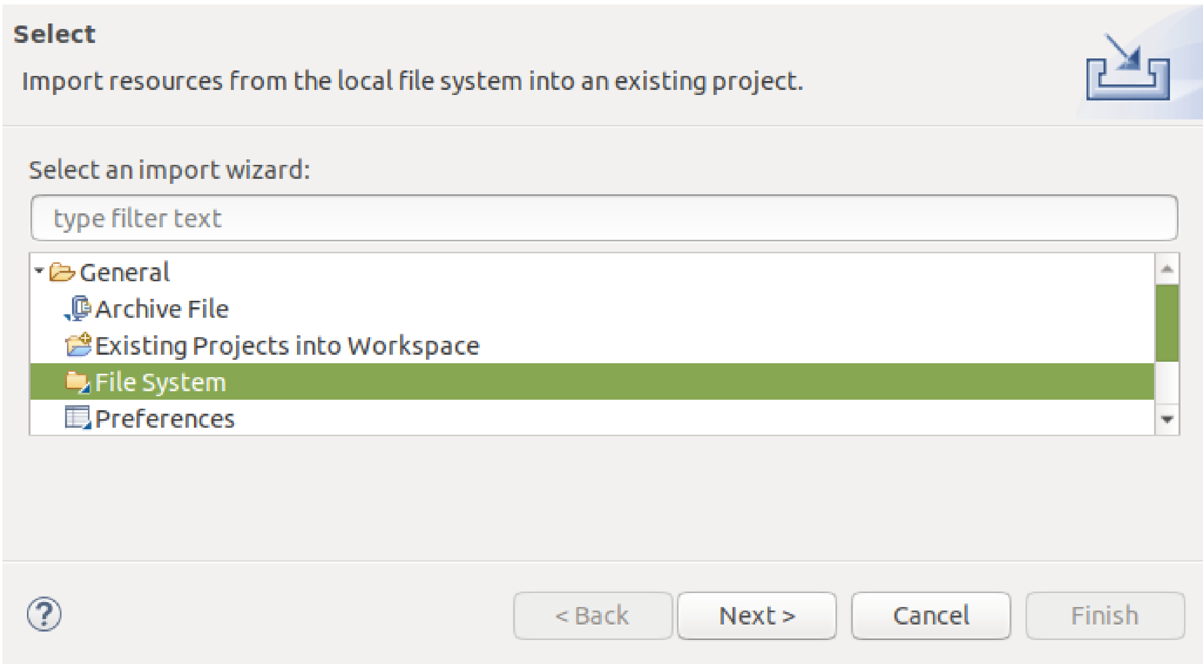
Najprostszym i jak sądzę najbardziej naturalnym sposobem wykorzystania biblioteki jest zaimportowanie kodu źródłowego jej klasy do własnego projektu. Zakładam, że już istnieje taki projekt i nazywa się `StdDrawTest` w skład którego wchodzi pojedynczy pakiet o nazwie `stddrawtest` i pojedynczy plik klasy `StdDrawTest` .

W tym celu w pierwszej kolejności należy zapisać zawartość strony podanej pod pierwszym z linków do pliku tekstowego o nazwie (dokładnie) `StdDraw.java` .

Następnie należy kliknąć prawym klawiszem na pakiet, a następnie z rozwijanego menu wybrać opcję `Import`.

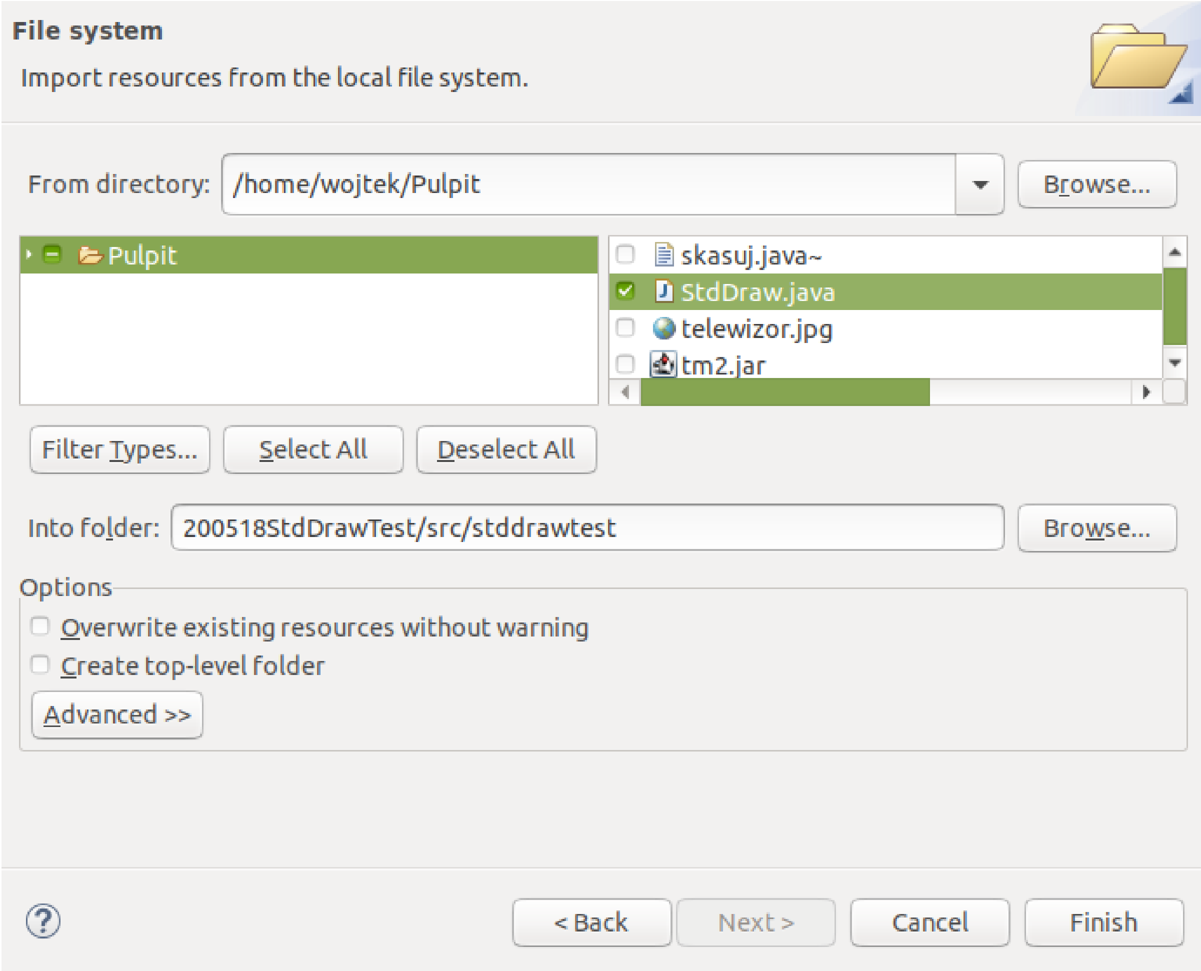


Następnie w oknie Import należy rozwinąć gałąź General , a następnie wybrać opcję `File System` i kliknąć `Next` .

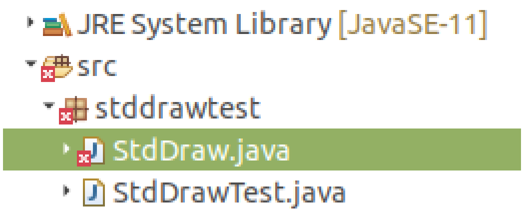


Pojawia się następne okienko w którym należy wybrać:

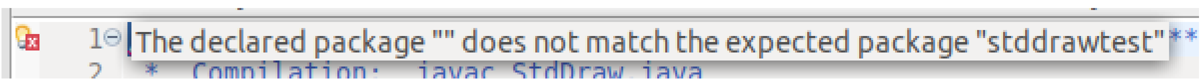
1. Katalog źródłowy (From directory) z którego chcemy importować zasoby do naszego projektu)
2. Pliki które chcemy zaimportować (w tym przypadku StdDraw.java)
3. Katalog docelowy (Into folder) do którego pliki mają zostać skopiowane)
4. I na koniec należy kliknąć przycisk `Finish` .



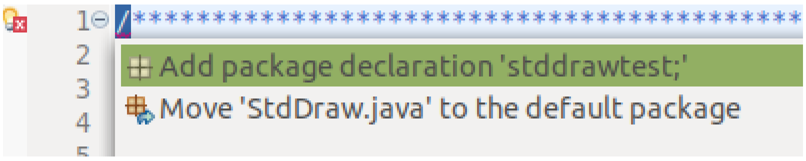
W rezultacie w naszym projekcie w oknie Eksploratora pakietów (po lewej stronie ekranu) pojawi się zaimportowana biblioteka.



Ostatnią czynnością, którą trzeba wykonać jest zmiana nazwy pakietu w pliku `StdDraw.java` na nazwe pakietu w naszym projekcie, co wykazywane jest po zaimportowaniu przez Eclipse jako błąd



W tym celu wystarczy kliknąć podwójnie lewym klawiszem myszy w nazwę biblioteki widoczną w oknie Eksploratora pakietów, a następnie raz lewym klawiszem myszy w miały znak x na czerwonym tle w oknie, a następnie wybrać proponowaną opcję Add package declaration `'stddrawtest'` .



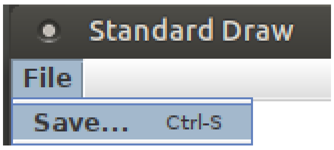
Podstawowe umiejętności

Korzystanie z `StdDraw` jest bardzo proste. Aby wyświetlić (pusty) interfejs graficzny wystarczy w kodzie źródłowym metody main klasy StdDrawTest wpisać jedno, proste polecenie, które wygeneruje standardowe, graficzne okienko.

```
StdDraw.show();
```

Okienko to będzie wyświetlane na ekranie tak długo, aż nie zostanie zamknięte poprzez kliknięcie lewym klawiszem myszy w krzyżyk znajdujący się w jego prawym górnym rogu.

Ciekawą i praktyczną właściwością tego okienka jest to, że jego zawartość można w każdej chwili zapisać do pliku graficznego. W tym celu wystarczy z menu tekstowego znajdującego się w jego górnej części po lewej stronie wybrać opcję File, a następnie Save.



Okienko domyślnie ma rozmiar 512 na 512 pikseli, a punkty na ekranie mają współrzędne liczone od lewego dolnego rogu do prawego górnego. Domyślnie zarówno skala pozioma, jak i pionowa ma zakres od 0 do 1 , ale w razie potrzeby można ją w każdej chwili zmienić.

Rysujemy!

Dobrym punktem wyjścia w korzystaniu z biblioteki `StdDraw` jest narysowanie pojedynczego punktu. W tym celu należy użyć metody `StdDraw.point(double x, double y)` . Deklaracja ta (określana także mianem interfejsu) oznacza, że podczas wywołania metody point z klasy StdDraw należy podać dwa argumenty typu zmiennoprzecinkowego określającego wartość dwóch współrzędnych punktu.

Czyli np. w celu narysowania punktu w środku ekranu należy napisać

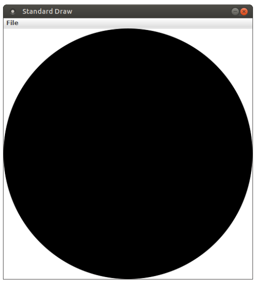
```
StdDraw.point(0.5, 0.5);
```

Narysowany punkt może być trudny do zauważenia, ponieważ jest bardzo mały.

Rysunki w okienku wykonuje się piórem o standardowej grubości, którą w każdej chwili można zmienić za pomocą polecenia `StdDraw.setPenRadius(double raduis).`

```
StdDraw.setPenRadius(1);
StdDraw.point(0.5, 0.5);
```

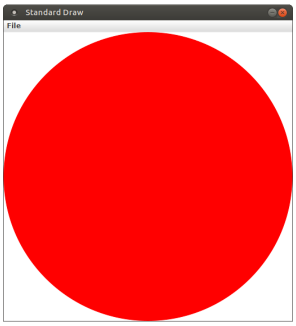
Wykonanie powyższych poleceń spowoduje narysowanie punktu o wielkości całego okna. Tak więc określenie radius należy rozumieć raczej jako średnicę pióra, a nie jego promień.



Kolor pióra zmieniamy za pomocą innego polecenia `StdDraw.setPenColor(StdDraw.RED)`, gdzie kolor w najprostszy sposób podaje się z predefiniowanej palety zapisywanej jak w powyższym przykładzie jako `StdDraw.NazwaKoloru` . Na przykład punkt o średnicy 1 i w kolorze czerwonym znajdujący się na środku ekranu narysujemy za pomocą poniższych poleceń.

```
StdDraw.setPenRadius(1);
StdDraw.setPenColor(StdDraw.RED);
StdDraw.point(0.5, 0.5);
```

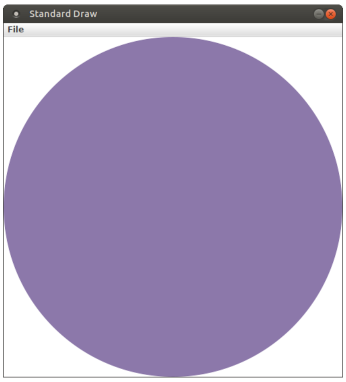
Efekt jest zgodny z oczekiwaniem.



Kolory można też deklarować zupełnie dowolnie korzystając z przestrzeni barw RGB — czerwony, zielony, niebieski — w której każdy kolor jest złożony z trzech podstawowych kolorów, a jego intensywność określona jest na skali od 0 do 255. Na przykład można zmienić kolor pióra na taki kolor:

```
StdDraw.setPenRadius(1);
StdDraw.setPenColor(140,120,170);
StdDraw.point(0.5, 0.5);
```

Efekt jest taki "miękki" (?) fiolet.



W ten sposób można też generować losowe kolory rysowanych elementów. Załóżmy, że chcemy wygenerować 20 kolorowych punktów o losowych współrzędnych. W tym celu piszemy:

```
Random rnd = new Random();
int r, g, b;

StdDraw.setPenRadius(0.08);

for (int i = 0; i < 20; i++){

     r = rnd.nextInt(256);
     g = rnd.nextInt(256);
     b = rnd.nextInt(256);

     StdDraw.setPenColor(r, g, b);

```

```
    StdDraw.point(Math.random(), Math.random());

}
```

A efekt jest taki, jaki miał być.