Logotipo

Descripción generada automáticamente con confianza mediaImagen que contiene Icono

Descripción generada automáticamente

Instituto Politécnico Nacional

Escuela Superior de Cómputo

Desarrollo de Aplicaciones Móviles Nativas

**“Retrofit”**

Alumno:

Malagón Baeza Alan Adrian

Profesor:

M. en C. José Asunción Enríquez Zárate

Grupo: 7CM1

**Introducción**

En el marco de mi práctica, he desarrollado una aplicación en Android que utiliza la API TheCatAPI para buscar y mostrar imágenes de gatos en un RecyclerView. A lo largo de este informe, presentaré el código relevante y explicaré cómo se implementaron las funcionalidades clave.

El código principal se encuentra en la clase MainActivity, la cual hereda de AppCompatActivity e implementa OnQueryTextListener. Esta clase es responsable de gestionar la interfaz de usuario y las interacciones con la API.

La funcionalidad principal de la aplicación radica en el método buscarPorNombre, el cual utiliza Retrofit para realizar solicitudes asíncronas a la API TheCatAPI. Las respuestas obtenidas se procesan y se muestran en el RecyclerView a través del adaptador GatoAdapter.

**Desarrollo**

class MainActivity : AppCompatActivity(), OnQueryTextListener {  
  
 private lateinit var binding: ActivityMainBinding  
 private lateinit var adapter: GatoAdapter  
 private val imagenesDeGatos = *mutableListOf*<String>()  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 binding = ActivityMainBinding.inflate(*layoutInflater*)  
 setContentView(binding.*root*)  
  
 binding.svGatos.setOnQueryTextListener(this)  
 iniciarRecyclerView()  
 }  
  
 fun obtenerRetrofit(): Retrofit{  
 return Retrofit.Builder()  
 .baseUrl("https://api.thecatapi.com/v1/images/")  
 .addConverterFactory(GsonConverterFactory.create())  
 .build()  
 }  
  
 private fun buscarPorNombre(query: String){  
 *CoroutineScope*(Dispatchers.IO).*launch* **{** val call: Response<List<GatosResponse>> = obtenerRetrofit()  
 .create(APIService::class.*java*)  
 .obtenerGatosPorRazas("search?limit=25&breed\_ids=$query&api\_key=live\_Z7b28jk1m38u0te0tPHxHfwcqM11dqSEPKIX2KmitFH2Q5yJmXQMBglTGVkjOej0")  
  
 runOnUiThread()**{** if(call.*isSuccessful*){  
 // mostrar recycler View  
 val gatitos: List<GatosResponse>? = call.body()  
 val imagenes: List<String> = gatitos?.*mapNotNull* **{ it**.url **}** ?: *emptyList*()  
 imagenesDeGatos.clear()  
 imagenesDeGatos.addAll(imagenes)  
 *print*("uwu"+imagenes.toString())  
 adapter.notifyDataSetChanged()  
 }else{  
 //mostrar errores  
 mostrarErrores()  
 }  
 ocultarTeclado()  
 **}  
  
 }** }  
  
 private fun mostrarErrores(){  
 Toast.makeText(this,"error", Toast.*LENGTH\_SHORT*).show()  
 }  
  
 override fun onQueryTextSubmit(query: String?): Boolean {  
 if(!query.*isNullOrEmpty*()){  
 buscarPorNombre(query.*lowercase*())  
 }  
 return true  
 }  
  
 override fun onQueryTextChange(newText: String?): Boolean {  
 return true  
 }  
  
 private fun iniciarRecyclerView() {  
 adapter = GatoAdapter(imagenesDeGatos)  
 binding.rvGatos.*layoutManager* = LinearLayoutManager(this)  
 binding.rvGatos.*adapter* = adapter  
 }  
  
 private fun ocultarTeclado(){  
 val imm = getSystemService(*INPUT\_METHOD\_SERVICE*) as InputMethodManager  
 imm.hideSoftInputFromWindow(binding.vistaPrincipal.*windowToken*,0)  
 }  
}

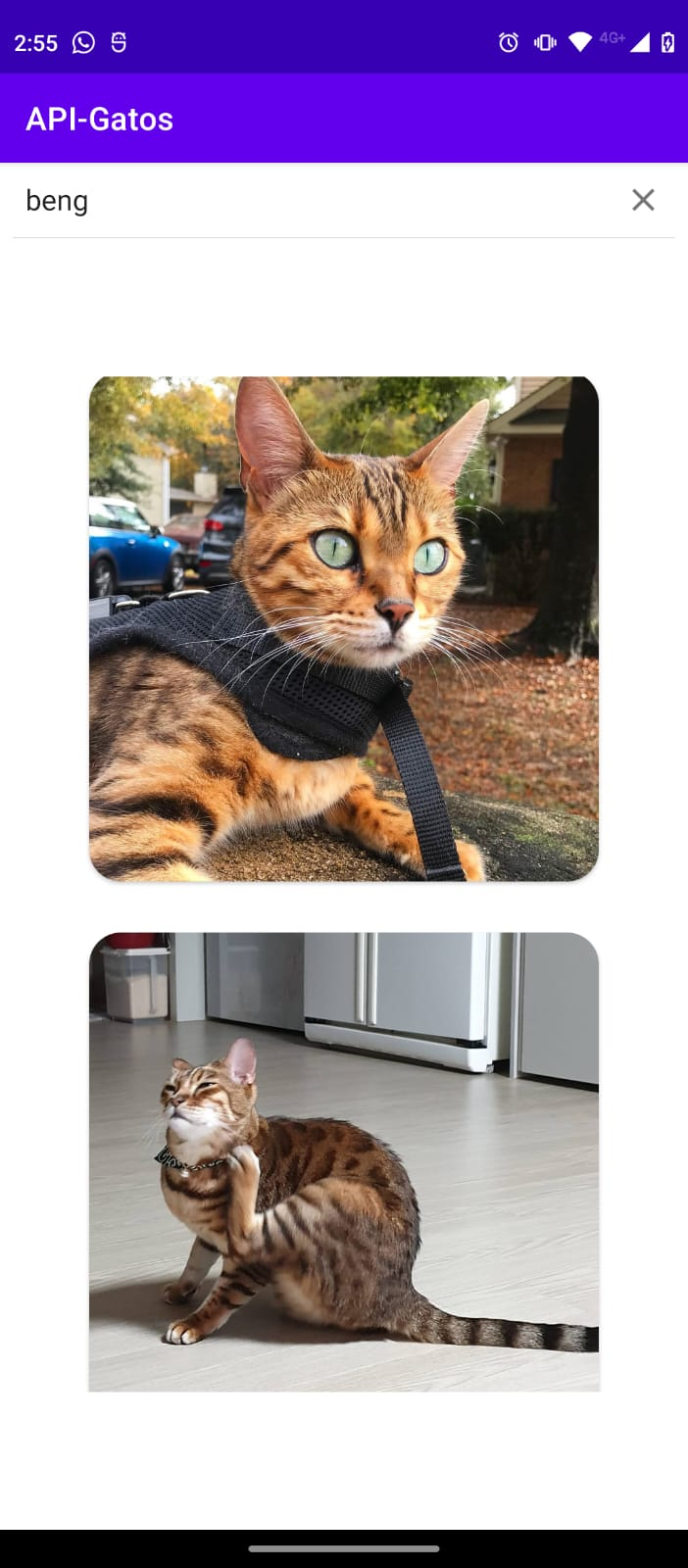
interface APIService {  
 @GET  
 suspend fun obtenerGatosPorRazas(@Url url: String): Response<List<GatosResponse>>  
}

data class GatosResponse (  
 @SerializedName("url")  
 var url: String  
)

class GatosViewholder (view: View): RecyclerView.ViewHolder(view) {  
  
 private val binding = ElementoGatoBinding.bind(view)  
  
 fun bind(imagen: String){  
 Picasso.get().load(imagen).into(binding.ivGato)  
 }  
}

class GatoAdapter (val imagenes:List<String>): RecyclerView.Adapter<GatosViewholder>(){  
 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): GatosViewholder {  
 val layoutInflater : LayoutInflater = LayoutInflater.from(parent.*context*)  
 return GatosViewholder(layoutInflater.inflate(R.layout.*elemento\_gato*,parent,false))  
 }  
  
 override fun getItemCount(): Int {  
 return imagenes.size  
 }  
  
 override fun onBindViewHolder(holder: GatosViewholder, position: Int) {  
 val elemento: String = imagenes[position]  
 holder.bind(elemento)  
 }  
}

**Resultados**

****

**Conclusiones**

En conclusión, el desarrollo de esta aplicación en Android para buscar y mostrar imágenes de gatos utilizando la API TheCatAPI ha sido una experiencia enriquecedora durante mi práctica. He aprendido a utilizar Retrofit para comunicarme con una API externa y a mostrar los resultados en un RecyclerView mediante el adaptador GatoAdapter. Esta práctica me ha permitido adquirir habilidades importantes en el desarrollo de aplicaciones móviles y comprender la importancia de integrar APIs en el proceso.

Además, el código presentado demuestra cómo implementar características clave, como la búsqueda por nombre de raza de gato, la carga y visualización de imágenes con Picasso, y la administración de las interacciones del usuario a través de OnQueryTextListener.

En resumen, esta práctica ha contribuido significativamente a mi conocimiento y habilidades en el desarrollo de aplicaciones móviles en Android, al tiempo que me ha permitido explorar el uso de APIs externas para enriquecer las funcionalidades de la aplicación.