Programación de Computadores: Java - Gráficas Orientadas a Objetos

Juan F. Pérez

Departamento MACC
Matemáticas Aplicadas y Ciencias de la Computación
Universidad del Rosario

juanferna.perez@urosario.edu.co

Segundo Semestre de 2017

Contenidos

Ejemplos Iniciales

Un ejemplo

```
package graficas;
import acm.graphics.*;
import acm.program.*;

public class GraficaV1 extends GraphicsProgram{
 public void run(){
  add(new GLabel("hola␣mundo"), 100, 50  );
 }
}
```

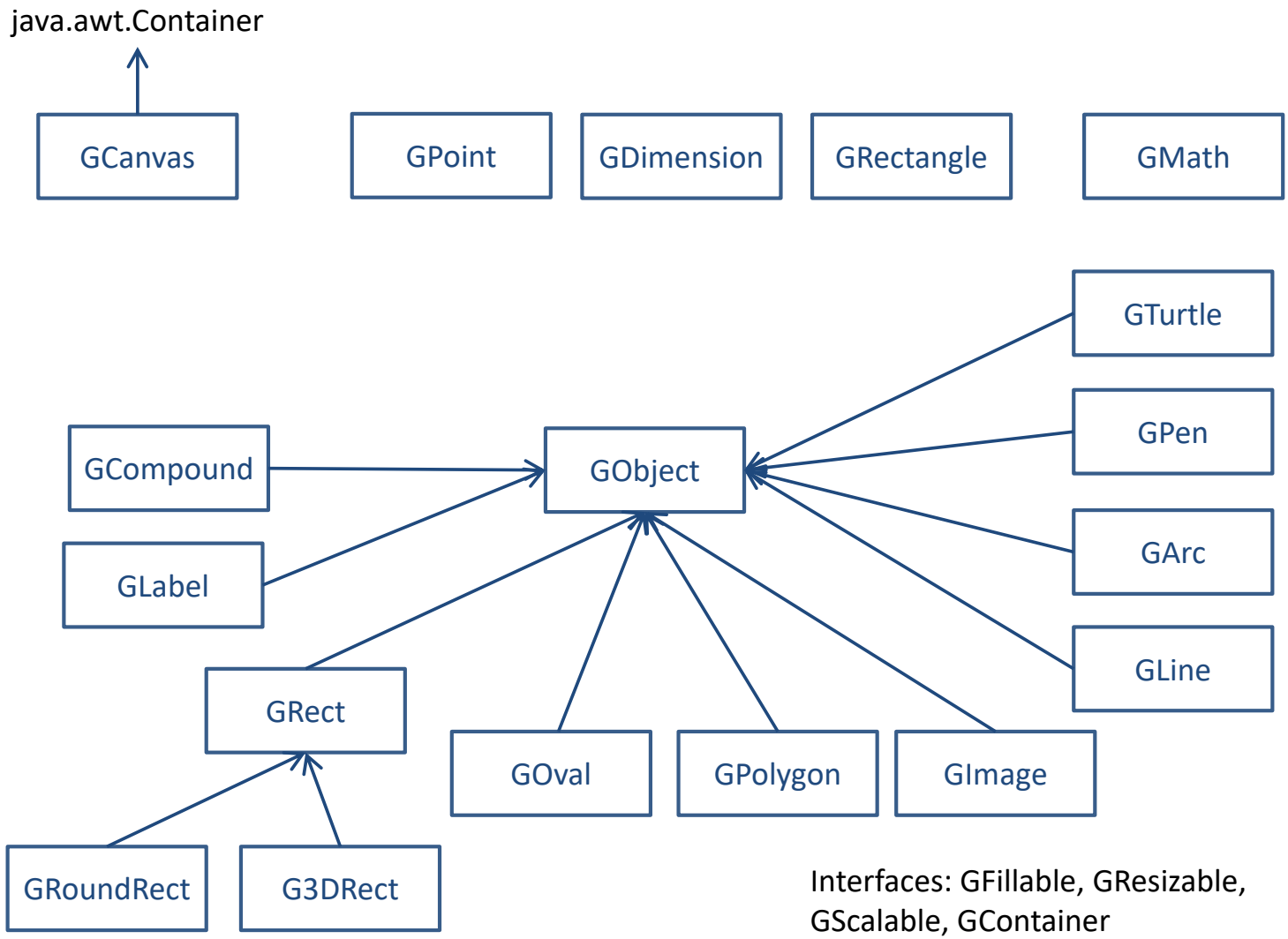
Otro ejemplo

```
package graficas;
import acm.graphics.*;
import acm.program.*;

public class GraficaV2 extends GraphicsProgram{
 public void run(){
  GLabel titulo = new GLabel("hola", 100, 50);
  add(titulo);
  for(int i = 0; i < 20; i++){
   titulo.setLabel("mundo");
   pause(300);
   titulo.setLabel("hola");
   pause(300);
  }
 }
}
```

El paquete acm.graphics

El paquete acm.graphics



El paquete acm.graphics

- `GCanvas`:
  - Contenedor donde se agregan y eliminan objetos gráficos.
  - Incluido al extender `GraphicsProgram`

El paquete acm.graphics

- `GCanvas`:
  - Contenedor donde se agregan y eliminan objetos gráficos.
  - Incluido al extender `GraphicsProgram`
- `GObject`:
  - Define objetos gráficos en general

El paquete acm.graphics

- `GCanvas`:
  - Contenedor donde se agregan y eliminan objetos gráficos.
  - Incluido al extender `GraphicsProgram`
- `GObject`:
  - Define objetos gráficos en general
  - `Clase abstracta`: define comportamiento y atributos de varias clases (que la extienden)

El paquete acm.graphics

- `GCanvas`:
  - Contenedor donde se agregan y eliminan objetos gráficos.
  - Incluido al extender `GraphicsProgram`
- `GObject`:
  - Define objetos gráficos en general
  - `Clase abstracta`: define comportamiento y atributos de varias clases (que la extienden)
  - No se pueden construir objetos de esta clase

El paquete acm.graphics

- `GCanvas`:
  - Contenedor donde se agregan y eliminan objetos gráficos.
  - Incluido al extender `GraphicsProgram`
- `GObject`:
  - Define objetos gráficos en general
  - `Clase abstracta`: define comportamiento y atributos de varias clases (que la extienden)
  - No se pueden construir objetos de esta clase
  - **NO**

```
GObject miObj = new GObject();
```

Algunos métodos en GCanvas y GraphicsProgram

`GCanvas` y `GraphicsProgram`:

1. `void add(GObject obj)`: agregar un objeto

Algunos métodos en GCanvas y GraphicsProgram

`GCanvas` y `GraphicsProgram`:

1. `void add(GObject obj)`: agregar un objeto
2. `void add(GObject obj, double x, double y)`: agregar un objeto en la posición (x, y) indicada

Algunos métodos en GCanvas y GraphicsProgram

`GCanvas` y `GraphicsProgram`:

1. `void add(GObject obj)`: agregar un objeto
2. `void add(GObject obj, double x, double y)`: agregar un objeto en la posición (x, y) indicada
3. `void remove(GObject obj)`: elimina un objeto

Algunos métodos en GCanvas y GraphicsProgram

`GCanvas` y `GraphicsProgram`:

1. `void add(GObject obj)`: agregar un objeto
2. `void add(GObject obj, double x, double y)`: agregar un objeto en la posición (x, y) indicada
3. `void remove(GObject obj)`: elimina un objeto
4. `setBackground(Color bg)`: cambia el color del fondo

Algunos métodos en GCanvas y GraphicsProgram

`GCanvas` y `GraphicsProgram`:

1. `void add(GObject obj)`: agregar un objeto
2. `void add(GObject obj, double x, double y)`: agregar un objeto en la posición (x, y) indicada
3. `void remove(GObject obj)`: elimina un objeto
4. `setBackground(Color bg)`: cambia el color del fondo

`GraphicsProgram`:

1. `void pause(double milisegundos)`: pausar ejecución

Algunos métodos en GCanvas y GraphicsProgram

`GCanvas` y `GraphicsProgram`:

1. `void add(GObject obj)`: agregar un objeto
2. `void add(GObject obj, double x, double y)`: agregar un objeto en la posición (x, y) indicada
3. `void remove(GObject obj)`: elimina un objeto
4. `setBackground(Color bg)`: cambia el color del fondo

`GraphicsProgram`:

1. `void pause(double milisegundos)`: pausar ejecución
2. `void waitForClick()`: suspende ejecución hasta que el usuario hac click

La clase Color del paquete java.awt

```
package graficas;
import java.awt.Color;
import acm.graphics.*;
import acm.program.*;

public class GraficaV3 extends GraphicsProgram{
 public void run(){
  GLabel titulo = new GLabel("hola", 100, 50);
  add(titulo);
  for(int i = 0; i < 20; i++){
   titulo.setLabel("mundo");
   setBackground(Color.BLUE);
   pause(300);
   titulo.setLabel("hola");
   setBackground(Color.WHITE);
   pause(300);
   }
  }
}
```

Definiendo nuevos colores

- Estándar RGB: red (rojo), green (verde), blue (azul)

Definiendo nuevos colores

- Estándar RGB: red (rojo), green (verde), blue (azul)
- Tres números para especificar la intensidad de cada color

Definiendo nuevos colores

- Estándar RGB: red (rojo), green (verde), blue (azul)
- Tres números para especificar la intensidad de cada color
- Cada número entre 0 y 255

Definiendo nuevos colores

- Estándar RGB: red (rojo), green (verde), blue (azul)
- Tres números para especificar la intensidad de cada color
- Cada número entre 0 y 255
- Cada número entre 00 y FF

Definiendo nuevos colores

```
package graficas;
import java.awt.Color;
import acm.graphics.*;
import acm.program.*;

public class GraficaV4 extends GraphicsProgram{
 public void run(){
  GLabel titulo = new GLabel("hola", 100, 50);
  add(titulo);
  for(int i = 0; i < 20; i++){
   titulo.setLabel("mundo");
   setBackground(new Color(128, 175, 100));
   pause(300);
   titulo.setLabel("hola");
   setBackground(Color.WHITE);
   pause(300);
  }
 }
}
```

Puntos y Rectángulos

- `GPoint(double x, double y)`: punto en la ubicación (x, y)

Puntos y Rectángulos

- `GPoint(double x, double y)`: punto en la ubicación (x, y)
- `GRectangle(double x, double y, double width, double height)`: rectángulo en la ubicación (x, y) con ancho *width* y alto *alto*

Puntos y Rectángulos

- `GPoint(double x, double y)`: punto en la ubicación (x, y)
- `GRectangle(double x, double y, double width, double height)`: rectángulo en la ubicación (x, y) con ancho *width* y alto *alto*
- Sirven para encapsular información (no se grafican)

Puntos y Rectángulos

- `GPoint(double x, double y)`: punto en la ubicación (x, y)
- `GRectangle(double x, double y, double width, double height)`: rectángulo en la ubicación (x, y) con ancho *width* y alto *alto*
- Sirven para encapsular información (no se grafican)
- Importante sobre las coordenadas:
  - Origen en la esquina superior izquierda

Puntos y Rectángulos

- `GPoint(double x, double y)`: punto en la ubicación (x, y)
- `GRectangle(double x, double y, double width, double height)`: rectángulo en la ubicación (x, y) con ancho *width* y alto *alto*
- Sirven para encapsular información (no se grafican)
- Importante sobre las coordenadas:
  - Origen en la esquina superior izquierda
  - Coordenada x crece de izquierda a derecha

Puntos y Rectángulos

- `GPoint(double x, double y)`: punto en la ubicación (x, y)
- `GRectangle(double x, double y, double width, double height)`: rectángulo en la ubicación (x, y) con ancho *width* y alto *alto*
- Sirven para encapsular información (no se grafican)
- Importante sobre las coordenadas:
  - Origen en la esquina superior izquierda
  - Coordenada x crece de izquierda a derecha
  - Coordenada y crece de arriba a abajo

La clase GMath

- Métodos útiles para graficar

La clase GMath

- Métodos útiles para graficar
- `double sinDegrees(double angle)`: seno de *angle* medido en grados

La clase GMath

- Métodos útiles para graficar
- `double sinDegrees(double angle)`: seno de *angle* medido en grados
- `double toDegrees(double radianes)`: convierte de radianes a grados

La clase GMath

- Métodos útiles para graficar
- `double sinDegrees(double angle)`: seno de *angle* medido en grados
- `double toDegrees(double radianes)`: convierte de radianes a grados
- `double toRadians(double grados)`: convierte de grados a radianes

La clase GMath

- Métodos útiles para graficar
- `double sinDegrees(double angle)`: seno de *angle* medido en grados
- `double toDegrees(double radianes)`: convierte de radianes a grados
- `double toRadians(double grados)`: convierte de grados a radianes
- `double distance(double x, double y)`: calcula la distancia del punto (x, y) al origen

La clase GMath

- Métodos útiles para graficar
- `double sinDegrees(double angle)`: seno de *angle* medido en grados
- `double toDegrees(double radianes)`: convierte de radianes a grados
- `double toRadians(double grados)`: convierte de grados a radianes
- `double distance(double x, double y)`: calcula la distancia del punto (x, y) al origen
- `double angle(double x, double y)`: calcula el ángulo del origen al punto (x, y) en grados

La clase GObject y sus subclases

La clase GObject

Clase abstracta:

- `void setLocation(double x, double y)`: fija la ubicación de este objeto en el punto (x, y)

La clase GObject

Clase abstracta:

- `void setLocation(double x, double y)`: fija la ubicación de este objeto en el punto (x, y)
- `void move(double x, double y)`: mueve este objeto x unidades horizontalmente y y unidades verticalmente

La clase GObject

Clase abstracta:

- `void setLocation(double x, double y)`: fija la ubicación de este objeto en el punto (x, y)
- `void move(double x, double y)`: mueve este objeto x unidades horizontalmente y y unidades verticalmente
- `double getWidth()`: retorna el ancho del objeto

La clase GObject

Clase abstracta:

- `void setLocation(double x, double y)`: fija la ubicación de este objeto en el punto (x, y)
- `void move(double x, double y)`: mueve este objeto x unidades horizontalmente y y unidades verticalmente
- `double getWidth()`: retorna el ancho del objeto
- `double getHeight()`: retorna el alto del objeto

La clase GObject

Clase abstracta:

- `void setLocation(double x, double y)`: fija la ubicación de este objeto en el punto (x, y)
- `void move(double x, double y)`: mueve este objeto x unidades horizontalmente y y unidades verticalmente
- `double getWidth()`: retorna el ancho del objeto
- `double getHeight()`: retorna el alto del objeto
- `void setColor(Color c)`: fija el color del objeto igual a c

La clase GObject

Clase abstracta:

- `void setLocation(double x, double y)`: fija la ubicación de este objeto en el punto (x, y)
- `void move(double x, double y)`: mueve este objeto x unidades horizontalmente y y unidades verticalmente
- `double getWidth()`: retorna el ancho del objeto
- `double getHeight()`: retorna el alto del objeto
- `void setColor(Color c)`: fija el color del objeto igual a c
- `Color getColor()`: retorna el color del objeto

La clase GObject

Clase abstracta:

- `void setLocation(double x, double y)`: fija la ubicación de este objeto en el punto (x, y)
- `void move(double x, double y)`: mueve este objeto x unidades horizontalmente y y unidades verticalmente
- `double getWidth()`: retorna el ancho del objeto
- `double getHeight()`: retorna el alto del objeto
- `void setColor(Color c)`: fija el color del objeto igual a c
- `Color getColor()`: retorna el color del objeto
- `void setVisible(boolean visible)`: fija el objeto como visible o no

La clase GObject

Clase abstracta:

- `void setLocation(double x, double y)`: fija la ubicación de este objeto en el punto (x, y)
- `void move(double x, double y)`: mueve este objeto x unidades horizontalmente y y unidades verticalmente
- `double getWidth()`: retorna el ancho del objeto
- `double getHeight()`: retorna el alto del objeto
- `void setColor(Color c)`: fija el color del objeto igual a c
- `Color getColor()`: retorna el color del objeto
- `void setVisible(boolean visible)`: fija el objeto como visible o no
- `boolean isVisible()`: retorna si el objeto es visible o no

La clase GRect

```
public void run(){
 GRect rec = new GRect(30, 40);
 add(rec, 10, 20);
 for(int i = 2; i <= 10; i++){
  add(rec, 10 + 10*i, 20+ 10*i);
  pause(300);
 }
 GRect rec2 = new GRect(30, 20);
 add(rec2, 100, 100);
 for(int i = 2; i <= 5; i++){
  add(rec2, 90 - 10*i, 100);
  pause(300);
 }
 rec2.setFilled(true);
 pause(300);
 rec2.setFillColor(Color.CYAN);
 pause(300);
 rec.scale(2.5);
}
```

La clase GLabel

- Cuadros de texto

La clase GLabel

- Cuadros de texto
- `new GLabel(String str, double x, double y)`: crea un cuadro con el texto *str* en la posición (x, y)

La clase GLabel

- Cuadros de texto
- `new GLabel(String str, double x, double y)`: crea un cuadro con el texto *str* en la posición (x, y)
- Posición: punto inferior izquierdo de la primera letra

La clase GLabel

- Cuadros de texto
- `new GLabel(String str, double x, double y)`: crea un cuadro con el texto *str* en la posición (x, y)
- Posición: punto inferior izquierdo de la primera letra
- Baseline: línea de escritura

La clase GLabel

- Cuadros de texto
- `new GLabel(String str, double x, double y)`: crea un cuadro con el texto *str* en la posición (x, y)
- Posición: punto inferior izquierdo de la primera letra
- Baseline: línea de escritura
- Height: distancia total entre lineas sucesivas

La clase GLabel

- Cuadros de texto
- `new GLabel(String str, double x, double y)`: crea un cuadro con el texto *str* en la posición (x, y)
- Posición: punto inferior izquierdo de la primera letra
- Baseline: línea de escritura
- Height: distancia total entre lineas sucesivas
- Ascent: máxima distancia de las letras por encima del baseline

La clase GLabel

- Cuadros de texto
- `new GLabel(String str, double x, double y)`: crea un cuadro con el texto *str* en la posición (x, y)
- Posición: punto inferior izquierdo de la primera letra
- Baseline: línea de escritura
- Height: distancia total entre lineas sucesivas
- Ascent: máxima distancia de las letras por encima del baseline
- Descent: máxima distancia de las letras por debajo del baseline

La clase GLabel

- Cuadros de texto
- `new GLabel(String str, double x, double y)`: crea un cuadro con el texto *str* en la posición (x, y)
- Posición: punto inferior izquierdo de la primera letra
- Baseline: línea de escritura
- Height: distancia total entre lineas sucesivas
- Ascent: máxima distancia de las letras por encima del baseline
- Descent: máxima distancia de las letras por debajo del baseline
- Fuente: tipo-estilo-tamaño:
  `myLabel.setFont(‘‘SansSerif-bold-24’’)`

La clase GLabel

```
package graficas;
import java.awt.Color;
import acm.graphics.*;
import acm.program.*;

public class GraficaV6 extends GraphicsProgram{
 public static final int APPLICATION_WIDTH = 400;
 public static final int APPLICATION_HEIGHT = 400;

 public void run(){
  GLabel lb = new GLabel("Esta es una etiqueta", 30, 40);
  add(lb);
  pause(300);
  lb.setFont("SansSerif-bold-20");
 }
}
```

Interfaces

Interfaces

- Definen un comportamiento a través de la definición de métodos, pero sin implementarlos

Interfaces

- Definen un comportamiento a través de la definición de métodos, pero sin implementarlos
- No implementan ningún método

Interfaces

- Definen un comportamiento a través de la definición de métodos, pero sin implementarlos
- No implementan ningún método
- Si una clase implementa una interfaz, tenemos seguridad de que posee un cierto comportamiento

Interfaces

- Definen un comportamiento a través de la definición de métodos, pero sin implementarlos
- No implementan ningún método
- Si una clase implementa una interfaz, tenemos seguridad de que posee un cierto comportamiento
- Ejemplos:

Interfaces

- Definen un comportamiento a través de la definición de métodos, pero sin implementarlos
- No implementan ningún método
- Si una clase implementa una interfaz, tenemos seguridad de que posee un cierto comportamiento
- Ejemplos:
  - `GFillable`: implementa e.g. `setFilled(boolean filled)`

Interfaces

- Definen un comportamiento a través de la definición de métodos, pero sin implementarlos
- No implementan ningún método
- Si una clase implementa una interfaz, tenemos seguridad de que posee un cierto comportamiento
- Ejemplos:
  - `GFillable`: implementa e.g. `setFilled(boolean filled)`
  - `GScalable`: implementa e.g. `scale(boolean filled)`

Interfaces

- Definen un comportamiento a través de la definición de métodos, pero sin implementarlos
- No implementan ningún método
- Si una clase implementa una interfaz, tenemos seguridad de que posee un cierto comportamiento
- Ejemplos:
  - `GFillable`: implementa e.g. `setFilled(boolean filled)`
  - `GScalable`: implementa e.g. `scale(boolean filled)`
  - `GContainer`: implementa e.g. `add(GObject obj)`

Interfaces

- Definen un comportamiento a través de la definición de métodos, pero sin implementarlos
- No implementan ningún método
- Si una clase implementa una interfaz, tenemos seguridad de que posee un cierto comportamiento
- Ejemplos:
  - `GFillable`: implementa e.g. `setFilled(boolean filled)`
  - `GScalable`: implementa e.g. `scale(boolean filled)`
  - `GContainer`: implementa e.g. `add(GObject obj)`
  - `GResizable`: implementa e.g. `void setSize(double width, double height)`

Interfaces

- Definen un comportamiento a través de la definición de métodos, pero sin implementarlos
- No implementan ningún método
- Si una clase implementa una interfaz, tenemos seguridad de que posee un cierto comportamiento
- Ejemplos:
  - `GFillable`: implementa e.g. `setFilled(boolean filled)`
  - `GScalable`: implementa e.g. `scale(boolean filled)`
  - `GContainer`: implementa e.g. `add(GObject obj)`
  - `GResizable`: implementa e.g. `void setSize(double width, double height)`
- `http://cs.stanford.edu/people/eroberts/jtf/javadoc/student/index.html`