# Lucrarea de laborator nr. 13

**Clasa *Graphics***

În cadrul acestui laborator se vor utiliza funcțiile clasei **`Graphics`** pentru a executa grafică liniară simplă în fereastra applet.
Ce este un applet? Un applet reprezintă un program Java ce gestionează o suprafată de afişare (container) ce poate fi inclusă într-o pagină Web.

Pachete: **`java.applet`**
Clasa principală extinde **`Applet`**

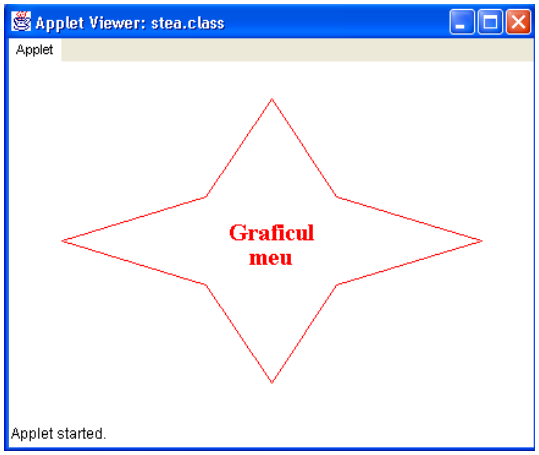
Un applet nu poate fi executat independent El este executat de către browser-ul în care a fost încărcată pagina Web ce conține appletul sau de către programe specializate(**appletviewer**).

Deşi programul applet-ului urmator este foarte simplu, el va constitui fundamentul care va fi folosit pentru crearea unor applet-uri mai complexe în viitor. După terminarea acestei şedințe de laborator, programatorul va înțelege următoarele concepte cheie:

- Pentru a efectua operațiuni de grafică în applet-ul creat, se utilizează clasa **`Graphics`**.
- Prin utilizarea funcției **`lineDraw`**, se pot desena linii într-o fereastră applet.
- Prin interconectarea liniilor pe care applet-ul le desenează folosind funcția **`lineDraw`**, aplicația poate trasa forme complexe.
- Prin apariții şi dispariții intermitente, chiar folosind grafică simplă, applet-ul creat poate capta rapid atenția utilizatorului.
- Folosind clasa **`FontMetrics`**, un applet poate determina lățimea unui şir în pixeli; aceasta este foarte important atunci când se combină text şi grafică într-o fereastră applet.

**Crearea applet-ului Stea**

Applet-ul **Stea** utilizează clasa Java *Graphics* pentru a desena o stea care “clipeşte” (se aprinde şi se stinge) în fereastra applet. La rularea applet-ului, Java deschide o fereastră applet şi afişează o stea clipitoare care conține un mesaj text, aşa cum se vede în figura următoare:




*Afişarea applet-ului Stea*

Scopul principal al applet-ului Stea este acela de a demonstra cum se pot crea ilustrații liniare simple utilizând biblioteca Java *Graphics* . Este posibilă şi combinarea stelei clipitoare cu o pagină de bază existentă pentru a capta atenția utulizatorului. În lucrările următoare, se va demonstra modul de combinare a elementelor grafice Java cu o imagine de fundal.

**Fişierul HTML**

Applet-ul Stea se axează pe programul folosit pentru realizarea graficii. De aceea, applet-ul nu acceptă parametri HTML. În acest moment, pentru a avea acces la applet, se foloseşte următoarea intrare HTML:

```
<applet code=Stea.class width=400 height=400> </applet>
```

**Stea** utilizează funcțiile clasei **`Graphics`** pentru a afişa text şi grafică liniară.Programul care urmează implementează applet-ul **Stea**:

```
//*****************************************
//Stea.java
//*****************************************

import java.applet.*;
import java.awt.*;

//*****************************************

public class stea extends Applet implements Runnable
     {
        boolean blink = false;
        //_____________________________________

public void start ()
  {
        (new Thread(this)).start ();}

//_____________________________________

public void run ()
{while (true)
  {
    repaint ();
```

```
    try
         {
          Thread.sleep(500);}
    catch (InterruptedException e)
                     {
                      }
               }
        }

//_____________________________________

public void paint (Graphics g)
{
//Structura urmatoare permite schimbarea culorilor in functie //de valoarea logica a
variabilei blink
  if  (blink)
    {
        g.setColor(Color.red);
        blink=false;
                       }
  else
     {
        g.setColor(Color.blue);
        blink=true;
                    }
   int width=size().width;
   int height=size().height;
   Font font=new Font("TimesRoman",  Font.BOLD,  20);
   g.setFont(font);
   FontMetrics  font_metrics  = g.getFontMetrics();
// Are loc declararea textului care ca constitui referinta desenului

g.drawString("Graficul",(width-font_metrics.stringWidth("Graficul"))/2, height/2);
g.drawString("meu",(width-font_metrics.stringWidth("meu"))/2,height/2+20);
// Are loc creearea elementului grafic
             g.drawLine(width/2,(height*9)/10,(width*3)/8,(height*5)/8);
            g.drawLine((width*3)/8,(height*5)/8,width/10,height/2);
g.drawLine(width/10,height/2,(width*3)/8,(height*3/8));
g.drawLine((width*3)/8, (height*3)/8, width/2, height/10);
g.drawLine(width/2, height/10,(width*5)/8,(height*3)/8);
g.drawLine((width*5)/8,(height*3)/8,(width*9)/10,height/2);
g.drawLine((width*9)/10,height/2,(width*5)/8,(height*5)/8);
g.drawLine((width*5)/8,(height*5)/8,width/2,(height*9)/10);
}
}
```

**Analiza applet-ului Stea**

Applet-ul **Stea** importă două pachete: pachetul **`java.applet`** furnizează clasele necesare applet-ului pentru a extinde clasa **`Applet`** ,iar pachetul **`java.awt`** conține **`Graphics`** şi **`Fonts`***:*

```
import java.applet.*,
import java.awt.*,
```

**Extinderea Applet-ului**

Ca şi applet-urile de tip text mobil precedente, applet-ul **Stea** extinde clasa **`Applet`** prin implementarea unei interfețe **`Runnable`***:*

```
public class star extends Applet implements Runnable
```

După cum s-a discutat, programul pentru applet-ul **Stea** este foarte simplu.Într-adevăr, applet-ul utilizează numai o singură variabilă , care specifică dacă steaua clipeşte sau nu. Dacă variabila **`blink`** este adevărată, applet-ul desenează steaua folosind o culoare roşie. Dacă variabila **`blink`** este falsa , applet-ul desenează steaua in culoarea albastră. Prin folosirea de diferite culori pentru redesenarea stelei , applet-ul face ca steaua să clipească:

```
boolean blink=false;
```

Deoarece applet-ul nu are variabile de inițializat, acesta nu furnizează funcția *init.* Când browser-ul rulează applet-ul, prima funcție de executat este funcția *start* care va fi discutată mai jos.

**Funcția Start**

În capitolele precedente , applet-urile create utilizau un obiect **`Thread`** pentru a deplasa un mesaj transversal prin fereastra applet. Într-un mod similar , applet-ul **Stea** utilizează un obiect **`Thread`** pentru a determina desenarea stelei în diferite culori. Atunci când browser-ul execută funcția **`start`** , applet-ul creează firul:

```
public void start( )
    {
        (new Thread(this)) . start  ( );
    }
```

**Funcția Run**

În cadrul funcției *start* prezentate aterior, applet-ul crează şi lansează obiectul *Thread* care, la randul său, va face steaua să clipească. Applet-ul foloseşte funcția **`run`** a firului, prezentată mai jos, pentru ca steaua să clipească la infinit sau până la sfârşitul applet-ului:

```
Public void run( )
     {
            while (true)
          {
            repaint( );
      try
          {
            Thread. sleep(500);
            }
            catch  (InterruptedException  e)
            {
            }
      }
 }
```

După cum se observă funcția **`run`** necesită o declarație **`while`** care se va executa la infinit. În cadrul ciclului, funcția apelează funcția **`repaint`** pentru a actualiza fereastra applet. În continuare, applet-ul

utilizează funcția **`Thread.sleep`** pentru o temporizare de ½ secunde(500milisecunde). Dacă funcția **`Thread.sleep`** este întreruptă, ea generează o excepție pe care declarația **`catch`** o prinde şi o ignoră.

### Funcția Paint

Funcția **`paint`** a applet-ului **Stea** efectuează cea mai mare parte din procesarea applet-ului. După cum s-a observat funcția **`run`** a firului apelează **`repaint`** la fiecare iterație a buclei care la rândul său apelează **`paint.`** Funcția **`paint`** ,la rândul său desenează steaua şi textul "**Stea**" în fereastra applet.

Funcția **`paint`** creează steaua clipitoare prin alternarea culorii stelei între albastru şi roşu. De aceea primul pas pe care applet-ul îl execută este acela de a determina ce culoare să folosească.
Funcția **`paint`** determină culoarea stelei prin examinarea variabilei **`blink`**. Dacă variabila **`blink`** este adevărată, **`paint`** foloseşte culoarea roşie. Dimpotrivă, dacă variabila **`blink`** este falsă, **`paint`** foloseşte culoarea albastră:

```
If (blink)
  {
      g.setcolor(color.read);
      blink=false;

         }
else
   {
      g.setcolor(color.blue);
      blink=true;
    }
```

În continuare, funcția **`paint`** utilizează metoda **`size`** pentru a determina lățimea şi înălțimea ferestrei applet:

```
int width = size().width;
int height = size().height;
```

Aşa cum s-a discutat, applet-ul Stea combină textul cu grafica. După cum se poate vedea, applet-ul utilizează un font TimesRoman de 20 puncte Bold. Suplimentar applet-ul utilizează metoda **`getFontMetrics`** pentru a determina lățimea fontului în pixeli:

```
Font font =new font("TimesRoman", Font.BOLD, 20);
g.setFont ( font ) ;
FontMetrics font_metrics = g.getFontMetrics ( ) ;
g.drawString("Graficul",(width-font_metrics.stringWidth ("Graficul"))/2, height/2);
g.drawString("meu",(width-font_metrics.StringWidth("Samples"))/2,height/2+20);
```

Obiectul **`FontMetrics`** furnizează applet-ului informații privind fontul, ca lățimea unui şir de pixeli. Cunoscînd lungimea unui şir în pixeli, applet-ul poate centra şirul. Aşa cum se pot vedea, pentru a centra şirul, applet-ul scade lățimea şirului în pixeli din lățimea ferestrei şi apoi împarte

rezultatul la 2. Pentru a centra un şir pe verticală într-o fereastră,applet-ul foloseşte jumătate din înălțimea ferestrei.

În cele din urmă, funcția **paint** utilizează o serie de apelări ale funcției **drawLine.**
Pentru a desena liniile care definesc steaua în patru colțuri. Dacă examinăm figura 8, se constată că steaua se compune din 8 linii. De aceea, applet-ul foloseşte 8 apelări ale funcției **drawLine** pentru a crea steaua:

```
g.drawLine(width/2,(height*9)/10,(width*3)/8,(height*5)/8);
            g.drawLine((width*3)/8,(height*5)/8,width/10,height/2);
g.drawLine(width/10,height/2,(width*3)/8,(height*3/8));
g.drawLine((width*3)/8, (height*3)/8, width/2, height/10);
g.drawLine(width/2, height/10,(width*5)/8,(height*3)/8);
g.drawLine((width*5)/8,(height*3)/8,(width*9)/10,height/2);
g.drawLine((width*9)/10,height/2,(width*5)/8,(height*5)/8);
g.drawLine((width*5)/8,(height*5)/8,width/2,(height*9)/10);
}
}
```

Primii doi parametriai funcției **drawLine** specifică coordonatele de pornire x şi y ale liniei.Următorii doi parametri specifică coordonatele finele x şi y ale liniei. Se menționează că coordonatele finale ale unui apel **drawLine** devin coordonatele de pornire ale următorului apel **drawLine** ultimul apel sfârşindu-se acolo unde a început primul apel.

Funcția **run** utilizează variabilele **width** şi **height** pentru a determina coordonatele, astfel încît poate desena steaua corect,indiferent de dimensiunile ferestrei.

## Concluzii

Applet-ul Stea demonstrează modul de realizare a unor elemente de grafică simplă într-o fereastră applet. Acest aplet a permis introducerea următoarelor concepte cheie:

- ✓ Funcția **lineDraw** face posibilă desenarea unei linii în fereastra applet.Pentru a utiliza funcția,applet-ul creat specifică coordonatele x si y ale punctelor extreme ale liniei.

- ✓ Utilizând funcția **lineDraw** pentru legarea liniilor între ele într-o fereastră applrt, pot fi create forme complexe.

- ✓ Pentru ca un obiect dintr-o fereastră să clipească, un applet schimbă pur şi simplu culoarea obiectului la intervale de timp specificate.

- ✓ Clasa **FontMetrics** furnizează uni applet informații privind fontul curent. Astfel, applet-ul Stea a utilizat funcția **stringWidth** pentru a determina lățimea unui şir text, pe care applet-ul a utilizat-o apoi pentru a centra şirul în fereastră.

## Chestiuni de studiat

1. Creați şi rulați în cadrul unui fişier html appletul Stea.

2. Introduceți în cadrul fisierului HTML şi alte applete realizate anterior. Ce observați?

3. Pe baza graficului anterior, realizați desenarea unui pătrat având ca bază de desenare textul:
   **Sunt**
   **Inconjurat**

4. Desenați un hexagon în interiorul căruia vă veți introduce numele şi prenumele (ca elemente pe baza cărora se vor realiza calculele de desenare).

5. Desenați un triunghi în interiorul căruia vă veți introduce numele secției de căre aparțineți( Informatică tehnică, Mechatronică, Electronică, Automatică).