



Unidade 2

Seção 2



Desenvolvimento para Dispositivos Móveis



Webaula 1

Armazenamento *key-value*

Nesta webaula vamos apresentar conceitos e técnicas para armazenamentos de dados no Android com estrutura *key-value*

Segundo Developer (2018), o Android oferece uma lista de opções para que os dados dos aplicativos fiquem salvos de maneira persistente, ou seja, é possível salvar os dados de um aplicativo Android e recuperá-los mesmo após o aplicativo ser encerrado.

Preferências Compartilhadas é a primeira opção disponível para armazenamento de dados no Android. Através das Preferências Compartilhadas é possível armazenar dados primitivos em uma estrutura de chave-valor, também conhecida como *key-value*.





Veja o exemplo a seguir:

```
{  
    "perfil" : "administrador",  
    "valor" : 75.4,  
    "quantidade" : 28,  
    "privilegio" : true  
}
```

Há quatro registros com chaves do tipo *String* e valores primitivos associados a cada chave.

Neste exemplo, as chaves são “perfil”, “valor”, “quantidade” e “privilegio”. Elas obrigatoriamente devem ser do tipo *String*.

Clique aqui para saber mais



Os valores associados a essas chaves podem ser do tipo *String*, *float*, *int* *boolean* ou *Set<String>*. Devemos utilizar o nome da chave para acessar o valor correspondente, por exemplo, a chave “perfil” possui o valor “administrador”.

Ao armazenar dados com as Preferências Compartilhadas no Android, deve-se utilizar a *Interface SharedPreferences*.

Segundo Developer (2018), um objeto *SharedPreferences* organiza os dados da seguinte maneira:

I) *SharedPreferences* armazena os dados em um arquivo com estrutura *chave*-*valor*,

II) Para utilizar o *SharedPreferences* deve-se fornecer um nome para o arquivo que terá os dados armazenados.

III) É possível possuir mais de um arquivo com dados armazenados pelo *SharedPreferences*, basta fornecer nomes diferentes para os arquivos.

IV) Um objeto *SharedPreferences* fornece métodos para escrever e ler os valores armazenados.

Para criar ou acessar um arquivo de *SharedPreferences* deve-se chamar um método. Explore a galeria e conheça os métodos.

getSharedPreferences(String nomeDoArquivo, int modoDeAcesso)

- a) Recebe dois parâmetros: o primeiro parâmetro refere-se ao nome do arquivo que será utilizado para gravar os dados; o segundo parâmetro refere-se ao modo de operação para gravar e/ou acessar o arquivo.
- b) Uma característica importante deste método é que outra *Activity* do aplicativo terá acesso aos dados gravados neste arquivo.

O parâmetro Modo de Operação para gravar e/ou acessar o arquivo define como o *SharedPreferences* irá tratar o arquivo. Clique nos botões para conhecer dois modos.

MODE_PRIVATE

É o modo mais utilizado. Ao utilizar este modo, o arquivo de preferências criado estará disponível apenas para o aplicativo que o criou.

MODE_APPEND

Verifica se o arquivo já existe e, se existir, adiciona novos dados no final do arquivo.

Em Java, a classe *Object* é a classe-pai para todas as demais classes que trabalhamos. A classe *Activity* herda a classe *Context*, que herda *Object*. *Context* possui informações sobre o ambiente do aplicativo. Também são disponibilizadas constantes na classe *Context* para que os aplicativos possam utilizá-las, entre elas, estão as constantes *MODE_PRIVATE* e *MODE_APPEND*.

Assim, uma *Activity* é filha da classe *Context* que possui as constantes *MODE_PRIVATE* e *MODE_APPEND*

Hierarquia da classe Activity

`java.lang.Object`

↳ `android.content.Context`

↳ `android.content.ContextWrapper`

↳ `android.view.ContextThemeWrapper`

↳ `android.app.Activity`

Fonte: <https://bit.ly/2KEIOvq>. Acesso em: 28 abr. 2018.

No exemplo, ao chamar os métodos citados, será retornado

```
18. private void gravarDados() {  
19.     SharedPreferences sharedpreferences =  
20.         getSharedPreferences("MeusDados", Context.MODE_PRIVATE);
```

Explicação do exemplo

Linha 18: foi criado um método na *Activity* para gravar os dados em um arquivo de preferência.

Linha 19: foi criado um objeto *SharedPreferences* com o nome de arquivo definido como “MeusDados” e no modo privado.

Explicação do exemplo.

Imagen texto acima

Para que possamos salvar dados, devemos chamar o método `edit()` que nos retornará um objeto `Editor`. Através do objeto `editor` criado é possível salvar os dados.

```
private void gravarDados() {  
    SharedPreferences sharedPreferences = getSharedPreferences( name: "MeusDados", Context.MODE_PRIVATE );  
    SharedPreferences.Editor editor = sharedPreferences.edit();  
}
```

A screenshot of the Android Studio code editor. The line of code `SharedPreferences.Editor editor = sharedPreferences.edit();` is highlighted with a red circle. A red arrow points from this circle to a tooltip window titled "Editor" which displays the method signature `edit()`.

Fonte: Captura de tela do Android Studio, elaborada pelo autor.

Após os dados serem direcionados para o objeto editor, deve-se então realmente realizar a gravação dos valores no arquivo através do método *apply()*. Veja a continuidade do exemplo apresentado anteriormente, observe o código para salvar os dados em um arquivo de preferência.

Clique na imagem para visualizar melhor.

```
15. private void gravarDados(){
16.     // Recebemos uma instância do arquivo de preferências
17.     SharedPreferences sharedPreferences =
18.         getSharedPreferences("MeusDados", MODE_PRIVATE);
19.
20.     // Recebemos uma instância para iniciarmos no modo de
21.     // edição
22.     SharedPreferences.Editor editor = sharedPreferences.edit();
23.
24.     // Adicionamos os dados que desejamos salvar
25.     editor.putString("perfil", "administrador");
26.     editor.putFloat("valor", 75.4f);
27.     editor.putInt("quantidade", 28);
28.     editor.putBoolean("privilegio", true);
29.
30.     // Realizamos a gravação do arquivo
31.     editor.apply();
```

Fonte: elaborada pelo autor.

Um objeto *SharedPreferences* fornece métodos que permitem recuperar os dados. Veja no exemplo o código para recuperar os valores de um arquivo de preferência.

Explicação do exemplo.

```
32. private void localizarD  
33.     // Recebemos uma in  
34.     SharedPreferences s  
getSharedPreferences("M  
35.  
36.         // Recuperamos os v  
String perfil = sha  
"usuário comum");  
38.         float valor = share  
39.         int quantidade = sh  
40.         boolean privilegio  
sharedPreferences.getBo  
41.  
42.             // Esta chave não e  
parâmetro fornecido ao chamar o método  
43.             float multa = sharedPreferences.getFloat("multa", 100);  
44. }
```

Explicação do exemplo

Linha 32: foi utilizado um método para recuperar os valores. Linhas 37 até 40: recuperado os valores que se deseja trabalhar e armazenados em variáveis locais. Linha 43: foi tentado recuperar um valor com uma chave inexistente. O valor retornado é 100 (cem), pois foi fornecido como valor padrão ao chamar o método *getFloat()*.

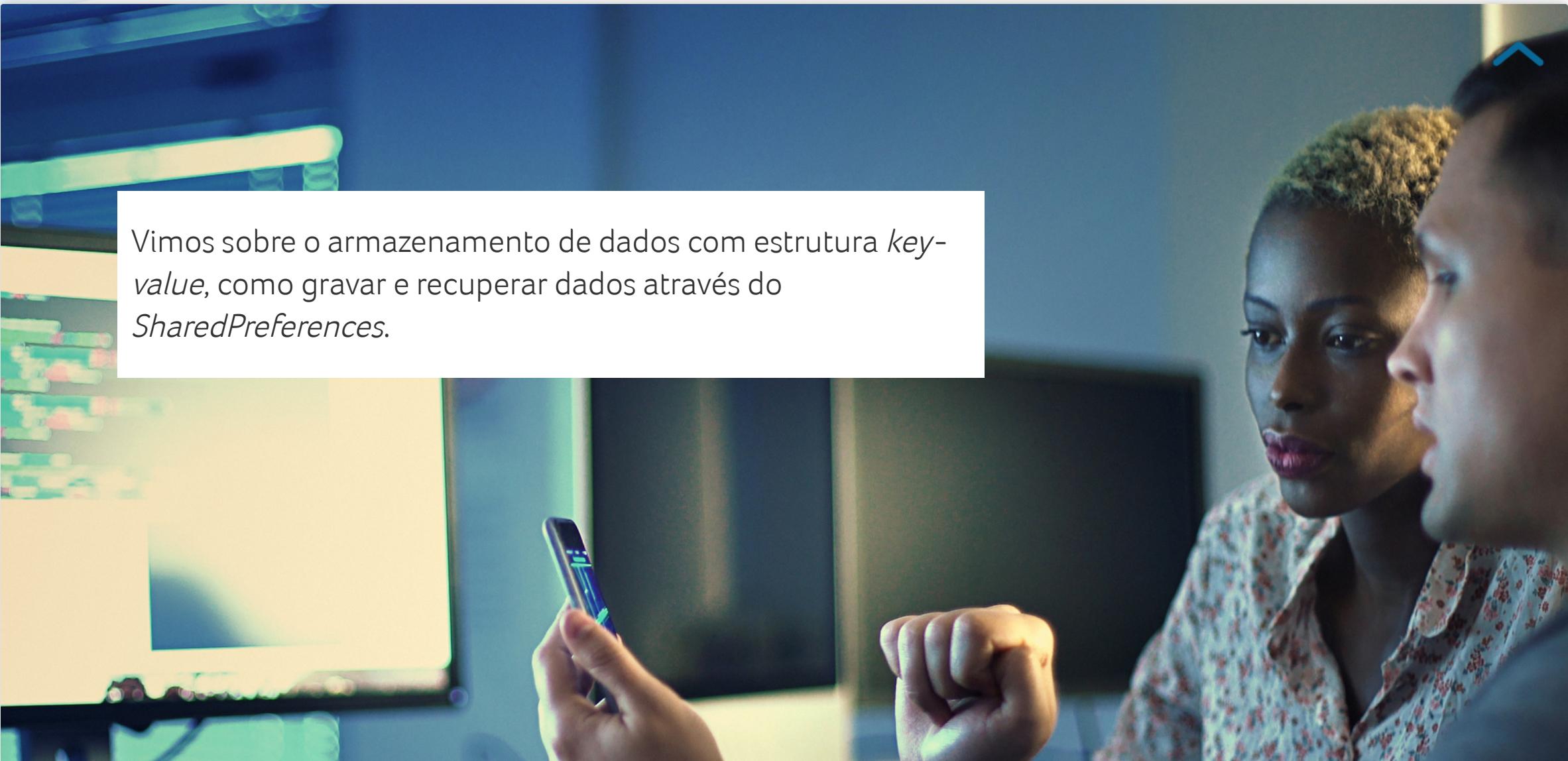
Fonte: elaborada pelo autor.



Link

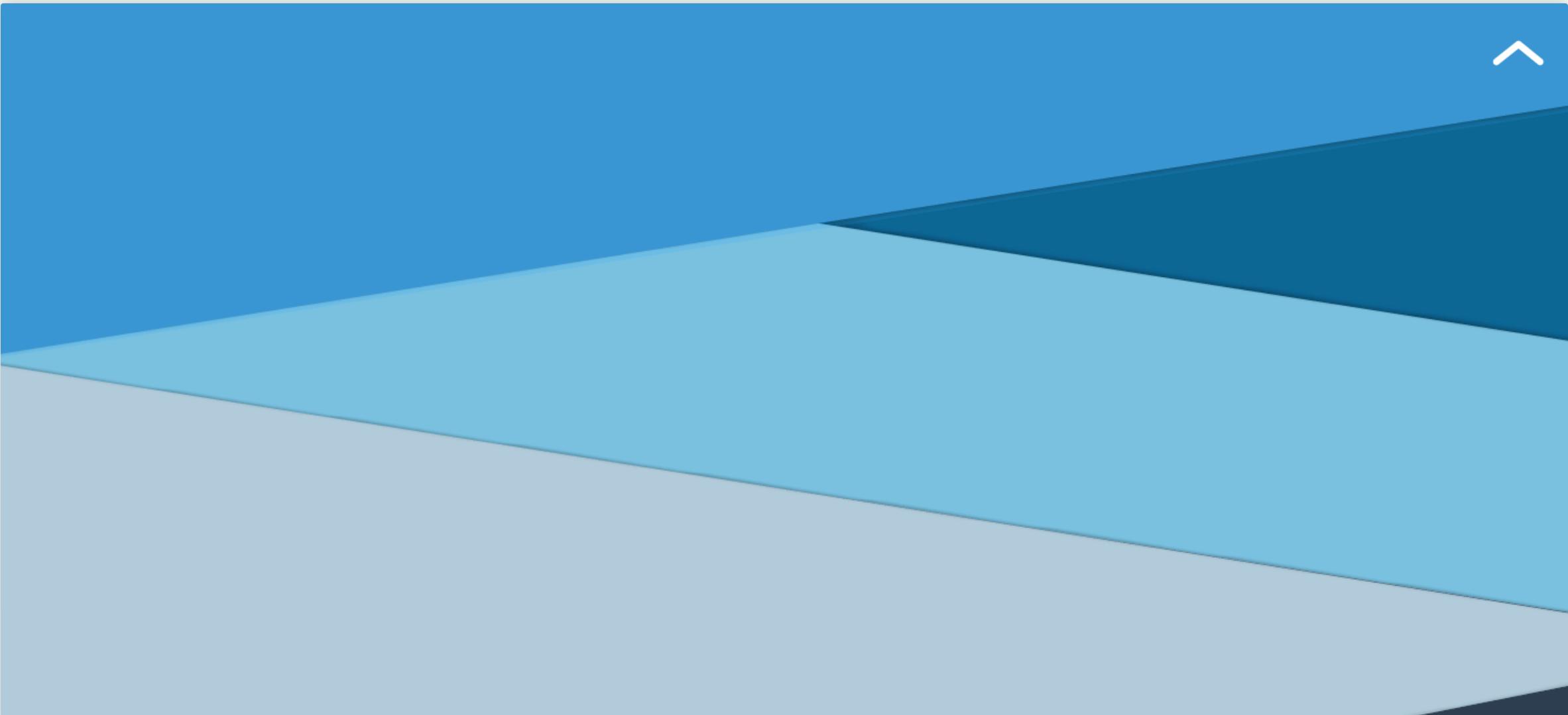
Você poderá encontrar mais informações sobre armazenamento de dados no site oficial de desenvolvimento do Android.

Disponível em:
<https://developer.android.com/guide/topics/data/data-storage#pref>. Acesso em: 29 abr. 2018.



Vimos sobre o armazenamento de dados com estrutura *key-value*, como gravar e recuperar dados através do *SharedPreferences*.





Bons estudos!

