Gráficos avanzados - Ejercicios

Índice

Copyright © 2012 Depto. CCIA All rights reserved.

1. Personalización del aspecto

Crea una nueva aplicación de Android llamada `Drawables` y haz que muestre un `TextView` que ocupe toda la pantalla y que tenga el siguiente aspecto:

- Un marco de color azul con esquinas redondeadas.
- El marco debe tener un grosor de 2 *density pixels*.
- Se dejará un margen de 10 *density pixels* entre el marco y el texto que contiene
- Se dejará un margen de 5 *density pixels* entre los bordes de la pantalla y el marco. ¿Encontramos algún atributo que haga esto en la definición del *drawable*? ¿Y en el *layout*?
- Un relleno de gradiente lineal para el marco, desde gris claro hasta blanco
- El texto será de color negro.

Ayúdate de la documentación oficial de Android para crear el recurso:

http://developer.android.com/guide/topics/resources/drawable-resource.html#Shape

2. Personalización de botones

Vamos a añadir un botón en la parte inferior de la pantalla para la aplicación anterior, que llenará todo el ancho. Se proporcionan en las plantillas de la sesión tres imágenes, una para cada uno de los estados del botón. Se pide:

a) Dar inicialmente al botón como aspecto la imagen correspondiente al estado sin pulsar.

b) Crear un *state list drawable* para que el botón cambie su aspecto para cada estado.

c) Convertir las imágenes a *nine-patch* y utilizar esta nueva versión en la aplicación.

Nos podemos ayudar de la documentación sobre *state list drawable*:

http://developer.android.com/guide/topics/resources/drawable-resource.html#StateList

3. Animaciones por fotogramas

Vamos a añadir a la pantalla de nuestra aplicación una vista de tipo `ProgressBar` que muestre que se está realizando un progreso de forma indeterminada. Se pide:

a) Ejecutar la aplicación y comprobar como aparece un círculo girando continuamente como indicativo de que se está realizando un progreso. Hacer que el indicador de progreso aparezca centrado en la horizontal (mirar el atributo `layout_gravity`).

b) Vamos ahora a personalizar la animación que figura en el progreso. En las plantillas de la sesión tenemos cuatro fotogramas de un armadillo caminando. Crea un *animation-list*

Copyright © 2012 Depto. CCIA All rights reserved.

drawable a partir de dichos fotogramas, que se reproduzca de forma continuada cambiando de fotograma cada 200ms.

c) Reemplaza la animación por defecto del progreso por la animación que hemos definido nosotros. Pista: se debe especificar como *drawable*, y el progreso continuo indeterminado (sin dar un porcentaje concreto) que queremos mostrar, se conoce en la API de Android como *indeterminate progress*. Busca entre los atributos de `ProgressBar` el que consideres adecuado.

Nos podemos ayudar de la documentación sobre *animation list drawable*:

http://developer.android.com/guide/topics/resources/animation-resource.html#Frame

4. Niveles

En este ejercicio vamos a añadir una barra de de selección de nivel (`SeekBar`), que podría sernos útil por ejemplo como control de volúmen, o para desplazarnos en la reproducción de vídeo o audio. Se pide:

a) Añadir un `SeekBar` que ocupe todo el ancho de la pantalla a nuestra aplicación y comprobar que se muestra correctamente y que al pulsar sobre ella podemos mover el cursor. Por defecto, el nivel de la barra va de 0 a 10000.

b) Vamos a personalizarla para que el fondo de la barra cambie de color según el nivel seleccionado. Los colores que debe mostrar son:

- **Verde** (`#FF00FF00`): Niveles de 0 a 2500.
- **Amarillo** (`#FFFFFF00`): Niveles de 2501 a 5000.
- **Naranja** (`#FFFF8800`): Niveles de 5001 a 7500.
- **Rojo** (`#FFFF0000`): Niveles de 7501 a 10000.

Crea un *drawable* de tipo rectángulo con esquinas redondeadas para cada estado. Introdúcelos en un *level list drawable*, y establécelo como `progressDrawable` en la barra.

Nos podemos ayudar de la documentación sobre *level list drawable*:

http://developer.android.com/guide/topics/resources/drawable-resource.html#LevelList

5. Creación de componentes propios

Vamos a crear un nuevo componente que muestre un gráfico de tipo tarta para indicar un determinado porcentaje. Dentro del círculo, aparecerá en rojo el sector correspondiente al porcentaje indicado, y en azul el resto. Para ello crearemos una nueva aplicación Android de nombre `Grafica`. Se pide:

a) Crear una vista con la gráfica que muestre por defecto un porcentaje del 25%.

Copyright © 2012 Depto. CCIA All rights reserved.

Mostrarla en la pantalla de forma programática.

b) Sobreescribir el método `onMeasure`, de forma que tenga un tamaño preferido de 50x50. Al ejecutarlo, ¿ha cambiado algo? ¿Por qué? (Pista: piensa en los parámetros de *layout* que se deben estar aplicando por defecto al introducir el componente de forma programática).

c) Declarar la vista en el XML, para incluirla en el *layout* creado por defecto para nuestra aplicación.

Atención

En este punto es necesario haber definido todos los constructores de la superclase `View`, ya que al inicializarlo desde XML podrá utilizar cualquiera de ellos según la información proporcionada.

d) Añadir un atributo `percentage` propio a la vista, e indicar dicho porcentaje en el XML.

e) Añade varias gráficas al *layout*, con diferentes porcentajes cada una de ellas.

f) Hacer que la gráfica se muestre con un gradiente radial, con un color más oscuro en el centro y más claro en los bordes.

Nota

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.

g) Añadir un borde a la figura del mismo color que el central y con grosor 2.

Ayuda

Primero deberemos dibujar el relleno, y luego el borde. Para ello utilizaremos `setStyle`.

6. Texto y métricas

Continuando con el código desarrollado en el ejercicio anterior, vamos ahora a mostrar también con texto el porcentaje que representa la gráfica. Este texto debe aparecer centrado horizontalmente y en la parte inferior, sin que se pueda llegar a cortar ninguna letra por el borde inferior de la vista. Añadir *antialiasing* al texto.

7. 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`

Copyright © 2012 Depto. CCIA All rights reserved.

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.

Copyright © 2012 Depto. CCIA All rights reserved.

Copyright © 2012 Depto. CCIA All rights reserved.