Consumo de API REST

Imágenes en Servidores (Parte III)

{desafío} latam_

Construir una aplicación Android que consume un servicio REST actualizando la interfaz de usuario, acorde al lenguaje Kotlin y a la librería Retrofit

- Unidad 1:
  Acceso a datos en Android
- Unidad 2:
  Consumo de API REST
- Unidad 3:
  Testing
- Unidad 4:
  Distribución del aplicativo Android

{desafío} latam_

¿Qué aprenderás en esta sesión?

- *Ventajas y desventajas: Glide, Fresco, Picasso*
- *¿Cómo usar Picasso?*

¿Puedes explicar el proceso de descarga y visualización de imágenes desde un servidor remoto a una app en Android?

{desafío} latam_

/* Introducción */

Introducción

¿Cómo funciona la gestión remota de imágenes en Android?

Anteriormente vimos que, cuando necesitamos mostrar una imagen guardada de forma local, basta manejarla como un recurso, dependiendo de donde se encuentre guardada, deberemos especificar la ruta y la vista en la cual queremos mostrarla y Android hará el resto.

Sin embargo, cuando se trata de mostrar imágenes que son parte de la aplicación, pero están guardadas de forma remota, el escenario cambia.

Recuerda, siempre que necesites hacer una petición a un servicio externo, debe agregar el permiso ”<uses-permission android:name="android.permission.INTERNET" />”

¿Cómo funciona la gestión remota de imágenes en Android?

1. **Solicitud de la imagen**: se realiza una solicitud de imagen desde la aplicación de Android a un servidor o CDN (red de entrega de contenido) que sirve la imagen. Esto se puede hacer usando una biblioteca de red como Retrofit.
2. **Carga de la imagen**: una vez que se realiza la solicitud, los datos de la imagen se recuperan y se cargan en la memoria del dispositivo Android.

¿Cómo funciona la gestión remota de imágenes en Android?

3. **Visualización de la imagen**: los datos de la imagen se procesan y se muestran en la pantalla del dispositivo. Esto se puede hacer usando una biblioteca de imágenes como Glide, Picasso o Fresco, que manejan la carga y decodificación de los datos de la imagen, así como el almacenamiento en caché y otras optimizaciones de rendimiento.

4. **Almacenamiento en caché de la imagen**: para mejorar el rendimiento, es común almacenar imágenes en caché localmente en el dispositivo para que puedan reutilizarse sin tener que solicitarlas nuevamente. Esto ayuda a reducir el uso de la red y acelera la visualización de imágenes.

/* Ventajas y desventajas: Glide, Fresco, Picasso */

Ventajas y desventajas: Glide

Ventajas:

- Fácil de usar y tiene una API simple.
- Puede manejar una amplia gama de tipos de imágenes (incluidos GIF y WebP).
- Tiene soporte incorporado para transformaciones y recortes de imágenes.
- Puede almacenar automáticamente imágenes en caché para mejorar el rendimiento.

Desventajas:

- Puede tener un uso de memoria ligeramente mayor que otras bibliotecas.
- Algunos desarrolladores han reportado errores ocasionales con ciertos tipos de imágenes.

Ventajas y desventajas: Fresco

Ventajas:

- Tiene un sistema de gestión de memoria altamente eficiente.
- Puede manejar imágenes muy grandes con facilidad.
- Puede almacenar automáticamente imágenes en caché para mejorar el rendimiento.
- Tiene soporte incorporado para transformaciones y recortes de imágenes.

Contras:

- Tiene una API más compleja que otras bibliotecas.
- Puede ser más difícil de instalar y configurar.

Ventajas y desventajas: Picasso

Ventajas:

- Tiene una API simple y fácil de usar.
- Puede almacenar automáticamente imágenes en caché para mejorar el rendimiento.
- Tiene soporte incorporado para transformaciones y recortes de imágenes.

Contras:

- Puede tener un uso de memoria ligeramente mayor que otras bibliotecas.
- No es compatible con una amplia gama de tipos de imágenes como otras bibliotecas (no gif o webp).

Recuerda: en última instancia, la elección entre estas bibliotecas dependerá de las necesidades específicas de tu proyecto. Glide y Picasso son las bibliotecas más populares y mejor documentadas, y Fresco es más para aplicaciones de alto rendimiento con imágenes grandes.

/* ¿Cómo usar Picasso? */

¿Cómo usar Picasso?

Para usar Picasso en un proyecto de Android, primero deberá agregar la biblioteca como una dependencia en su archivo build.gradle. Puede hacer esto agregando la siguiente línea a la sección de dependencias:

```
implementation 'com.squareup.picasso:picasso:2.8.2'
```

Una vez que haya agregado la dependencia, puede usar Picasso para cargar una imagen desde una URL o ruta de archivo en un ImageView como este:

```
ImageView imageView;
//...
String imageUrl = "http://example.com/image.jpg";
Picasso.get().load(imageUrl).into(imageView);
```

¿Cómo usar Picasso?

También puede especificar la imagen de marcador de posición mientras carga la imagen

```
String imageUrl = "http://example.com/image.jpg";
Picasso.get().load(imageUrl).placeholder(R.drawable.placeholder).into(imageView);
```

También puede usar el método `fit()` para reducir la escala de la imagen para que quepa en ImageView manteniendo la relación de aspecto, o el método `centerInside()` para reducir la imagen de modo que ocupe completamente ImageView manteniendo la relación de aspecto.

También puede usar el método `error()` para especificar una imagen para mostrar si la imagen no se carga.

```
String imageUrl = "http://example.com/image.jpg";
Picasso.get().load(imageUrl).error(R.drawable.error).into(imageView);
```

Estos son algunos de los usos básicos de Picasso, tiene muchas más funciones para uso avanzado como el almacenamiento en caché y el manejo prioritario de solicitudes.

Ejercicio
"¿Cómo usar Glide?"

¿Cómo usar Glide?

Crea un proyecto con un activity vacío y sigue los siguientes pasos:

1. Agrega la biblioteca como una dependencia en tu archivo build.gradle

```
implementation 'com.github.bumptech.glide:glide:4.11.0'
```

2. Una vez que haya agregado la dependencia, puedes usar Glide para cargar una imagen desde una URL o ruta de archivo en un ImageView como este:

```
ImageView imageView;
String imageUrl = "http://example.com/image.jpg";
Glide.with(context).load(imageUrl).into(imageView);
```

3. Reemplaza "http://example.com/image.jpg" con la URL de la imagen que quieras mostrar.

4. Recuerda agregar el permiso para Internet.

/* Otras formas de usar Glide */

Otras formas de usar Glide

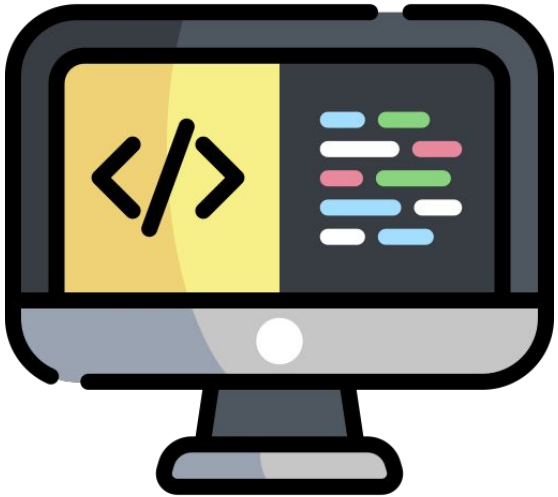
También puede especificar varias opciones para cargar y mostrar imágenes, como cambiar el tamaño, recortar y transformar. Aquí hay un ejemplo de cómo puede usar Glide para cambiar el tamaño de una imagen antes de mostrarla en ImageView:

```
String imageUrl = "http://example.com/image.jpg";
Glide.with(context)
    .load(imageUrl)
    .override(500, 500) // fija alto y ancho
    .centerCrop()
    .into(imageView);
```

También puedes especificar un placeholder mientras la imagen carga:

```
String imageUrl = "http://example.com/image.jpg";
Glide.with(context)
    .load(imageUrl)
    .placeholder(R.drawable.placeholder)
    .into(imageView);
```

Otras formas de usar Glide



Estos son algunos de los usos básicos de Glide, tiene muchas más funciones para uso avanzado como almacenamiento en caché, manejo prioritario de solicitudes e incluso soporte para imágenes animadas.

También debe tener en cuenta que cuando use Glide, es importante borrar los recursos cuando ya no los necesite usando el método `clear()` o el adaptador `RecyclerView`.

El manejo de imágenes en Android es una tarea compleja. Sin embargo, al aprovechar las herramientas y técnicas adecuadas, es posible crear apps visualmente impactantes y de alto rendimiento que brindan una experiencia de usuario atractiva.

Próxima sesión...

- *Guía de ejercicios*

{desafío}
latam_

{desafío}
latam_

Academia de
talentos digitales