

СЪБИТИЯ

ЛЕКЦИОНЕН КУРС “ПРОГРАМИРАНЕ НА JAVA”



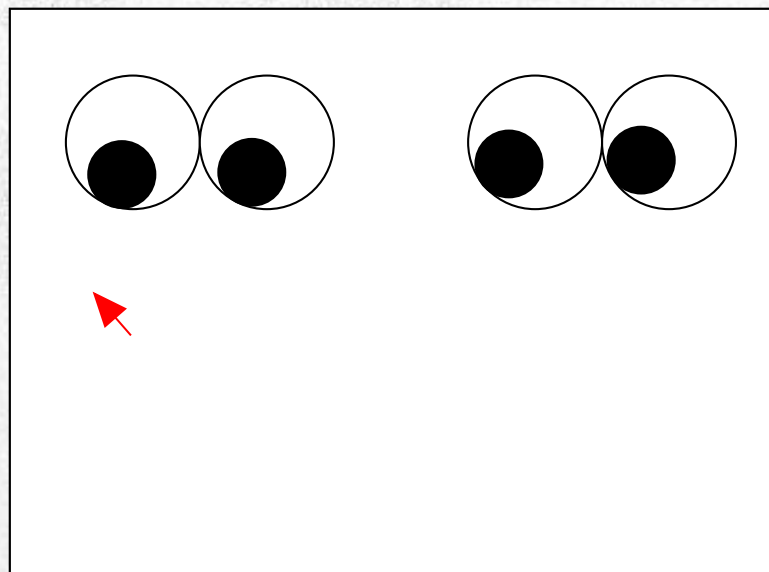
СТРУКТУРА НА ЛЕКЦИЯТА

- Въвеждащ пример
- Обща характеристика
- Събитиен цикъл
- Обработка на събития


ЗАДАЧА (АПЛЕТИ, СЪБИТИЯ, ГРАФИКА)

1 Идея за реализацията?

Да се разработи аplet, в който очите да следват курсора (движи се от мишката)

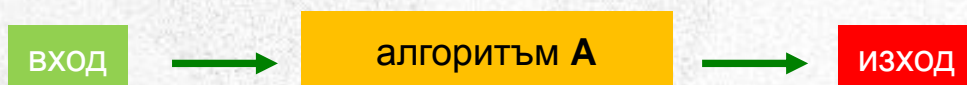


СЪБИТИЯ

- **Особено случване извън програмата ...**
 - Задействане клавиатурата (напр. Enter-клавиш)
 - Движение на мишката
 - Обслужване на графичен интерфейс (напр. натискане на бутон)
- 
- **... предизвиква определена реакция на програмата**
 - Изход на Grad-Celsius-стойност (**TempApplet.java**)
 - Промяна положение на очите (**Eyes.java**, **EyesApplet.java**)
 - Появяване на графичен обект

СЪБИТИЯ-ОРИЕНТИРАНО ИЗПЪЛНЕНИЕ

„Нормално“ управление на програма:
стартира се с `main()` или `init()`



Събитийно-ориентирано управление на програма:

Инициализация
(напр. създаване
на интерфейс)



`main()` или `init()`

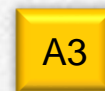
Натискане
бутон



Движение
на мишка



Натискане
enter-клавиш



...

A_i – „малки“ алгоритми

УПРАВЛЯВАНИ ОТ СЪБИТИЯ ПРОГРАМИ



КЪМ ДЕФИНИЦИЯ НА ИЗИСКВАНИЯТА

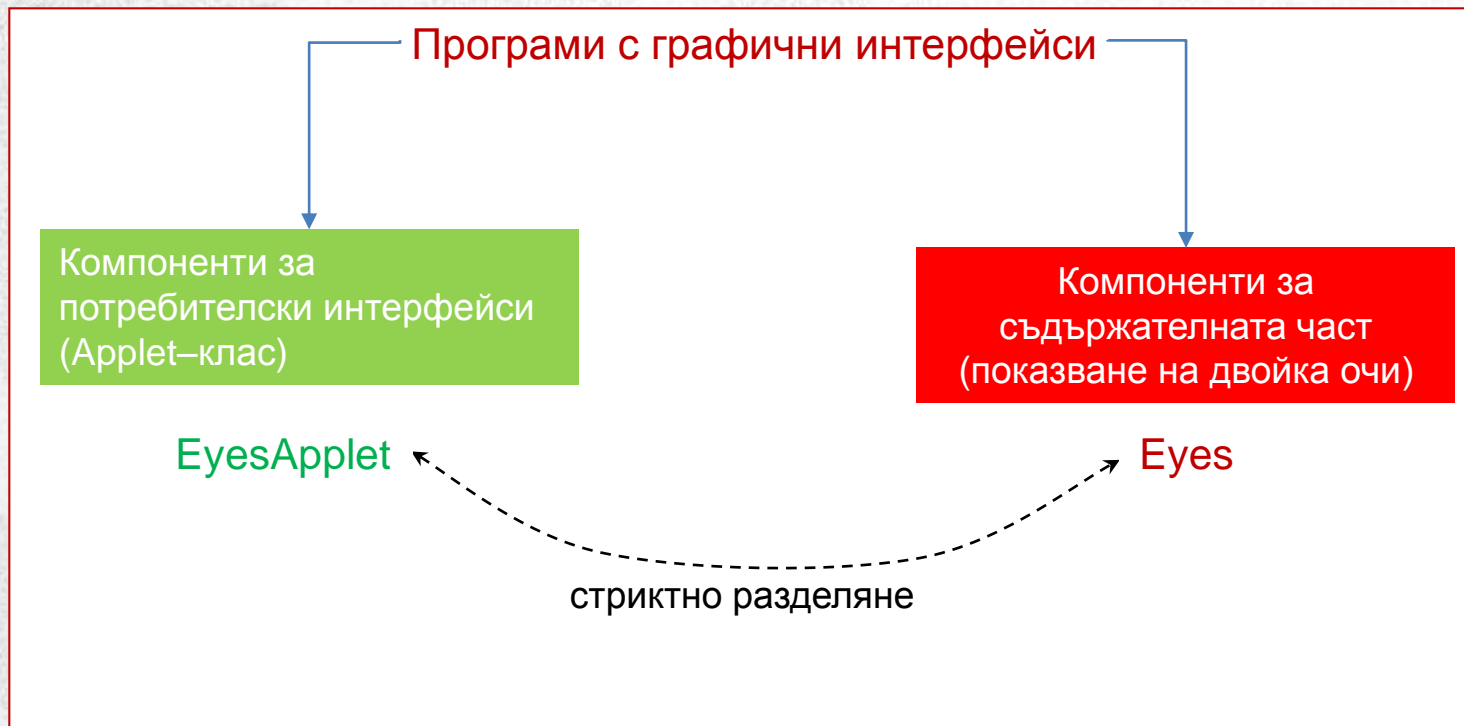
Да се създаде аплет, в който очите следват курсора (движи се от мишката)

Проблеми за изясняване:

- ▶ Брой двойки очи *)
- ▶ Позициониране на очите *)
- ▶ Размери на зениците и очните ябълки *)
- ▶ В какви интервали от време ще се променя посоката на погледа?
(постоянно: в най-малката възможна единица от време)

*) твърди стойности след стартиране на програмата, но да могат да се модифицират лесно чрез константи в програмата

СОФТУЕРНА АРХИТЕКТУРА



Броят двойки очи променлив:
Клас 'Eyes' с произволно създаване на инстанции (обекти)
→ Вид компоненти: ADT за рисуване на двойка очи

СОФТУЕРНА АРХИТЕКТУРА: UML

Спецификация на
двойка очи

Eyes

- left, right : Point
- leftPupil, rightPupil : Point
- EYE_RADIUS, PUPIL_RADIUS: int

Eyes (p: Point)
stare (cursor: Point)

направляван

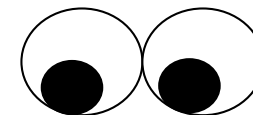
Принцип: потребителският
интерфейс познава приложението
(не обратно)

Потребителски
интерфейс

EyesApplet

init ()
paint (Graphics g)
mouseMoved (MouseEvent e)
mouseDragged (MouseEvent e)

КЛАС 'EYES'



1 Как в Java?

Към спецификация на двойка очи

Eyes

Състояние

- left, right : Point
- leftPupil, rightPupil : Point
- EYE_RADIUS, PUPIL_RADIUS: int

Дясно/ляво око: среда на очната ябълка и зеницата, както и радиус

Поведение

Eyes (p: Point)
stare (cursor: Point)

Създай двойка очи с позиция p

Рисувай двойка очи в посока 'cursor, (*stare* = (втречено) гледам)

Каква промяна на състоянието чрез stare?

Пълна форма: `stare(g: Graphics, cursor: Point)`

КЛАС EYESAPPLET

- Основи:
 - Обработка на събития (клас `MouseMotionListener`)
 - Аплети (клас `Applet`)
- Извикване на всички методи чрез интерпретатора

Инициализация
(при начало на програмата)

Извикване при движение на
мишката

EyesApplet

```
init ( )  
paint (Graphics g)  
mouseMoved (MouseEvent e)  
mouseDragged (MouseEvent e)
```

Директно извикване след
`init()`: рисуват се две двойки

Извикване при движение на
мишката с натиснат бутон
(drag)

java.applet Class Applet

```
java.lang.Object
├── java.awt.Component
│   ├── java.awt.Container
│   │   ├── java.awt.Panel
│   │   └── java.applet.Applet
```

```
public class Applet
extends Panel
```

An applet is a small program that is intended not to be run on its own, but rather to be embedded inside another application and their

The Applet class must be the superclass of any applet that is run in a web browser environment.

API-Class Applet

An applet is a small program that is intended not to be run on its own, but rather to be embedded inside another application and their

Constructor Summary

Constructor	Description
Applet()	Creates a new Applet object

Method Summary

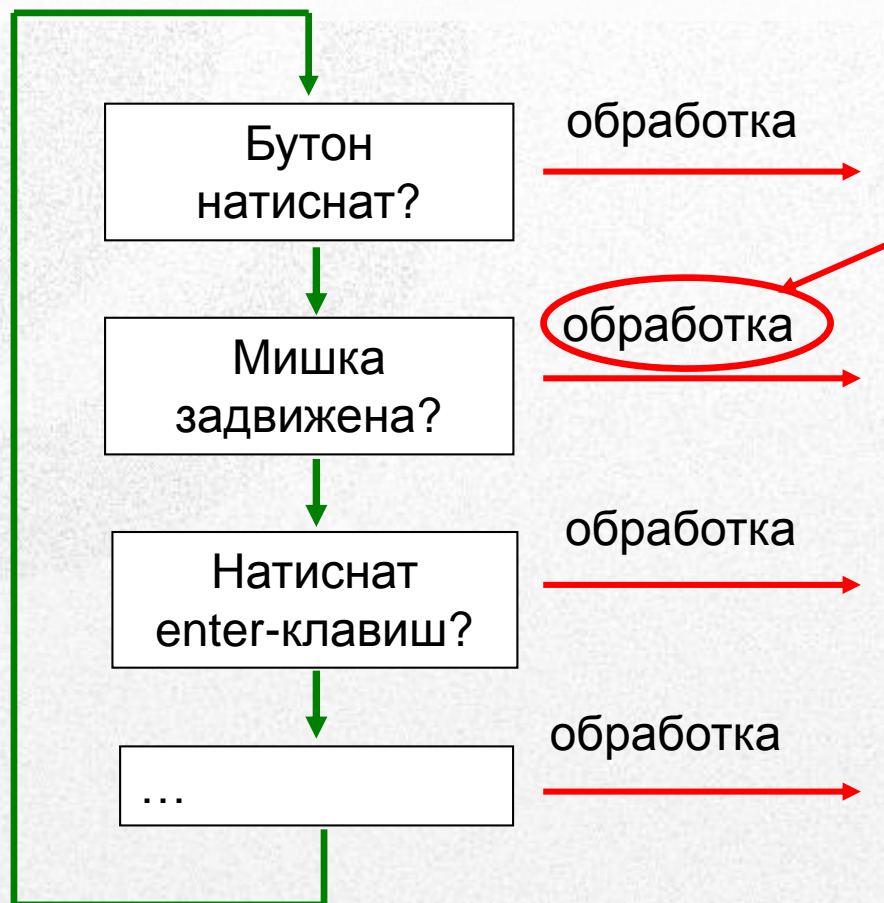
void	destroy() Called by the browser or applet viewer to inform this applet that it is being reclaimed and that it should destroy any resources that it has allocated.
AppletContext	getContext() Determines this applet's context, which allows the applet to query and affect the environment in which it runs.
AudioClip	getAudioClip(URL url, String name) Returns the AudioClip object specified by the URL and name arguments.
URL	getDocumentBase() Gets the URL of the document in which this applet is running.
String	getParameter(String name) Returns the value of the named parameter in the document.
void	init() Called by the browser or applet viewer to inform this applet that it has been loaded into the system.
boolean	isActive() Determines if this applet is active.
void	play(URL url, String name) Plays the audio clip given the URL and a specifier that is relative to it.
void	resize(int width, int height) Requests that this applet be resized.
void	start() Called by the browser or applet viewer to inform this applet that it should start its execution.
void	stop() Called by the browser or applet viewer to inform this applet that it should stop its execution.

void init()

Called by the browser or applet viewer to inform this applet that it has been loaded into the system

УПРАВЛЯВАНИ ОТ СЪБИТИЯ ПРОГРАМИ: СЪБИТИЕН ЦИКЪЛ

JVM
управлява
събитийния
цикъл



EyesApplet.java

```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;

public class EyesApplet extends Applet
    implements MouseMotionListener {
    Point cursor;
    Eyes e1, e2;

    public void init () {
        // Register the Listener.
        addMouseMotionListener(this);
        setSize(500,400);
        setBackground(Color.LIGHT_GRAY);
        e1 = new Eyes(new Point (63,30));
        // center of one eye
        e2 = new Eyes(new Point (437,30));
        // center of the other
        cursor = new Point(250, 2000);
        // initial cursor
    }

    public void paint(Graphics g) {
        e1.stare(g, cursor);
        e2.stare(g, cursor);
    }

    public void mouseMoved (MouseEvent e) {
        cursor = e.getPoint();
        repaint();
    }

    public void mouseDragged (MouseEvent e) {}
}
```

Eyes.java

```
public class Eyes {

    private Point left, right, leftPupil, rightPupil;
    private final int EYE_RADIUS = 30, PUPIL_RADIUS = 10;

    public Eyes(Point c) {
        left = new Point(c.x-EYE_RADIUS-3, c.y);
        right = new Point(c.x+EYE_RADIUS+3, c.y);
    }

    private void fillCircle (Graphics g,
                             Point center, int radius) {
        g.fillOval(center.x-radius, center.y-radius,
                   2*radius, 2*radius);
    }

    public void stare (Graphics g, Point cursor) {

        // Draw the white eyes
        g.setColor(Color.WHITE);
        fillCircle(g, left, EYE_RADIUS);
        fillCircle(g, right, EYE_RADIUS);

        // Draw the pupils
        g.setColor(Color.black);
        leftPupil = compute (cursor, left);
        fillCircle(g, leftPupil, PUPIL_RADIUS);
        rightPupil = compute (cursor, right);
        fillCircle(g, rightPupil, PUPIL_RADIUS);
    }

    private Point compute (Point cursor, Point eye) {
        double d = Math.sqrt((cursor.x-eye.x)*(cursor.x-eye.x)
                               + (cursor.y-eye.y)*(cursor.y-eye.y));
        int r = EYE_RADIUS - PUPIL_RADIUS;
        return new Point (eye.x + (int)((cursor.x-eye.x)*r/d),
                           eye.y + (int)((cursor.y-eye.y)*r/d));
    }
}
```

ОБРАБОТКА НА СЪБИТИЯ (EVENT-HANDLING)

- Особено случване извън програмата: сигнали от заобикалящата среда на програмата
 - Задействане клавиатурата (напр. enter-клавиш)
 - Движение на мишката
 - Обслужване на графичен интерфейс (напр. натискане на бутон)



- ... водят до създаване на едно събитие
 - Обект на един Event-клас (сравни обекти-изключения, създадени при грешка по време на изпълнение на програма)

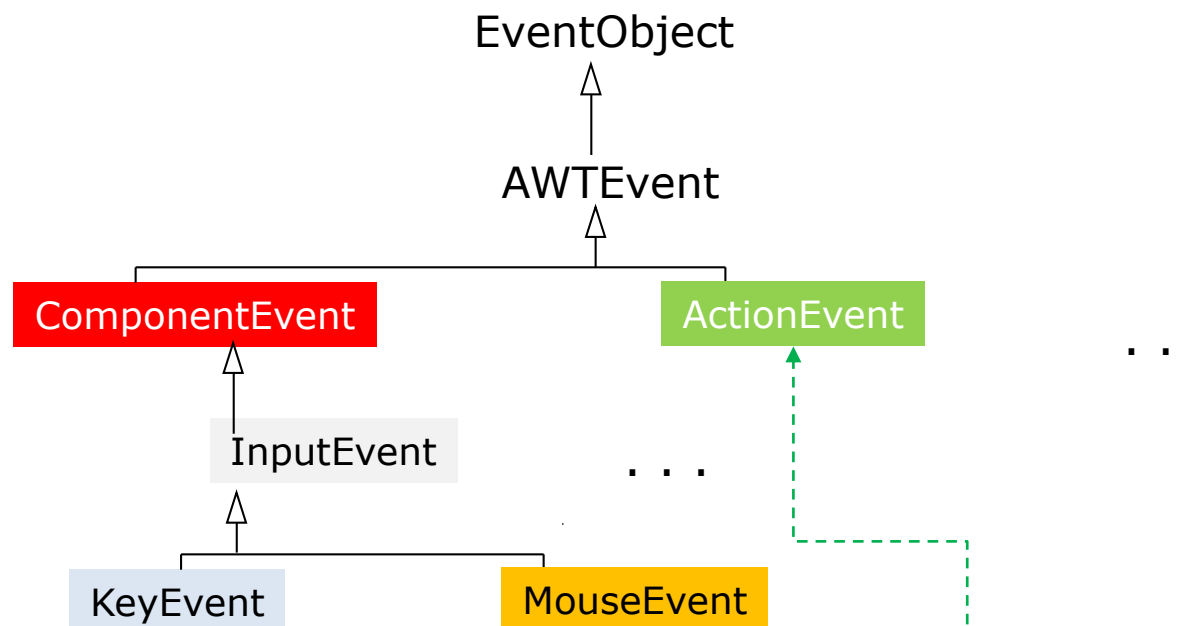
Вид: MouseEvent

getPoint () : Point

. . .

ИЕРАРХИЯ НА КЛАСОВЕТЕ ЗА СЪБИТИЯ

Част от Java-API



Източник: mouse

Източник: напр. Enter-клавиш
(Temperature-Applet)

java.awt.event

Class MouseEvent

```
java.lang.Object
├── java.util.EventObject
│   └── java.awt.AWTEvent
│       ├── java.awt.event.ComponentEvent
│       │   ├── java.awt.event.InputEvent
│       │   └── java.awt.event.MouseEvent
```

All Implemented Interfaces:

[Serializable](#)

Direct Known Subclasses:

[MenuDragMouseEvent](#), [MouseWheelEvent](#)

```
public class MouseEvent
extends InputEvent
```

API-Class MouseEvent

An event which indicates that a mouse action occurred in a component...

An event which indicates that a mouse action occurred in a component. A mouse action is considered to occur in a particular component if and only if the mouse cursor is over the unobscured part of the component's bounds when the action happens. Component bounds can be obscured by the visible component's children or by a menu or by a top-level window. This event is used both for mouse events (click, enter, exit) and mouse motion events (moves and drags).

Field Summary

static int	BUTTON1	Indicates mouse button #1; used by getButton() .
static int	BUTTON2	Indicates mouse button #2; used by getButton() .
static int	BUTTON3	Indicates mouse button #3; used by getButton() .
static int	MOUSE_CLICKED	The "mouse clicked" event.
static int	MOUSE_DRAGGED	The "mouse dragged" event.
static int	MOUSE_ENTERED	The "mouse entered" event.
static int	MOUSE_EXITED	The "mouse exited" event.
static int	MOUSE_FIRST	The first number in the range of ids used for mouse events.
static int	MOUSE_LAST	The last number in the range of ids used for mouse events.
static int	MOUSE_MOVED	The "mouse moved" event.
static int	MOUSE_PRESSED	The "mouse pressed" event.
static int	MOUSE_RELEASED	The "mouse released" event.

static int MOUSE_CLICKED
The "mouse clicked" event

static int MOUSE_MOVED
The "mouse moved" event

Point getPoint()
Returns the x,y position (curser) of the event relative to the source component

АНАЛОГИЯ: СЪБИТИЯ - ИЗКЛЮЧЕНИЯ

По време на изпълнение: особена ситуация

Грешка по време
изпълнение

Сигнал от средата: периферия
(клавиатура/мишка)

създава

Exception object

Event-Object

getMessage()

Вид: NumberFormat
Детайли: a1
StackTrace: ...

getPoint() ...

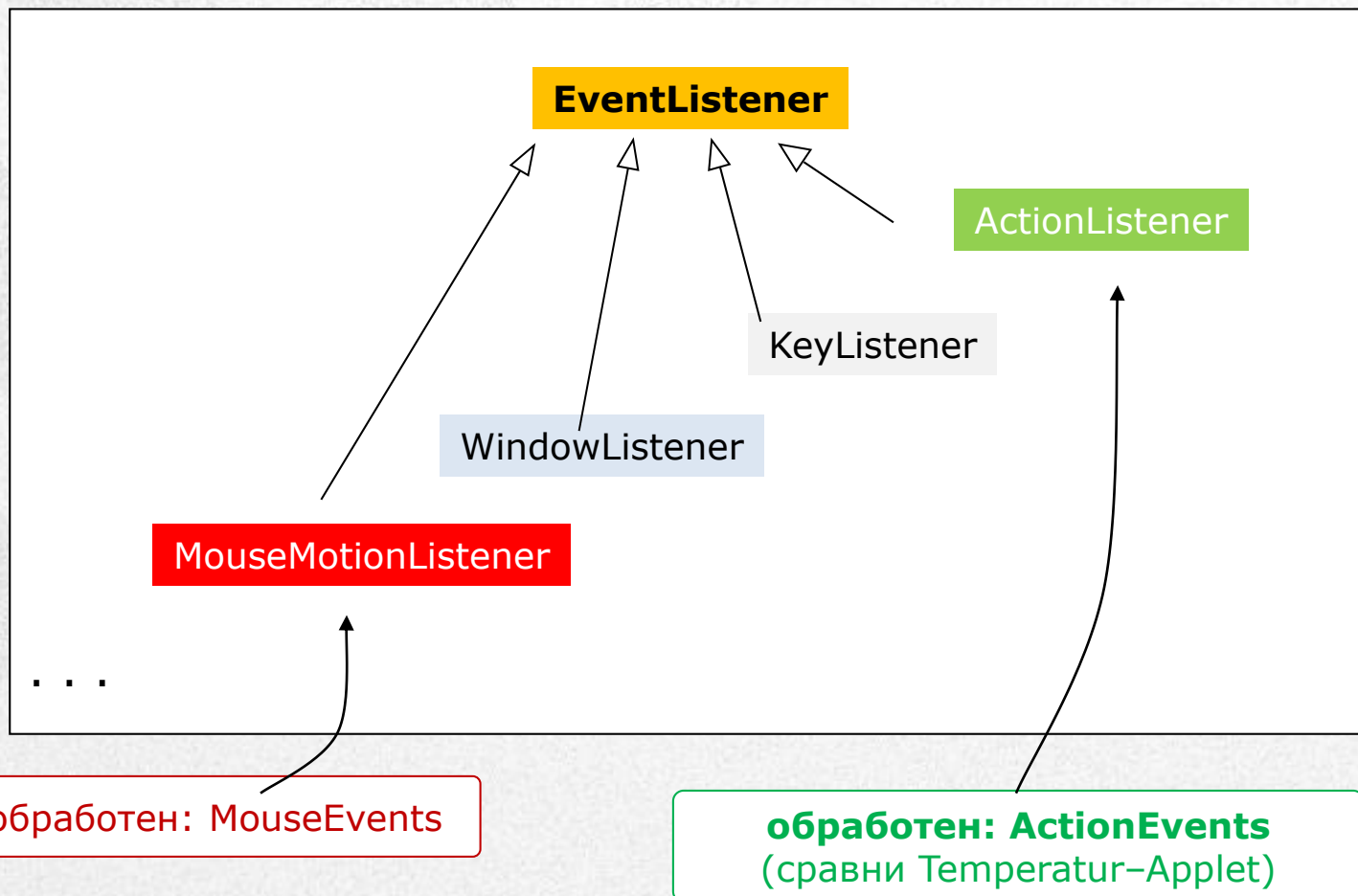
Вид: MouseEvent

обработен чрез

try-catch

Обект на подклас **Interface EventListener**
(част от Java - API)

КОМПОНЕНТИ ЗА ОБРАБОТКА НА СЪБИТИЯ



Interface **MouseMotionListener**API-Class (Interface)
MouseMotionListener

```
public interface MouseMotionListener
extends EventListener
```

The listener interface for receiving mouse motion events on a component. (For clicks and other mouse events, use the `MouseListener`.)

Method Summary

void	mouseDragged (MouseEvent e)	Invoked when a mouse button is pressed on a component and then dragged.
void	mouseMoved (MouseEvent e)	Invoked when the mouse cursor has been moved onto a component but no buttons have been pushed.

The listener interface for receiving mouse motion events on a component. (For clicks and other mouse events, use `MouseListener`.)

Method Detail**mouseDragged**

```
void mouseDragged(MouseEvent e)
```

Invoked when a mouse button is pressed on a component where the drag originated until the mouse button is released.

Due to platform-dependent Drag&Drop implementation

void mouseDragged(MouseEvent e)
Invoked when a mouse button is pressed on a component and then dragged

```
void mouseMoved(MouseEvent e)
```

Invoked when the mouse cursor has been moved onto a component but no buttons have been pushed

to the component
(component).

operation.

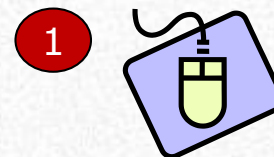
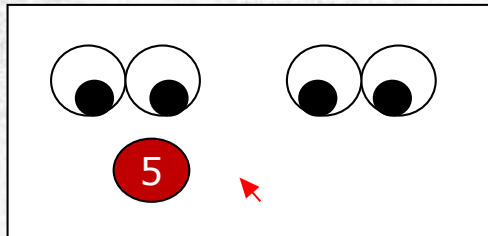
mouseMoved

```
void mouseMoved(MouseEvent e)
```

Invoked when the mouse cursor has been moved onto a component but no buttons have been pushed.



ТЕХНИКА И ПРОТИЧАНЕ ОБРАБОТКА НА СЪБИТИЯ



1. Hardware изпраща сигнал към операционната система
2. Java-VM-Mashine получава сигнала от операционната система
3. Java-VM-Mashine създава събитийен обект **e**
4. Java-VM-Mashine изпраща съобщение към обекта за обработка на събитието (т.е. чрез извикване на един метод): напр.

mouseMoved(e)

5. Методът за обработка на събитието оценява събитийния обект и реагира,
напр.

e.getPoint();repaint();

ОЦЕНКА НА MOUSEEVENTS ЧРЕЗ MOUSEMOTIONLISTENER

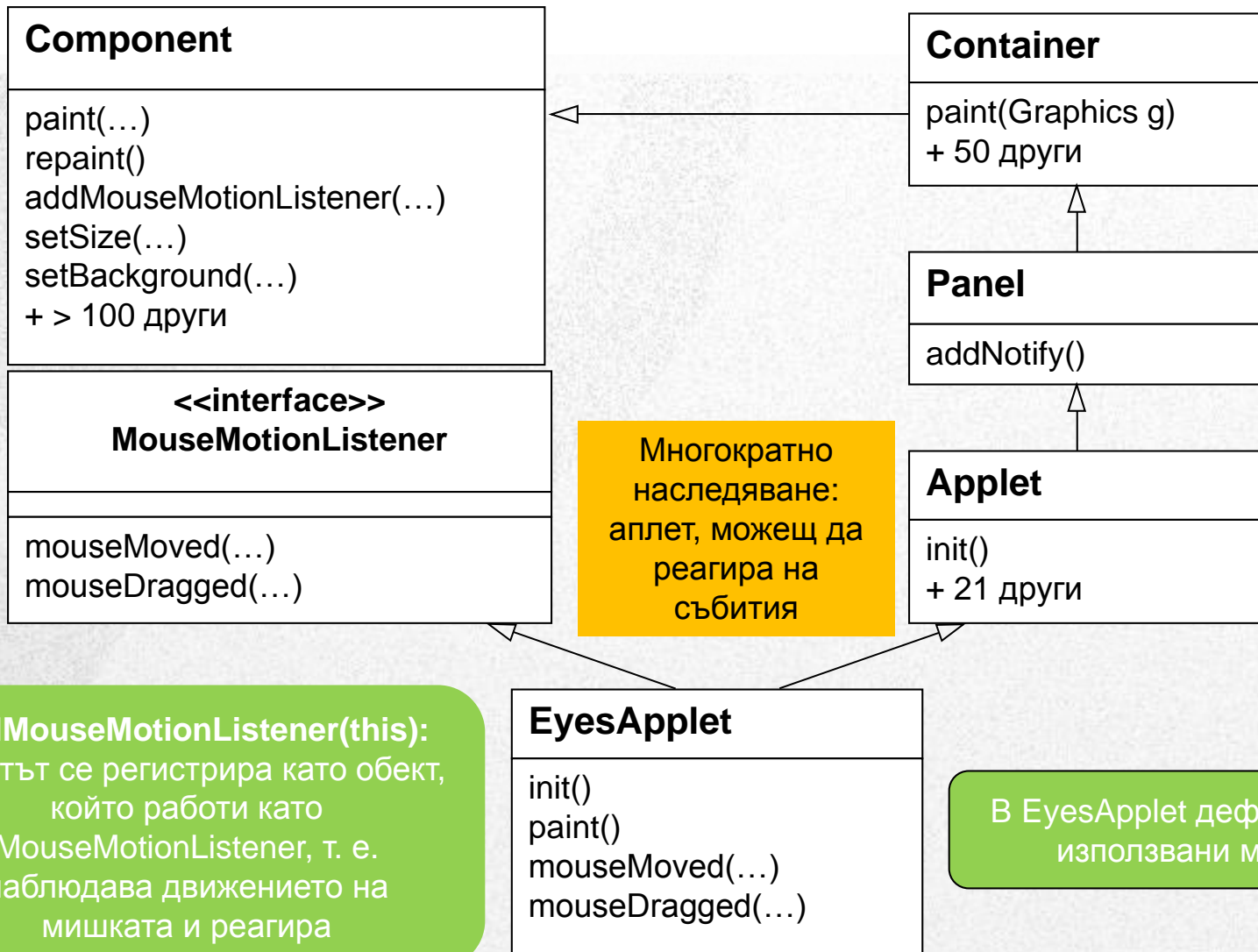
```
class EyesApplet extends Applet
    implements MoouseMotionListener {

    public void mouseMoved (MouseEvent e)      {
        cursor = e.getPoint();
        repaint();
    }
    ...
}
```

Последователност на извиквания при движение на мишка

- Създаване на обект **e** от тип MouseEvent
e получава информация чрез позицията на курсора
- Извикване mouseMoved(e)
- Питане за новата позиция на мишката: e.getPoint
- repaint(): извиква paint() отново – ново рисуване на очите

СОФТУЕРНА АРХИТЕКТУРА



КЛАС EYES

Рисуване на очите

Очна ябълка
с постоянно
положение

проблем:

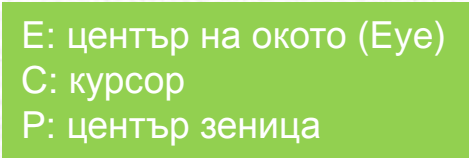
дадено: очна ябълка+позиция курсор
търси се: къде се намира зеницата ?

→ Питагор
→ подобни триъгълници

Курсор



КЛАС EYES



$$P.y = E.y + (C.y - E.y) * r / d$$

STARE И COMPUTE

Ново рисуване на двойка очи:

- Стара позиция на очната ябълка (бял)
- Зеницата гледа в посока ,cursor' (черен)

```
public void stare (Graphics g, Point cursor)
```

Прилагане на геометрични основи:
изчислява точка P

```
private Point compute (Point cursor, Point eye)
```

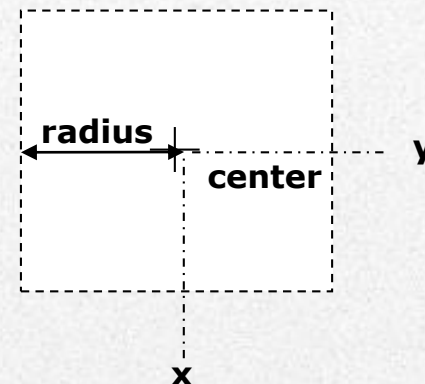
FILLCIRCLE: КРЪГ С RADIUS ОКОЛО CENTER

```
private void fillCircle (Graphics g,  
                        Point center,  
                        int radius)  
{  
    g.fillOval (center.x-radius,  
               center.y-radius,  
               2*radius, 2* radius);  
}
```

Рисува кръг в квадрат

лява горна ъглова точка

Дължина и широчина
на правоъгълника



API-Class Graphics

[java.lang.Object](#)
↳ [java.awt.Graphics](#)

public abstract class **Graphics**
extends [Object](#)

The Graphics class is the abstract

The Graphics class is the abstract base class for all graphics contexts that allow an application to draw onto components that are realized on various devices, as well as onto off-screen images

Constructor Summary

protected	Graphics ()	Constructs a new graphics object.
-----------	-----------------------------	-----------------------------------

Method Summary

abstract void	clearRect (int x, int y, int width, int height)	Clears the specified rectangle by filling it with the background color of the current drawing surface.
---------------	---	--

abstract void	clipRect (int x, int y, int width, int height)	Intersects the current clip with the specified rectangle.
---------------	--	---

abstract void	copyArea (int x, int y, int width, int height, int dx, int dy)	Copies an area of the component to a new location.
---------------	--	--

abstract Graphics	create ()	Creates a new Graphics object that is a copy of this Graphics object.
-----------------------------------	---------------------------	---

Graphics	create (int x, int y, int width, int height)	Creates a new Graphics object based on this Graphics object, but with a new translation and clip area.
--------------------------	--	--

abstract boolean	drawImage (Image img, int x, int y, int width, int height, ImageObserver observer)	Draws as much of the specified image as has already been scaled to fit inside the specified rectangle.
------------------	---	--

abstract void	drawLine (int x1, int y1, int x2, int y2)	Draws a line, using the current color, between the points (x1, y1) and (x2, y2) in this graphics context's coordinate system.
---------------	---	---

abstract void	drawOval (int x, int y, int width, int height)	Draws the outline of an oval bounded by the specified rectangle with the current color.
---------------	--	---

abstract void	fillOval (int x, int y, int width, int height)	Fills an oval bounded by the specific rectangle with the current color
---------------	--	--

void	drawPolygon (int[] xPoints, int[] yPoints, int nPoints)	Draws the outline of a polygon defined by the specified Polygon object.
------	---	---

abstract void	drawPolyline (int[] xPoints, int[] yPoints, int nPoints)	Draws a sequence of connected lines defined by arrays of x and y coordinates.
---------------	--	---

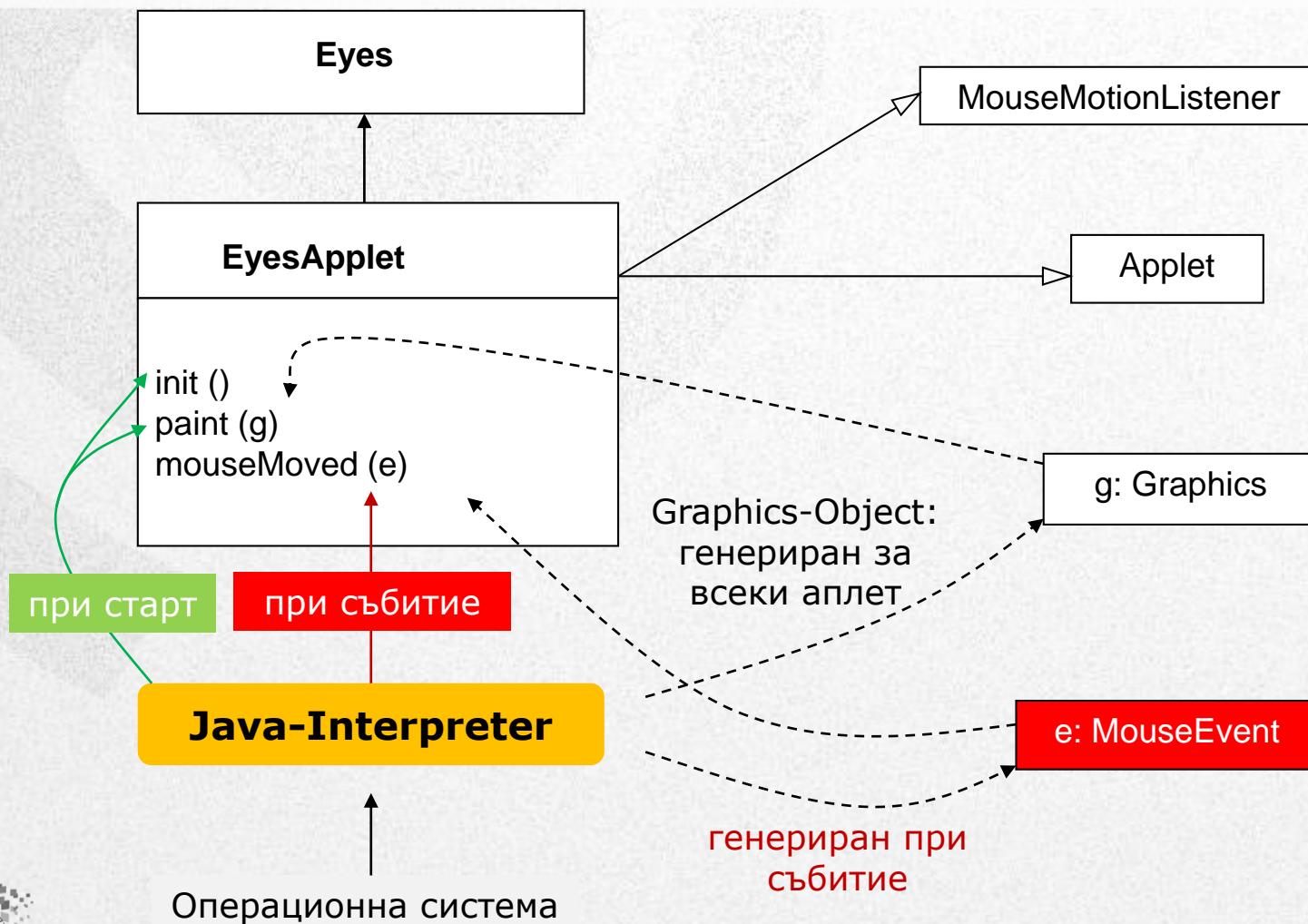
void	drawRect (int x, int y, int width, int height)	Draws the outline of the specified rectangle.
------	--	---

abstract void	drawRoundRect (int x, int y, int width, int height, int arcWidth, int arcHeight)	Draws an outlined round-cornered rectangle using this graphics context's current color.
---------------	--	---

abstract void	drawString (AttributedCharacterIterator iterator, int x, int y)	Draws the text given by the specified iterator, using this graphics context's current color.
---------------	--	--

abstract void	drawText (String text, int x, int y)	Draws the text given by the specified string, using this graphics context's current color.
---------------	---	--

ДИНАМИЧЕН АСПЕКТ КЪМ 'EYESAPPLET'



КЛАС EYESAPPLET: ИЗВИКВАНЕ НА МЕТОД

```
public class EyesApplet extends Applet  
    implements MouseMotionListener {
```

```
    Point cursor;  
    Eyes e1, e2;
```

```
    public void init () {...}
```

```
    public void paint(Graphics g) {  
        e1.stare(g, cursor);  
        e2.stare(g, cursor);  
    }
```

```
    public void mouseMoved (MouseEvent e) {  
        cursor = e.getPoint();  
        repaint();  
    }
```

```
    public void mouseDragged (MouseEvent e) {...}
```

Аплет: инициализиран и регистриран като MouseMotionListener

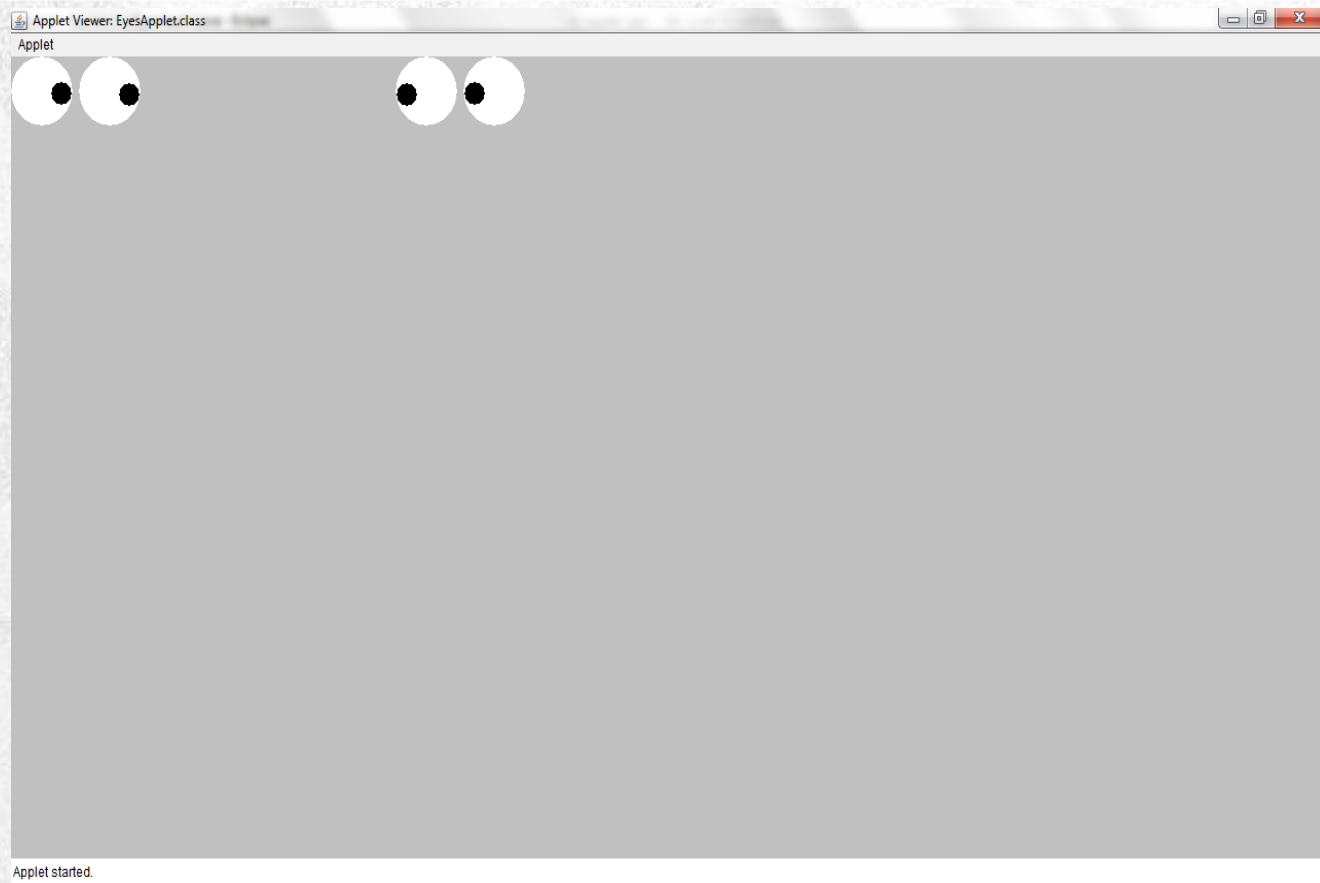
Директно извикване след `init()`

Автоматично извикване при движение на мишката

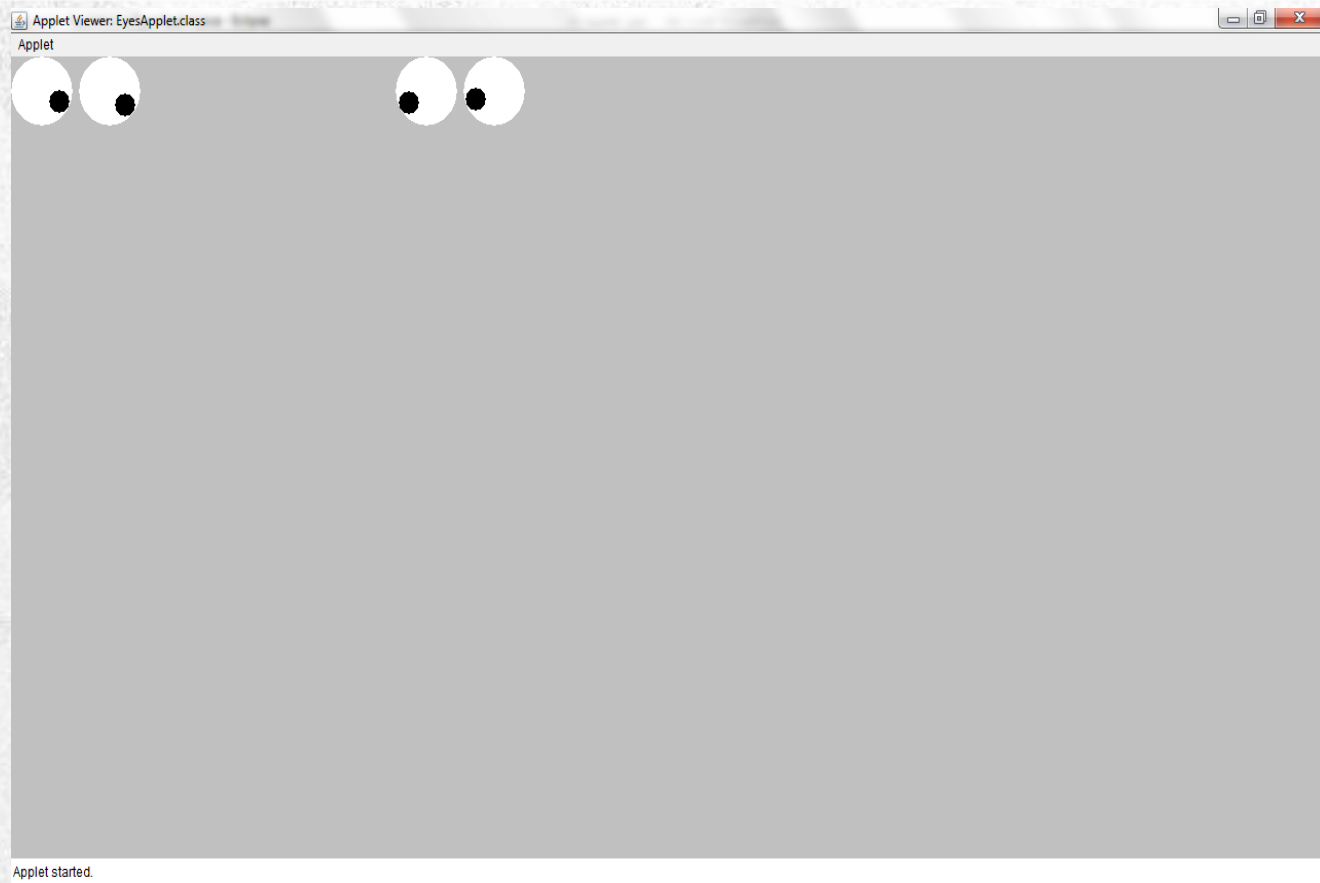
Доставя физическата позиция на мишката

- Към всеки аплет, JVM генерира Graphics-Object g за рисуване в аплета
- Всички методи: чрез JVM-Mashine извиквани

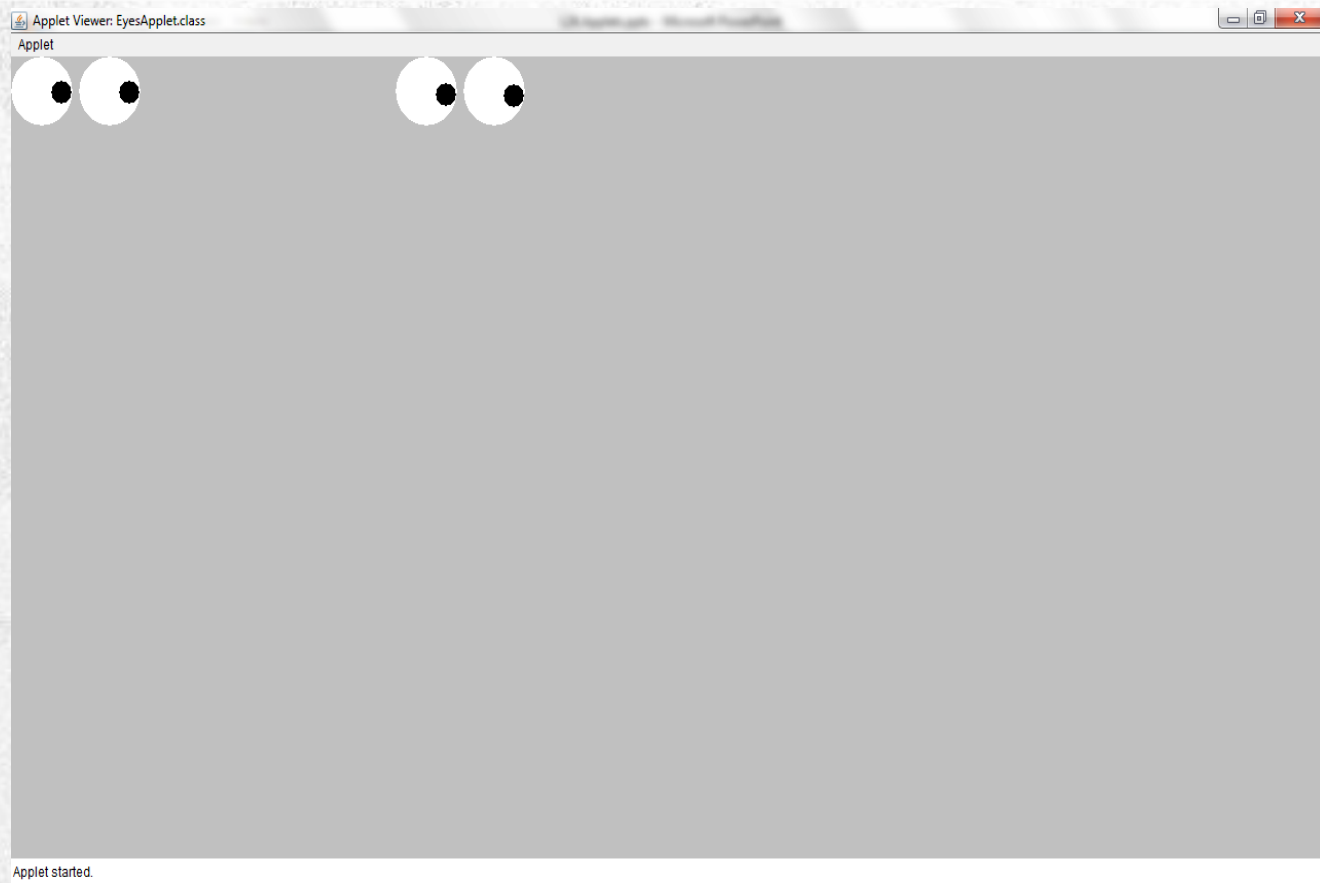
АПЛЕТ



АПЛЕТ

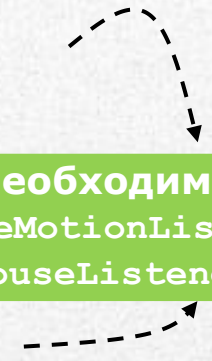


АПЛЕТ



ЗАДАЧИ ЗА САМОПОДГОТОВКА

- Използване на програмата и пълно разбиране (Java-API)
- Промяна позиционирането на очите
- Други мерки: големина на зеницата и очната ябълка
- Вместо две очи: четири ...
- Динамично създаване на нови двойки очи
- Нова визия: клепачи ...
- При клик на мишката: клепачите се затварят



необходими:
`MouseMotionListener`
`MouseListener`

БЛАГОДАРЯ ЗА ВНИМАНИЕТО!

КРАЙ “СЪБИТИЯ”

