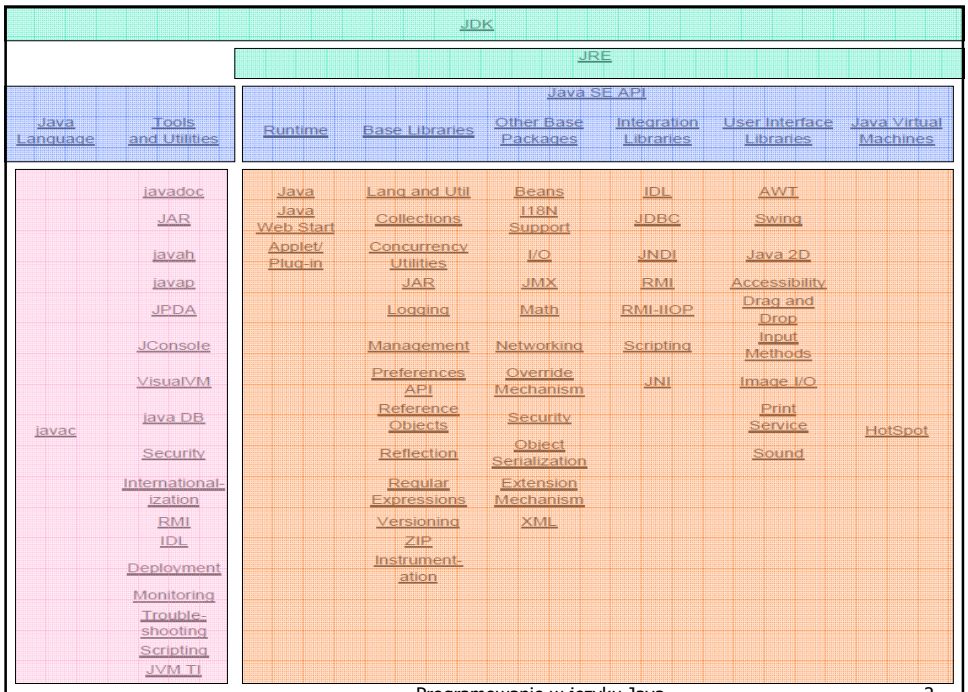
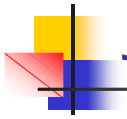
Programowanie w języku Java

Wykład 11: Java Platform, Standard Edition (Java SE)





JVM

- Client VM
  - java –client …
  - Szybszy start
  - Dla aplikacji klienckich
- Server VM
  - java –server …
  - Lepsza wydajność
  - Dla aplikacji serwerowych

Oracle Rockit JDK

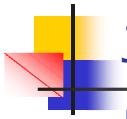
- Zoptymalizowane dla architektur Intel
- Zawiera Oracle Rockit JVM
- Zalecane dla Oracle WebLogic Server

Java SE – pakiety UI (1)

- AWT
  - Biblioteka dla GUI (w tym metody natywne)
- Swing
  - API dla GUI
- Java 2D
  - Grafika 2D
- Accesibility
  - MSWindows assistive technology w Javie (interfejsy dla osób niepełnosprawnych)
  - Java Access Bridge
  - Java Accessibility API (JAAPI)

Java SE – pakiety UI (2)

- Drag and Drop
  - Mechanizm transferu danych pomiędzy komponentami GUI (implementacja w AWT i Swing)
- Input Methods
  - Wprowadzanie danych tekstowych w różnych językach lub z różnych źródeł np. pismo ręczne, głosowo
- Image I/O
  - Wprowadzanie/wyproiwadzanie danych w formatach graficznych (JPEG, BMP, FIF, PNG)
- Print Service
  - API dla programowania obsługi drukarek
- Sound



Java SE – pakiety UI (3)

- Sound
  - Przechwytywanie, przetwarzanie i odtwarzanie dźwięków
  - Obsługa formatów audio AIFF, AU, WAV
  - Obsługa formatów plików muzycznych MIDI, RMF
  - Miksowanie i synteza audio

Java SE – pakiety dla aplikacji rozproszonych

- Java IDL
  - Implementacja standardu CORBA
- Java JDBC
  - Java Database ConnectivityTechnology
- Java JNDI
  - Java Naming and Directory Interface
- RMI
  - Biblioteki i narzędzia dla RMI
- RMI-IIOB
  - RMI dla standardu CORBA
- Scripting
  - Narzędzie jrunscript
- JNI
  - Java Native Interface - inrterfejs dla metod natywnych

Java SE–inne pakiety standardowe (1)

- Java Beans Component API
  - Biblioteka do zarządzania komponentami
- Java Internatonalization (i18n) Support
  - Obsługa wersji narodowych
- I/O
  - Wejście/wyjście
- JMX
  - Java Management Extensions
  - API do zarządzania zasobami (aplikacje, urządzenia, usługi)

Java SE–inne pakiety standardowe (2)

- Networking
  - Pakiety java.net, ssl, HttpServer
- Java Endorsed Standards Overriding Mechanizm
  - Mechanizm aktualizacji technologii „zewnętrznych”
- Security
  - Java Authentication and Authorization Service (JAAS)
  - Java Secure Socket Extension (JSSE)
  - Java Generic Security Services (Java GSS-API)
  - Public Key Infrastructure (PKI)
  - Simple Authentication and Security Layer (SASL)

Java SE–inne pakiety standardowe (3)

- Object serialization
  - Serializacja obiektów (pakiet java.io)
- Extension Mechanism
  - Mechanizm rozszerzania JVM (np.. Metoda ładowania klas)
- XML
  - The Java™ API For XML Processing (JAXP)
  - The Java Architecture For XML Binding (JAXB)
  - The Java API for XML Web Services (JAX-WS)

Java SE – pakiety podstawowe(1)

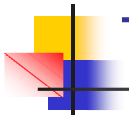
- Lang i util
- The Collections Framework
- Java Concurrency Utilities
- Java Archive (JAR) Files
  - Java.util.jar
  - Java.net.jarURLConnection
- Java Logging Technology
  - Mechanizm raportowania (log)
  - Java.util.logging

Java SE – pakiety podstawowe(2)

- Monitoring and Management for the Java Platform
  - Monitorowanie i zarządzanie JVM oraz SO
- Core Java Preferences API
  - java.util.prefs
  - Zarządzanie preferencjami użytkownika i aplikacji
- Pakiet java.lang.ref
  - Operowanie na wskaźnikach do obiektów
- Java Reflection API
  - java.lang.reflect
- Wyrażenia regularne
  - java.util.regex
  - Definiowanie i dopasowywanie wzorców
- Package Version Identification
  - zarzadzanie wersjami pakietów. JRE i VM

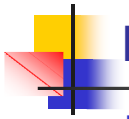
Java SE – pakiety podstawowe(3)

- Kompresja danych ZIP i GZIP
  - java.util.zip
- Pakiet java.lang.instrument
  - Narzędzia do instrumentalizacji aplikacji np. monitorowanie, badanie wydajności.



Technologia Java Web Start

- Uruchamianie aplikacji z poziomu przeglądarki internetowej lub systemu operacyjnego
- JNLP – Java Network Launching Protocol



Desktop Java – pakiety opcjonalne

- Java Media Framework (JMF)
- Java 3D API
- Java Advanced Imaging API (JAI)
- Java Speech API
- Java Communications API
- Java Mail
- Java Binding for OpenGL
- Java Telephony API

Przykład: Aplet – media player

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.lang.String;
import java.net.URL;
import java.net.MalformedURLException;
import java.io.IOException;
import java.util.Properties;
import javax.media.*;

public class SimplePlayerApplet extends Applet
implements ControllerListener {
    Player player = null;
   Component visualComponent = null;
   Component controlComponent = null;
   Component progressBar = null;
   boolean firstTime = true;
   long CachingSize = 0L;
   Panel panel = null;
   int controlPanelHeight = 0;
   int videoWidth = 0;
   int videoHeight = 0;
```

```
public void init() {
 setLayout(null);
 setBackground(Color.white);
 panel = new Panel();
 panel.setLayout( null );
 add(panel);
 panel.setBounds(0, 0, 320, 240);
 String mediaFile = null;
 MediaLocator mrl = null;
 URL url = null;
 if ((mediaFile = getParameter("FILE")) == null)
     Fatal("Invalid media file parameter");
 try {
    url = new URL(getDocumentBase(), mediaFile);
    mediaFile = url.toExternalForm();
  } catch (MalformedURLException mue) {
 }
```


media player, cd.

```
try {
  if ((mrl = new MediaLocator(mediaFile)) ==
                              null)
     Fatal("Can't build URL for " + mediaFile);
     try {
       player = Manager.createPlayer(mrl);
     } catch (NoPlayerException e) {
               System.out.println(e);
               Fatal("Could not create player for „
                              + mrl);
     }
  player.addControllerListener(this);
} catch (MalformedURLException e) {
         Fatal("Invalid media file URL!");
} catch (IOException e) {
         Fatal("IO exception creating player for „
           + mrl);
}
}
```

```
public void start() {
   if (player != null)
       player.start();
}

public void stop() {
 if (player != null) {
         player.stop();
         player.deallocate();
      }
}

public void destroy() {
    player.close();
}
```

media player, cd.

```
public synchronized void controllerUpdate
        (ControllerEvent event) {
 if (player == null)   return;
 if (event instanceof RealizeCompleteEvent) {
   if (progressBar != null) {
      panel.remove(progressBar);
      progressBar = null;
   }
   int width = 320;
   int height = 0;
   if (controlComponent == null)
     if (( controlComponent =
       player.getControlPanelComponent()) !=
       null) {
        controlPanelHeight = controlComponent.
                        getPreferredSize().height;
        panel.add(controlComponent);
        height += controlPanelHeight;
       }
```

```
if (visualComponent == null)
       if (( visualComponent =
         player.getVisualComponent())!= null) {
           panel.add(visualComponent);
           Dimension videoSize=
            visualComponent. getPreferredSize();
          videoWidth = videoSize.width;
          videoHeight = videoSize.height;
          width = videoWidth;
          height += videoHeight;
          visualComponent.setBounds(0, 0,
                    videoWidth, videoHeight);
       }
 panel.setBounds(0, 0, width, height);
 if (controlComponent != null) {
  controlComponent.setBounds(0, videoHeight,
            width, controlPanelHeight);
   controlComponent.invalidate();
 }
}
```


media player, cd.

```
else if (event instanceof CachingControlEvent){
   if (player.getState() > Controller.Realizing)
                          return;
CachingControlEvent e =(CachingControlEvent)
                          event;
CachingControl cc = e.getCachingControl();
if (progressBar == null) {
  if ((progressBar = cc.getControlComponent())
                != null) {
    panel.add(progressBar);
    panel.setSize(progressBar.
                getPreferredSize());
    validate();
  }
 }
} else if (event instanceof EndOfMediaEvent) {
    player.setMediaTime(new Time(0));
    player.start();
}
```

```
else if (event instanceof ControllerErrorEvent) {
   player = null;
   Fatal(((ControllerErrorEvent)event).
            getMessage());
}else if (event instanceof ControllerClosedEvent)
{   panel.removeAll();
}
}

void Fatal (String s) {
  System.err.println("FATAL ERROR: " + s);
  throw new Error(s);
  }
}
```

Formatowanie liczb (1)

- public PrintStream format(String format, Object... args)
  public PrintStream printf(String format, Object... args)
- public PrintStream format(Locale l, String format, Object... args)
  public PrintStream printf(Locale l, String format, Object... args)

Znaki konwersji:

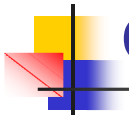
d - liczba całkowita, f – liczba rzeczywista, n - \n ,
tB - pełna nazwa miesiąca, td - dzień miesiąca 2-cyfrowa liczba,
te – dzień miesiąca 1 lub 2-cyfrowa liczba,
ty – rok (2cyfry), tY – rok (4 cyfry), tl – godz.(0-12), tM – min,
tp – am/pm, tm – miesiąc (2 cyfry), tD – data: %tm/%td/%ty

Znaczniki konwersji:

n.m – liczba cyfr, + - ze znakiem, - - wyrównanie do lewej,
0n – z wiodącymi zerami.

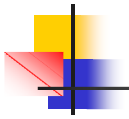
Formatowanie liczb (2)

```
long n = 461012;
System.out.format("%d%n", n);          // --> "461012"
System.out.format("%08d%n", n);        // --> "00461012"
System.out.format("%+8d%n", n);        // --> " +461012"
System.out.format("%,8d%n", n);        // --> " 461,012"
System.out.format("%+,8d%n%n", n); // --> "+461,012"
double pi = Math.PI;
System.out.format("%f%n", pi);                                  // --> "3.141593"
System.out.format("%.3f%n", pi);                                // --> "3.142"
System.out.format("%10.3f%n", pi);                              // --> " 3.142"
System.out.format("%-10.3f%n", pi);                             // --> "3.142"
System.out.format(Locale.FRANCE, "%-10.4f%n%n", pi);    // --> "3,1416"
Calendar c = Calendar.getInstance();
System.out.format("%tB %te, %tY%n", c, c, c);   // --> "May 29, 2006"
System.out.format("%tl:%tM %tp%n", c, c, c);    // --> "2:34 am"
System.out.format("%tD%n", c);                  // --> "05/29/06"
```



Ciągi znaków

- String()
  - niemodyfikowalny
- StringBuilder()
  - z metodami: append(), insert(), delete(), replace(), inverse(), itp.
- StringBuffer()
  - j/w ale synchronizowany



Koniec