Spis treści

Temat: Aplety.
Celem wykładu jest zdefiniowanie sieciowej aplikacji Java^{TM}, zwanej apletem i opis budowy apletu. Aplet można połączyć z "main". Taki program może działać jako aplet na stronie WWW i samodzielnie jako aplikacja.

1 Aplet

Aplet jest nazwą zapożyczoną z języka angielskiego. Jest to rodzaj programu napisanego w Java^{TM} i wykonywanego na stronie WWW w przeglądarce sieciowej, takiej jak InternetExplorer lub Netscape. Ze względu na specyfikę srodowiska, w którym działa aplet, jego struktura różni sie znacznie od struktury samodzielnej aplikacjii Java^{TM}. Zazwyczaj, kilka metod kontroluje zachowanie apletu. Prócz tego należy go osadzić w środowisku graficznym przeglądarki. Służą do tego specjalne znaczniki języka opisu strony HTML.

2 Od aplikacji do apletu

Zamienimy pierwszy program z poprzedniego wykładu na aplet. Wygląda on nastepujaco.

```
// ---------------------------------------------------------------
/*
Fizyka komputerowa, IV, 2001.  Java(TM).
Program #1.
A. Baran, IFiz UMCS, 2000.
http://tytan/umcs.lublin.pl/baran
*/
// ---------------------------------------------------------------
//
```

```
import java.applet.Applet;
import java.awt.Graphics;
import java.awt.Color;

public class ProstyApplet extends Applet{

  String text;

  public void init() {
        text = "Prosty aplet";
        setBackground(Color.cyan);
  }

  public void start() {
        System.out.println("start...");
  }

  public void stop() {
        System.out.println("stop...");
  }

  public void destroy() {
        System.out.println("zakończ...");
  }

  public void paint(Graphics g){
        System.out.println("Paint");
        g.setColor(Color.blue);
        g.drawRect(0, 0,
                   getSize().width -1,
                   getSize().height -1);
        g.setColor(Color.red);
        g.drawString(text, 15, 25);
  }
}
```

Klasa `ProstyApplet` jest klasą publiczną. Przeglądarka lub `appletviewer` może go uruchomic. Aby to zrobić osadzimy aplet w środowisku `WWW`. W tym celu utworzymy plik `ProstyApplet.htm` lub `ProstyApplet.html` dla przeglądarki sieciowej i wpiszemy w nim następujący tekst

```
<HTML>
<BODY>
<APPLET CODE=ProstyApplet.class WIDTH=200 HEIGHT=100>
</APPLET>
</BODY>
</HTML>
```

Po kompilacji programu `ProstyApplet.java`:

```
javac ProstyApplet.java
```

uruchomiamy przeglądarkę `InternetExplorer` lub `Netscape` na pliku `ProstyApplet.htm`. Zobaczymy coś takiego co przedstawia rysunek niżej.

Prosty aplet w przeglądarce

Możemy zrobić to inaczej, uruchamiając program `appletviewer` - program do przeglądania appletów:

```
appletviewer ProstyApplet.htm
```

W tym drugim przypadku, na ekranie zobaczymy okno

Prosty aplet w oknie systemu WINDOWS

Wynik ten jest inny od apletu oglądanego w przeglądarce `WWW`, ale własnie o to chodzi. W przeglądarce widzimy tylko samą grafikę apletu, natomiast w środowisku `WINDOWS`, aplet jest w oknie podobnym do innych, posiada `menu Applet`, z którego można nim sterować oraz typowe przyciski kasujące, opuszczające i powiększające okno. Wymiary grafiki apletu są w obu przypadkach jednakowe (u nas 200px X 100px). W środowisku `WWW` apletem steruje przeglądarka, w środowisku `WINDOWS` - system operacyjny komputera wraz z JVM.

Uwaga! Jeśli przeglądarka nie obsługuje apletów Java^{TM} należy zainstalować tzw. wtyczkę Java^{TM}(plug-in). Można ja pobrac ze strony .

3 Budowa apletu

Aplet z przykładu różni się znacznie od aplikacji z poprzednich zajęć. Po pierwsze pojawiły się polecenia

```
import java.applet.Applet;
import java.awt.Graphics;
import java.awt.Color;
```

Pozwalają one dołączyć do naszego programu istniejące już klasy z bibliotek Java^{TM}. W tym przypadku importujemy klasy `java.applet.Applet, java.awt.Graphics` oraz `java.awt.Color`. Pozwalają one wykonywać różne operacje graficzne na aplecie.

Nasza klasa `ProstyApplet` pochodzi w prostej linii od klasy `Applet`. Wynika to z deklaracji `extends`, która oznacza, że nasza klasa *rozszerza* klasę `Applet`.

W środowisku programowania obiektowego, a takim jest Java^{TM}, oznacza to, że klasa `ProstyApplet` dziedziczy, a więc i posiada wszystkie te pola i metody, które posiada oryginalna, wzorcowa klasa `Applet`. Klasę `ProstyApplet` nazywamy *potomkiem* klasy `Applet`, która jest z kolei *przodkiem* naszej klasy.

Ponieważ klasa `Applet` dziedziczy po klasach `Object, Container` i `Panel`, to klasa `ProstyApplet` jest również spadkobiercą wypisanych klas.

Struktura klasowa programu jest następująca

```
Object
  |
  +---> Container
           |
           +---> Panel
                    |
                    +---> Applet
                             |
                             +---> ProstyApplet
```

Klasa `Object` jest *najstarszą* w hierarchii klasą Java^{TM}. Od niech wywodzą się wszystkie klasy wraz ze wszystkimi polami i metodami.

W naszym przykladzie metody "deklarowane"w klasie `ProstyApplet`, a więc `init, start, stop, destroy` oraz metoda `paint` są również metodami klasy `Applet`. Te "deklaracjeńazywa się rozszerzaniem ((*ang.* overriding)). W tym sensie zamieniamy istniejące już metody nowymi metodami, które ćośżobią w odróżnieniu od wzorcowych metod klasy `Applet`. To rozszerzanie metod jest ważną i potrzebną cechą Java^{TM}. Można w ten sposób korygować istniejące metody.

Klasa `Applet` posiada metody `init, start, stop, destroy` oraz metoda `paint`. Sterują one pracą apletu. Metoda `init` inicjuje aplet, `start` - uruchamia go, `stop` - zatrzymuje, `destroy` - niszczy go, a metoda `paint`, której oryginał znajduje się w klasie `Container`, *maluje* aplet. Prócz tego mamy w klasie `Applet` metodę `update`, która go odświeża.

4 Cykl życia apletu

Cykl życia apletu wygląda następująco. Środowisko programu `appletviewer` lub przeglądarka, zapoczątkowuje aplet - metoda `init`. Dalej wywoływana jest metoda `start`. Kiedy zmienimy stronę w przeglądarce wówczas aplet jest zatrzymywany metodą `stop`. W chwili powrotu na stronę z apletem, aplet startuje jeszcze raz (`start`) i następnie uruchamia się metoda `paint`. W momencie gdy przeglądarka jest zatrzymana, aplet jest niszczony metodą `destroy`. Wywołanie metody `start` jest początkiem życia apletu. Cały proces jest wątkiem, który dotyczy apletu.

Po uruchomieniu, nasz aplet wypisze w oknie gdzie został uruchomiony komunikaty z metod `start, paint, ....` Dostaniemy coś takiego:

```
start...
Paint
Paint
Paint
stop...
start...
Paint
Paint
Paint
```

```
Paint
stop...
zakoncz...
```

Widzimy, że metoda `paint` uruchamiana jest najczęściej, co jakiś czas. Jest to więc proces, wątek, który wciąż działa. Nie jest to tak, że aplet raz namalowany pozostaje tym samym. Jest on wciąż odnawiany. Można to wykorzystać. Wątek związany z apletem jest jednym z możliwych wątków, które można utworzyć w Java^{TM}. Pozwala to na symulacje zachowań dynamicznych różnych obiektów, np. na symulacje procesów. To jest nasz cel.

Sprawdźmy to dodając do naszego apletu polecenie rysowania przypadkowego prostokąta. Prostokąt można namalować metodą `fillRect(x0,y0,x1,y1` z klasy Graphics. Zauważmy, że zmienna `g` występująca w nagłówku metody `paint` jest właśnie egzemplarzem klasy `Graphics`. Wobec tego polecenie

```
g.fillRect(...)
```

namaluje rzeczony prostokąt. Jeszcze tylko musimy podać współrzędne jego wierzchołków: górnego-lewego (x0,y0) i dolnego-prawego (x1,y1). Ponieważ chcemy by były to położenia przypadkowe, użyjemy metody `Math.random()` by wygenerować te położenia. Należy przy tym pamiętać, że `random()` generuje liczby typu `float`. W procedurze `fillRect(...)` należy je skonwertowac do typu `int`. Wyglada to mniej więcej tak:

```
int x0, y0, x1, y1;
...
//// inne polecenia
...
x0 = (int)(Math.random(50));
y0 = (int)(Math.random(50)) + 40;
x1 = (int)(Math.random(50));
y1 = (int)(Math.random(50)) + 40;

g.fillRect(x0,y0,x1,y1);
```

Kiedy aplet znika, np. chowa sie pod innym oknem, a następnie pojawia się, zmienia się jego wygląd. Nie jest to jeszcze zbyt ciekawe, ale się poprawimy. Wszystko jeszcze przed nami. Chodzi nam tylko o to by zauważyć, że aplet *żyje*.

5 Aplet jako aplikacja

Po niewielkich zmianach aplet może pracowac jako samodzielna aplikacja. Takie programy można uruchamiać zarówna na stronach `WWW` jak i z użyciem polecenia `java` bezpośrednio w systemie operacyjnym.

```
// ---------------------------------------------------------------
/*
Fizyka komputerowa, IV, 2001.  Java(TM).
Program #2.
A. Baran, IFiz UMCS, 2000.
```

```
http://tytan/umcs.lublin.pl/baran
*/
// ---------------------------------------------------------------------
//

/*
 * Program samodzielny i applet w jednym.
 */
import java.awt.Graphics;
import java.awt.Frame;
import java.applet.Applet;

public class ApiApl extends Applet {

  public void init() {
    resize(200,60);
  }
  public void paint(Graphics g) {
    g.drawString("Witajcie ...", 60, 30);
  }

  public static void main(String[] args) {
    System.out.println("... w main");

    ApiApl h = new ApiApl();
    h.init();

    Frame f = new Frame("Aplet i Aplikacja");
    f.resize(200, 30);

    f.add("Center", h);
    f.show();
  }
}
```

Program działa teraz wszędzie. Można go dodatkowo zmodyfikować, dodając na początku dwie linie kodu w języku opisu strony HTML i opatrując je komentarzem Java^{TM}.

```
<APPLET CODE=ApiApl.class WIDTH=200 HEIGHT=60>
</APPLET>
```

Teraz możemy uruchomić program na trzy sposoby. Zadanie 1.

Jak to zrobić?