**Gráficos en Java**

La clase *java.awt.Graphics* proporciona un mecanismo muy sencillo para dibujar formas. Sólo se ha de seleccionar el color y el modo de dibujo y llamar a métodos como *drawRect* o *fillRect* para dibujar un rectángulo o un rectángulo relleno.

El paquete Java 2D soporta muchas más opciones (importar *java.awt.geom.\**). Para dibujar una forma se han de seguir los pasos que sean necesarios de la siguiente lista:

* Obtener un objeto de la clase *Graphics2D*. Si se utiliza la versión adecuada del J2SE los métodos *paint* y *paintComponent* tiene como argumento un objeto de la clase *Graphics2D*, pero hay que hacer el siguiente *typecast*:
* public void paintComponent(Graphics g) {
* super.paintComponent(g);
* Graphics2D g2 = (Graphics2D) g;
* ...
* }
* Utilizar el método *setStroke* para establecer el trazo. El trazo se utiliza para dibujar el contorno de la forma. Se puede seleccionar el grosor y elegir entre líneas continuas o discontinuas.
* Stroke stroke = ...;
* g2.setStroke(stroke);
* Utilizar el método *setPaint* para establecer la forma de pintar. Se utiliza un objeto *paint* para rellenar áreas, o la parte interior de una forma. Se pueden crear colores sólidos, o con degradados o crear un relleno compuesto con una trama.
* Paint paint = ...;
* g2.setPaint(paint);
* Utilizar el método *setTransform* para establecer una transformación desde el espacio del usuario hacia el espacio de la aplicación. Se ha de utilizar una transformación si resulta más fácil definir formas en un sistema de coordenadas personal que utilizando unas coordenadas basadas en pixels.
* AffineTransform transform = ...;
* g2.setTransform(transform);
* Crear una forma usando uno de los métodos de la clase que implemente la interfaz *Shape*.
* Shape shape = ...;
* Dibujar la forma ya sea hueca o rellena.
* g2.draw(shape);
* g2.fill(shape);

En el paquete de graficos Java 2D se proporcionan las siguientes clases que implementan la interfaz *Shape* y que permiten crear objetos con esas formas: *Line2D*, *Rectangle2D*, *Ellipse2D*, *Arc2D*,*QuadCurve2D*, *CubicCurve2D* y *GeneralPath*.

Se pueden construir secuencias arbitrarias de segmentos de línea, curvas cuadráticas y curvas cúbicas y almacenarlos en un objeto del tipo *GeneralPath*. La primera coordenada de la figura (un camino, o ruta - *path*) se ha de especificar con el método *moveTo*. Después se sigue el camino llamando a los métodos *lineTo*, *quadTo* o *curveTo*. Por ejemplo:

GeneralPath path = new GeneralPath();

path.moveTo(10,20); // se establece la posición inicial

path.lineTo(20,30); // se dibuja un segmento de linea hasta el 30

El sistema de coordenadas que proporciona Java para dibujar no es el más adecuado para la representación de funciones matemáticas. En primer lugar no está orientado a la forma convencional de trabajar de utilizar los ejes de coordenadas con el origen en el centro y los cuatro cuadrantes. En segundo lugar los píxel no son las unidades más convenientes para trabajar. El paquete Java 2D ofrece un sistema para cambiar las coordenadas. Una transformación afín de un plano se puede definir de la forma:

$x' = ax + by + c$

$y'= dx + ey + f $

Las transformaciones de este tipo tienen la propiedad de que las líneas se transforman en líneas y los conjuntos de líneas paralelas se transforman en conjuntos de líneas paralelas. Para crear una transformación de este tipo usamos el constructor:

AffineTransform t = new AffineTransform(a, b, c, d, e, f);

Por ejemplo, el método *paintComponent* puede aplicar una transformación que le permita hacer que el origen esté en el centro del panel de visualización, para ello tiene que establecer la transformación de coordenadas a una traslación:

g2.translate(getWidth()/2, getHeight()/2);

Existen cuatro tipos de transformaciones fundamentales:

* Escalado: ampliación o reducción de todas las distancias a partir de un punto fijo.

*objetoGráfico.scale(double sx, double sy)*

* Rotación: rotación de todos los puntos a partir de un centro fijo.

*objetoGráfico.rotate(double a)*

* Traslación: desplazamiento de todos los puntos a una distancia determinada.

*objetoGráfico.translate(double tx, double ty)*

* Distorsión: manteniendo una línea fija, deslizar todas las líneas paralelas a ella una distancia proporcional a la distancia que hay desde la línea fija.

*objetoGráfico.shear(double sx, double xy)*

La operación *draw* de la clase *Graphics2D* dibuja el contorno de una forma utilizando el trazo que este predeterminado. Por defecto, el trazo es una línea sólida con un grosor de un píxel. Se puede establecer un trazo diferente llamando a *setStroke*. Para construir un objeto que modifique los valores de trazo por defecto se debe instanciar un objeto de la clase *BasicStroke*. Por ejemplo:

// Se establece el grosor de las líneas a 10 píxels

BasicStroke stroke = new BasicStroke(10.0f);

g2.setStroke(stroke);

g2.draw(new Line2D.Float(50,50,250,150));

Cuando los trazos tienen un grosor mayor de un píxel, se puede establecer la forma del extremo del trazo, el estilo de la unión, el ángulo en el que una unión a inglete se representa igual que una biselada, las porciones rellenas y blancas alternativas del trazo discontinuo y la fase de la discontinuidad. Por ejemplo:

BasicStroke stroke = new BasicStroke(

10f, // grosor: 10 píxels

BasicStroke.CAP\_BUTT, // terminación: recta

BasicStroke.JOIN\_ROUND, // unión: redondeada

1f, // ángulo: 1 grado

new float[] {10, 5, 5, 5}, // línea de 10, 5 blancos, línea de 5, 5 blancos

2 // fase

);