Gráficos avanzados - Ejercicios

Índice

1 Primitivas básicas..... 2
2 Texto..... 2
3 Imágenes y animación..... 3
4 Gráficos 3D..... 4

1. Primitivas básicas

Crea una nueva aplicación de Android `graficos` y haz que muestre una pantalla con:

a) Fondo gris oscuro.

b) Un marco de color azul con esquinas redondeadas. El marco debe tener un grosor de 2 pixels y se dejará un margen de 10 pixels con los bordes de la pantalla.

c) Un relleno de gradiente lineal para el marco, desde gris claro hasta blanco.

Pista

El gradiente se establece con el método `setShader`, que toma como parámetro un objeto de tipo `Shader`. Busca entre sus subclases para encontrar el *shader* apropiado.

2. Texto

Vamos a continuar trabajando con el proyecto del ejercicio anterior. Consideramos que el marco que habíamos dibujado en la pantalla es una página donde mostraremos texto. Se pide:

a) Añadir un texto centrado en la parte superior del marco como título de la página. El texto lo mostraremos con una letra de tamaño 16, de color rojo, en negrita, subrayado, y con antialiasing.

b) Ahora vamos a añadir texto en múltiples líneas. Para ello vamos a crearnos un método que reciba como parámetro una cadena de texto larga y el lienzo donde dibujarla, y se encargue de cortar el trozo en múltiples líneas y las dibuje en pantalla utilizando la separación recomendada. El texto no debe salirse de los límites del marco creado en el ejercicio anterior, incluso se debería dejar algo de margen (unos 5 pixels al menos).

Importante

Recuerda que las métricas del texto pueden tener signo positivo o negativo según correspondan a un ascenso o descenso.

c) De forma opcional, podemos mejorar el método anterior haciendo que el texto se corte sólo donde encuentre un carácter de espacio, siempre antes de exceder el ancho de línea. Si no encuentra ninguno en la línea, cortará en el carácter justo antes de llegar a dicho límite.

Ayuda

Deberemos ayudarnos de métodos de la clase `String` para manipular la cadena de texto. Podemos por ejemplo obtener una subcadena con `substring` o consultar el carácter en una

determinada posición con `charAt`. Consulta la especificación de estos métodos en la documentación de Java para saber cómo utilizarlos.

3. Imágenes y animación

En las plantillas de la sesión se proporcionan varias imágenes con los fotogramas de una animación. Se pide:

a) Crear una nueva aplicación `animacion` y mostrar en ella un `ImageView` que por el momento simplemente muestre uno de los fotogramas de la animación como imagen fija. Define dicho componente en el XML `main.xml`.

b) Definir la animación por fotogramas en XML. Reproducir dicha animación en el `ImageView`.

Recuerda

No se puede poner en marcha la animación en el método `onCreate`. Una posible solución es guardar la animación (`AnimationDrawable`) como campo global de la clase, y llamar a su método `start` cuando la actividad obtenga el foco (`onWindowFocusChanged`).

Ayuda

La animación se mostrará como fondo del `ImageView` (`setBackgroundDrawable`). Si se muestra como imagen (`setImageDrawable`) no se reproducirá correctamente. Para evitar que la imagen se deforme, podemos hacer que el *layout* del `ImageView` se adapte a su contenido (en `main.xml` indicaremos como *layout* `wrap_content`, tanto para la altura como para la anchura).

b) Añadir una animación por interpolación que haga que el `ImageView` anterior rote continuamente alrededor de la pantalla (como si el armadillo estuviese caminando en círculos).

Ayuda

Se recomienda consultar la documentación sobre recursos de animación. Concretamente nos fijaremos en *tween animation* de tipo *rotate*. Las propiedades *pivot* nos sirven para fijar el punto alrededor del cual se producirá la rotación. También nos será de utilidad especificar el interpolador a utilizar. Nos interesará utilizar un interpolador lineal, para que no acelere ni frene al comienzo y al final de la animación, respectivamente. También necesitaremos hacer que la animación se reproduzca indefinidamente, lo cual se tendrá que hacer de forma programática (métodos `setRepeatCount` y `setRepeatMode` de la clase `Animation`, una vez haya sido obtenida en nuestro código.

c) De forma opcional, se puede probar a implementar las animaciones de forma programática, en lugar de XML.

4. Gráficos 3D

En las plantillas de la sesión tenemos una aplicación `graficos` en la que podemos ver un ejemplo completo de cómo utilizar `SurfaceView` tanto para gráficos 2D con el `Canvas` como para gráficos 3D con OpenGL, y también de cómo utilizar `GLSurfaceView`.

a) Si ejecutamos la aplicación veremos un triángulo rotando alrededor del eje Y. Observar el código fuente, y modificarlo para que el triángulo rote alrededor del eje X, en lugar de Y.

b) También podemos ver que hemos creado, además de la clase `Triangulo3D`, la clase `Cubo3D`. Modificar el código para que en lugar de mostrar el triángulo se muestre el cubo.