





Requisições Simuladas e Manipulação de JSON



Objetivo: Simular consumo de API usando Gson, converter JSON para objetos

Kotlin, e exibir os dados com Jetpack Compose.

JSON: Formato leve para trocar dados entre sistemas. Muito usado em APIs e apps móveis. Estrutura: chave-valor

Gson no Kotlin: É uma biblioteca que transforma: JSON → Objeto Kotlin | Objeto

Kotlin → JSON

Atenção:

A estrutura JSON

precisa bater com

a data class.

```
val json = """{"nome":"Ana", "idade":28}"""

val cliente = gson.fromJson(json, Cliente::class.java)
```

val novoJson = gson.toJson(cliente)

Campos nulos ou extras devem ser tratados com atenção.



Para usar Gson no projeto



1 – vá para build.gradle.kts (Module) > dependencies

Acrescente a linha de implementação.

Rebuild gradle.

Q

```
com.example.test (andr
                                                                  dependencies {
                                                           42
                                   com.example.test (test)
                                res
                                                                       implementation(libs.androidx.core.ktx)
                                                           44
                                                                       implementation(libs.androidx.lifecycle.runtime.ktx)
                                res (generated)
                                                           45
                                                                       implementation(libs.androidx.activity.compose)
                              @ Gradle Scripts
                                                           40
                                                                       implementation(platform(libs.androidx.compose.bom))
                                E build.gradle.kts (Project: T
                                                                       implementation(libs.androidx.ui)
                                                           48
                                R build.gradle.kts (Module :a
                                                                       implementation(libs.androidx.vi.graphics)
                                                           49

≡ proguard-rules.pro (ProGu

                                                                       implementation(libs.androidx.ui.tooling.preview)
                                                           50
                                gradle.properties (Project
                                                                       implementation(libs.androidx.material3)
                                                           51
                                 gradle-wrapper.properties
                                                           52
                                                                       testImplementation(libs.junit)
                                 Ibs.versions.toml (Version)
                                                           53
                                                                       androidTestImplementation(libs.androidx.junit)
                                (SDK Local
                                                                       androidTestImplementation(libs.androidx.espresso.cor
                                                           54
                                er settings.gradle.kts (Project
                                                           55
                                                                       androidTestImplementation(platform(libs.androidx.com
                                                                       androidTestImplementation(libs.androidx.ui.test.juni
                                                           56
                                                                       debugImplementation(libs.androidx.vi.tooling)
                                                           57
Sync Project with Gradle Files Ctrl+Shift+O
                                                                       debugImplementation(libs.androidx.vi.test.manifest)
                                                           58
                                                                       implementation("com.google.code.gson:gson:2.10.1")
                                                           59
                                                           68
```

Para usar Gson no projeto



2 – Depois de instalar a dependência, basta importar a classe no seu código:

import com.google.gson.Gson

Agora você já pode usar:

```
val gson = Gson()
val cliente = gson.fromJson(json, Cliente::class.java)
```



Integração com Compose



Usando dados dinâmicos em @Composable:

```
@Composable
fun ExibirCliente(clienteJson: String) {
    val gson = remember { Gson() }
    val cliente = gson.fromJson(clienteJson, Cliente::class.java)
   Column(modifier = Modifier.padding(16.dp)) {
       Text("Nome: ${cliente.nome}")
        Text("Idade: ${cliente.idade}")
```

Remember: objeto será criado apenas uma vez, e nas recomposições seguintes, a mesma instância será reutilizada.



Integração com Compose



De Classe para JSON:

```
val novoJson = gson.toJson(cliente)
Text("JSON gerado: $novoJson")
```

1 - Criando STRING JSON:

2 - Converter e exibir:

```
val jsonProdutos = """
[
    {"nome": "Caneta", "preco": 2.0},
    {"nome": "Caderno", "preco": 10.0},
    {"nome": "Lápis", "preco": 1.5}
]
""".trimIndent()
```

```
val gson = remember { Gson() }
val lista = gson.fromJson(jsonProdutos, Array<Produto>::class.java).toList()

LazyColumn {
   items(lista.size) { i ->
      val item = lista[i]
      Text("${item.nome} - R$ ${item.preco}")
   }
}
```





FPF tech

Vamos separar em pastas models, services, views

Models -> as classes até agora.

Services -> crie um novo chamado *ProdutoService*

Views -> Todas as listagens (exceto o MainActivity)

Utilizar ClassProduto

Criar string JSON com produtos

Converter string JSON em lista de objetos

Exibir os produtos em uma LazyColumn

Gerar novo JSON a partir de um objeto







FPF tech

1 - ProdutoService

```
package com.example.test.services
ifests
                              import com.example.test.models.Produto
                        3
in+java
om.example.test
                        5
                              val jsonProdutos = """
models
                        ő
services
                                {"nome": "Caneta", "preco": 2.0, "emPromocao": true},

    □ produtoService.kt

                                {"nome": "Caderno", "preco": 10.0, "emPromocao": true},
 ui.theme
                        9
                                {"nome": "Lápis", "preco": 1.5, "emPromocao": true}
 views
                       10
                              """.trimIndent()
 11
                       12
 val objProdudos = listOf(
                       13
  RessoaCard.kt
                                  Produto("Camiseta", 59.90, true),
                       14
  Produto("Tênis", 199.90, false),
                       15

    ✓ MainActivity.kt

                                  Produto("Boné", 39.90, true)
                       16
om.example.test (androidT
                       17
om.example.test (test)
```

package com.example.test.views Mão na Mass 2 ifests import androidx.compose.foundation.layout.Column 2- ListaAPIprodutosn+java 3 import androidx.compose.foundation.layout.Spacer 4 :om.example.test import androidx.compose.foundation.layout.fillMaxWidth S models import androidx.compose.foundation.layout.height ô. services import androidx.compose.foundation.layout.padding ui.theme import androidx.compose.foundation.lazy.LazyColumn 8 i views import androidx.compose.material3.Card 9 10 import androidx.compose.material3.CardDefaults import androidx.compose.material3.MaterialTheme 11 import androidx.compose.material3.Text 12 □ PessoaCard.kt import androidx.compose.runtime.Composable 13 import androidx.compose.runtime.remember 14 import androidx.compose.ui.Modifier 15 :om.example.test (androidT import androidx.compose.ui.text.font.FontStyle 16 om.example.test (test) import androidx.compose.ui.tooling.preview.Preview 17 import androidx.compose.ui.unit.dp 18 (generated) 19 import com.example.test.models.Produto Scripts import com.example.test.services.jsonProdutos 28 import com.example.test.services.objProdudos 21 22 import com.google.gson.Gson

FPFtech Escola Tecnológica

2 - ListaAPIprodutos

```
@Composable
fun ExibirProduto(produtos: List<Produto>) {
    Column(modifier = Modifier.padding(8.dp)) {
        val gson = remember { 6son() }
        produtos.forEach { produto ->
            Card(
                modifier = Modifier
                    .fillMaxWidth()
                    .padding(vertical = 4.dp),
                elevation = CardDefaults.cardElevation(defaultElevation = 4.dp)
                Column(modifier = Modifier.padding(16.dp)) {
                    Text(
                        text = "Produto: ${produto.nome}",
                        style = MaterialTheme.typography.titleMedium
```



2 - ListaAPIprodutos

```
Text(
   text = "Preço: R$${produto.preco}",
   style = MaterialTheme.typography.bodyLarge
Spacer(modifier = Modifier.height(8.dp))
Text(
   text = "JSON: ${gson.toJson(produto)}",
   style = MaterialTheme.typography.bodySmall,
   fontStyle = FontStyle.Italic
```

2- ListaAPIprodutos

```
@Composable

y fun listaAPIprodutosPreview(){
     listaAPIprodutos(jsonProdutos);
 OPreview
 @Composable
 fun ExibirProdutoPreview(){
     ExibirProduto(objProdudos)
```





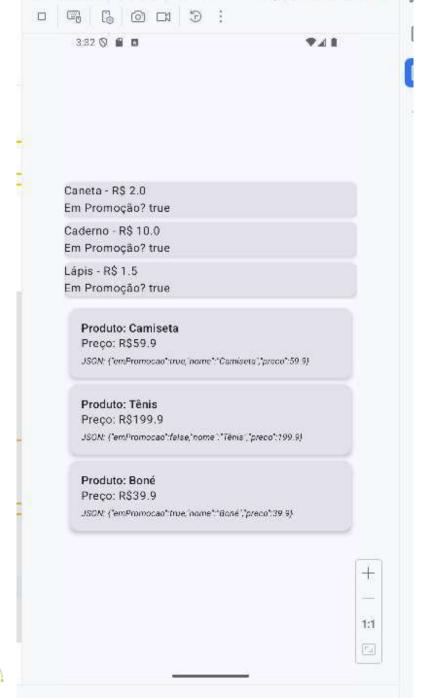


```
@Composable
fun Greeting(name: String, modifier: Modifier = Modifier) {
    Column (
        modifier = modifier.then(Modifier.fillMaxSize()),
        verticalArrangement = Arrangement.Center,
        horizontalAlignment = Alignment.CenterHorizontally
        listaAPIprodutosPreview();
        ExibirProdutoPreview();
```



4 - Resultado

000



Q Out Of Date S











Obrigado!











in @fpftech.educacional