Аплети

Аплетът се реализира като наследник на класа JApplet**.**

Жизнен цикъл на аплет

1. **Зареждане на аплет в браузъра**
   - **създаване на екземпляр на класа, управляващ аплета;**
   - **инициализиране на аплета;**
   - **стартиране на аплета.**
2. **Напускане на страницата с аплета и отново връщане към страницата**
   - **спиране на аплета;**
   - **стартиране на аплета.**
3. **Презареждане на аплет от някои браузъри**
   - **спиране на аплета;**
   - **разрушаване на аплета (почистване на ресурсите);**
   - **създаване на екземпляр;**
   - **инициализиране на аплета;**
   - **стратиране на аплета.**
4. **Напускане (изход) от браузъра**
   - **спиране на аплета;**
   - **разрушаване на аплета.**

Клас JApplet (javax.swing)

Използва се за изпълнение на програма във Web страница.

public JApplet()
Конструира нов екземпляр на аплет.

public void init()
Инизиализира аплета. Извиква се при зареждането на аплета.

public void start()
Стартира аплета. Извиква се след инициализирането на аплета.

public void stop()
Спира аплета. Извиква се след деактивирането на аплета.

public Container getContentPane()
Връща съдържанието на аплета.

public void setContentPane(Container contentPane)
Установява съдържанието на аплета.

public Component add(Component comp)
Добавя компонтата comp **към аплета.**

public void paint(Graphics gwin)
Изчертава аплета с графично съдържание gwin**.**

public void repaint()
Пречертава текущия аплет.

public Image getImage(URL url, String name)
Възстановява изображение от определения URL (унифициран локатор на ресурси).

public URL getCodeBase()
Връща URL, на който се намира байткодът на аплета.

Клас Color (java.awt)

Използва се за представяне на цветовете. Цветът се състои от три компоненти: червена, зелена и синя – всяка със стойност от 0 до 255.

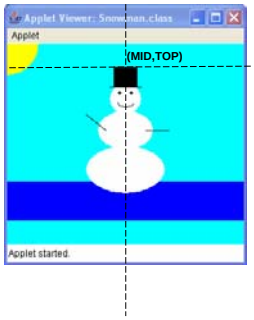
Цвят	Обект	RGB стойност
черен	Color.BLACK	0,0,0
син	Color.BLUE	0,0,255
синьозелен	Color.CYAN	0,255,255
сив	Color.GRAY	128,128,128
тъмно сив	Color.DARK_GRAY	64,64,64
светло сив	Color.LIGHT_GRAY	192,192,192
зелен	Color.GREEN	0,255,0
пурпурен	Color.MAGENTA	255,0,255
оранжев	Color.ORANGE	255,200,0
розов	Color.PINK	255,175,175
червен	Color.RED	255,0,0
бял	Color.WHITE	255,255,255
жълт	Color.YELLOW	255,255,0

public void setBackground(Color color)
Установява фоновия цвят.

public void setSize(int width, int height)
Установява размера на аплета: ширина width **и височина** height**.**

public void setColor(Color color)
public void getColor(Color color)
Установява/връща цвета на графиката (методи на класа Graphics**).**

Пример: **Изчертава снежен човек.**




```
import java.awt.*;
import javax.swing.*;
public class Snowman extends JApplet {
   private final int MID = 150;
   private final int TOP = 50;
   public void init() {
     setSize(300, 255);
   }
   public void paint(Graphics page) {
     setBackground(Color.CYAN);                    // фон
     page.setColor(Color.BLUE);
     page.fillRect(0,175,300,50);                  // земя
     page.setColor(Color.YELLOW);
     page.fillOval(-40,-40,80,80);                 // слънце
     page.setColor(Color.WHITE);
     page.fillOval(MID-20,TOP,40,40);              // глава
     page.fillOval(MID-35,TOP+35,70,50);           // горен торс
     page.fillOval(MID-50,TOP+80,100,60);          // долен торс
     page.setColor(Color.BLACK);
     page.fillOval(MID-10,TOP+10,5,5);             // ляво око
     page.fillOval(MID+5,TOP+10,5,5);              // дясно око
     page.drawArc(MID-10,TOP+20,20,10,190,160);    // усмивка
     page.drawLine(MID-25,TOP+60,MID-50,TOP+40);   // лява ръка
     page.drawLine(MID+25,TOP+60,MID+55,TOP+60);   // дясна ръка
     page.drawLine(MID-20,TOP+5,MID+20,TOP+5);     // периферия
     page.fillRect(MID-15,TOP-20,30,25);           // шапка
   }
}
```

```
<HTML>
<HEAD>
<TITLE> Снежен човек</TITLR>
</HEAD>
<BODY>
<APPLET CODE="Snowman.class" WIDTH=300 HEIGHT=225>
</APPLET>
</BODY>
</HTML>
```

Събития и слушатели

Събития

Събитие е обект, който представя интересна случка за потребителя.

Примери:

- **натискане бутон на мишката;**
- **натискане клавиш от клавиатурата;**
- **натискане на графичен бутон или плъзгач, генерирани от графичния потребителски интерфейс (GUI).**

Програмата трябва за разпознае и да отговори на събитията.

Клас EventObject **(**java.util**)**

Супер клас, чиито наследници представят събития.

public Object getSource()

Връща обекта-източник на събитието.

Слушатели

Слушател е обект, който чака да се случи събитие и да генерира подходящ отговор.

I. Създаване на обект-слушател за даденото събитие;

1. Чрез слушателски интерфейс – предефинират се всички методи на интерфейса

2. Чрез адаптерен клас, реализиращ слушателския интерфейс – предефинират се само необходимите методи на адаптерния клас.

II. Добавяне на слушателя към графичната компонента, която може да генерира събитието.

Когато се случи събитието, автоматично се извиква подходящият метод на слушателя. Методът получава като параметър обект, представящ събитието.

I. Събития, генерирани при използване на мишка – клас MouseEvent **(**java.awt.event**)**

Методи на класа MouseEvent**:**

Point getPoint ()

Връща координатите на мястото, в което се е случило събитието с мишка.

int getX ()

int getY ()

Връща x **/** y **координатата на мястото, в което се е случило събитието с мишка.**

int getClickCount ()

Връща броя на бързите последователни натискания, представени със събитието с мишка.

1. Събития при натискане бутон на мишка

Слушателски интерфейс MouseListener **(**java.awt.event**)**

Адаптерен клас MouseAdapter **(**java.awt.event**) – реализира** MouseListener

public void addMouseListener(MouseListener l)

Добавя слушател на събития с мишка към компонентата.

- **натиснат бутон на мишка надолу;**
  void mousePressed (MouseEvent event)
- **освободен бутон на мишка;**
  void mouseReleased (MouseEvent event)
- **натиснат и освободен бутон на мишката без преместване;**
  void mouseClicked (MouseEvent event)
- **преместване на указателя на мишката над компонента;**
  void mouseEntered (MouseEvent event)
- **преместване на указателя на мишката извън компонентата.**
  void mouseExited (MouseEvent event)

2. Събития с движение на мишка

Слушателски интерфейс MouseMotionListener **(**java.awt.event**)**

Адаптерен клас MouseMotionAdapter **(**java.awt.event**), който реализира** MouseMotionListener

public void addMouseMotionListener (MouseMotionListener l)

Добавя подходящ слушател на събития с движения на мишка към компонентата.

- **преместване на мишка над компонента;**
  void mouseMoved (MouseEvent event)
- **влачене на мишка над компонента.**
  void mouseDragged (MouseEvent event)

II. Събития от клавиатурата – клас KeyEvent **(**java.awt.event**)**

Метод на класа KeyEvent**:**

public int getKeyCode()

Връща кода на натиснатия клавиш.

Слушателски интерфейс KeyListener **(**java.awt.event**)**

Адаптерен клас KeyAdapter **(**java.awt.event**), който реализира** KeyListener

public void addKeyListener (KeyListener l)

Добавя слушател на събития от клавиатурата.

1. **Натиснат клавиш надолу** – KEY_PRESSED:
   - VK_LEFT – **стрелка наляво;**
   - VK_RIGHT – **стрелка надясно;**
   - VK_UP – **стрелка нагоре;**
   - VK_DOWN – **стрелка надолу.**

   void keyPressed (KeyEvent l)
2. **Освободен клавиш** – KEY_RELEASED

   void keyReleased (KeyEvent l)
2. **Натиснат символ – натиснат клавиш или клавишна комбинация, произвеждаща символ** KEY_TYPED

   void keyTyped (KeyEvent l)

```
// Аплет – събитие „натискане надолу бутона на мишката“
// (метод mouseClicked)

// Директно реализира всички методи на интерфейса MouseListener.
// Аплетът е слушател на събитието.
public class <име_на_клас> extends JApplet implements MouseListener {
  ...
  <някакъв_обект>.addMouseListener (this);
  ...
  public void mouseClicked (MouseEvent e) {
    // реализира слушател на събитието
    ...
  }
  // Празни дефиниции на методи
  public void mousePressed (MouseEvent e) { }
  public void mouseReleased (MouseEvent e) { }
  public void mouseEntered (MouseEvent e) { }
  public void mouseExited (MouseEvent e) { }
}
```

Аплетът е слушател на събитието

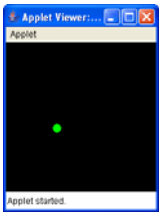
```
// Аплет – събитие „натискане надолу бутона на мишката“
// (метод mouseClicked)

// Слушател на събитието е вътрешен клас, който наследява адаптерния
// клас MouseAdaptaer и предефинира само метода mouseClicked.
public class <име_на_клас> extends JApplet {
  ...
  <някакъв_обект>.addMouseListener (new MyAdapter());
  ...
  // вътрешен клас
  class MyAdapter extends MouseAdapter {
    public void mouseClicked (MouseEvent e) {
      // реализира слушател на събитието
      ...
    }
  }
}
```

```
// Аплет – събитие „натискане надолу бутона на мишката“
// (метод mouseClicked)

// Слушател на събитието е анонимен вътрешен клас, който наследява
// адаптерния клас MouseAdapter и предефинира само метода
// mouseClicked.
public class <име_на_клас> extends JApplet  {
  ...
  <някакъв обект>.addMouseListener(new MouseAdapter() {
    public void mouseClicked(MouseEvent e) {
      // реализира слушател на събитието
      ...
    }
  });
  ...
}
```

Пример: Аплет – изчертава зелена точка при натискане бутона на мишката (събитие «натиснат бутон на мишката надолу» MouseEvent **чрез анонимен клас, който реализира метода** mousePressed **на адаптерния клас** MouseAdapter**).**

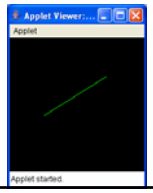


Клас Point **(**java.awt**)**

Представя точка с координати (x, y).

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class Spot extends JApplet {
  private Point clickPoint = null;
  private static final int RADIUS = 6;
  public void init () {
    addMouseListener (new MouseAdapter () {
      public void mousePressed (MouseEvent event) {
        clickPoint = event.getPoint();     // точка, в която е натисната мишката
        repaint();
      }
    });
    setBackground (Color.BLACK);
  }
  public void paint (Graphics page) {
    page.clearRect(0, 0, getWidth(), getHeight());
    page.setColor (Color.green);
    if (clickPoint != null)                    // изчертава кръг
      page.fillOval
            (clickPoint.x-RADIUS,clickPoint.y-RADIUS,RADIUS*2,RADIUS*2);
    }
}
```

Пример: Аплет – изчертава зелена гъвкава линия чрез натискане и влачене бутона на мишката. Началната точка се определя от събитието „натискане бутона на мишката надолу“ MouseEvent **(чрез анонимен клас, който реализира метода** mousePressed **на адаптерния клас** MouseAdapter**). Крайната точка се определя от събитието „влачене на мишка“** MouseEvent **(чрез анонимен клас, който реализира метода** mouseDragged **на адаптерния клас** MouseMotionAdapter**), пречертава се всеки път и линията се „разтяга“.**



```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class RubberLine extends JApplet {
  private Point point1 = null;      // начална точка
  private Point point2 = null;      // крайна точка
  private static final int APPLET_WIDTH = 200;          // широчина на аплета
  private static final int APPLET_HEIGHT = 200;         // височина на аплета
  public void init () {
    setBackground (Color.BLACK);                        // фонов цвят
    setSize (APPLET_WIDTH, APPLET_HEIGHT);   // размер на аплета
    addMouseListener (new MouseAdapter() {
      public void mousePressed (MouseEvent event) {
        point1 = event.getPoint();       // точка, в която е натисната мишката
      }
    });
```

```
    addMouseMotionListener (new MouseMotionAdapter () {
      public void mouseDragged (MouseEvent event) {
        point2 = event.getPoint();          // точка при влачене на мишката
        repaint();
      }
    });
  }
  public void paint (Graphics page) {
    page.clearRect(0, 0, getWidth(), getHeight());
    page.setColor (Color.GREEN);
    if (point1 != null && point2 != null)     // изчертава линия
      page.drawLine (point1.x, point1.y, point2.x, point2.y);
  }
}
```

Пример: Събития от клавиатура. Аплет – мести изображението на стрелка при натискане на клавиш-стрелка от клавиатурата. Използва събитието „натискане клавиш от клавиатурата“ KeyEvent **(чрез анонимен клас, който реализира метода** keyPressed **на адаптерния клас** KeyAdapter **– определя кой клавиш е натиснат, като извиква метода** getKeyCode**).**

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class Direction extends JApplet {
  private final int APPLET_WIDTH = 200;
  private final int APPLET_HEIGHT = 200;
  private final int JUMP = 5; // нарастване за преместване на изображението
  private final int IMAGE_SIZE = 31;
  private Image up, down, right, left, currentImage;
  private int x, y;          // координати на изображението
  public void init() {
    x = y = 0;               // установява координатите на изображението
    // Зарежда изображенията на стрелките
    up = getImage(getCodeBase(), "arrow_up.gif");
    down = getImage(getCodeBase(),"arrow_down.gif");
    left = getImage(getCodeBase(), "arrow_left.gif");
    right = getImage(getCodeBase(), "arrow_right.gif");
    currentImage = right;                          // установява дясна стрелка,
    setBackground (Color.black);                   // черен фон и
    setSize (APPLET_WIDTH, APPLET_HEIGHT);         // размера на аплета
```

```
    // Слушател на събития от клавиатурата – анонимен вътрешен клас
    addKeyListener (new KeyAdapter() {
      public void keyPressed (KeyEvent event) {
        switch (event.getKeyCode()) {
        case KeyEvent.VK_UP:  currentImage = up;        // стрелка нагоре
                              if (y > 0) y -= JUMP;
                              break;
        case KeyEvent.VK_DOWN: currentImage = down; // стрелка надолу
                                if (y <APPLET_HEIGHT-IMAGE_SIZE)
                                  y += JUMP;
                                break;
        case KeyEvent.VK_LEFT: currentImage = left;     // стрелка наляво
                              if (x > 0) x -= JUMP;
                              break;
        case KeyEvent.VK_RIGHT: currentImage = right; // стрелка надясно
                                if (x < APPLET_WIDTH-IMAGE_SIZE)
                                  x += JUMP;
                                break;
        }
        repaint();
      }
    });
  }
```

```
  // Изчертава текущото изображение в текущото местоположение.
  public void paint (Graphics page) {
    page.clearRect(0, 0, getWidth(), getHeight());
    page.drawImage (currentImage, x, y, this);
  }
}
```