Programação Para Dispositivos Móveis

Aula 9

Paulo Tomé

Escola Superior de Tecnologia e Gestão de Viseu

ptome@estgv.ipv.pt

Ano letivo 2024-2025

Sumário

Método InitState

- O *InitState* é um método da classe *State*. Este método é utilizado em *widgets Stateful* para realizar operações antes de ser executado o método *build*.

- Estes método pode ser utilizado para:

  - inicializar valores de variáveis;
  - Carregar valores de bases de dados;
  - Ir buscar valores a APIs;
  - Enviar valores para APIs.

Método InitState

- Exemplo de método:

```
@override void initState() {
 super.initState();

 bd.apagprods(); // apaga valores da tabela
 Servidor se = Servidor();
 se.url="https://dummyjson.com/products";
 se.carregaprodutos(bd); //insere novo conjunto de valores
}
```

- A abordagem anterior é a mais fácil de implementar. Apagam-se os valores todos e carregam-se de novo todos os valores.

- Uma abordagem diferente é:
  - não se apagam os valores da tabela local;
  - só se descarregam os valores posteriores à data da última execução/atualização.

Método InitState

- Este método não pode ser de executado de forma assíncrona.

Classe WidgetsFlutterBinding

- Esta classe permite fazer a ligação ao ”Flutter Engine”.
- Esta classe assegura métodos que, por exemplo, asseguram que determinados elementos estão inicializados.
- Exemplo de utilização do método *ensureInitialized*:

```
Future<void> main() async {
 WidgetsFlutterBinding.ensureInitialized();

 final cameras = await availableCameras(); // a)
 final primeiracam = cameras.first; // b)
 runApp(MyApp(cam: primeiracam));
}
```

Classe WidgetsFlutterBinding

- No exemplo anterior há que ter em atenção:
  - Programa principal é assíncrono;
  - Em a) determinam-se as câmaras disponíveis;
  - Em b) seleciona-se a primeira câmara;
  - Invoca-se Myapp com a primeira câmara disponível.

Utilização da câmara

- Em flutter há várias livrarias para interagir com a câmara:
  - *camera*: `https://pub.dev/packages/camera`
  - *camerawesome* : `https://pub.dev/packages/camerawesome`

- Fases para utilizar a câmara:
  - Adicionar as dependências necessárias;
  - Obter lista de câmaras disponíveis;
  - Criar e inicializar *CameraController*;
  - Utilizar *CameraPreview* (pré-visualização da imagem/video);
  - Registar através do *CameraController*;
  - Mostrar dados recolhidos.

Utilização da câmara

- Adicionar as dependências necessárias:

```
dependencies:
  flutter:
    sdk: flutter
  sqflite:
  http:
  camera:
```

- Obter lista de câmaras disponíveis: Diapositivo 6.

Utilização da câmara

- Criar e inicializar *CameraController*: Esta operação deve ser feita na *route* que vai "tirar" a foto.

- O código pode ser:

```
@override
void initState()  {
  super.initState();
  _controller = CameraController(widget.cam, ResolutionPreset.medium, ); // a)
  _initializeControllerFuture = _controller.initialize();
}
```

- Ter em consideração que em a) está-se a aceder à primeira câmara identificada no programa principal.

Utilização da câmara

- Utilizar *CameraPreview* (pré-visualização da imagem/video):

```
@override build(BuildContext context) {
  return Column(
  children: [FutureBuilder<void>(
  future: _initializeControllerFuture,
  builder: (context, snapshot) {
  List<Widget> children;
  if (snapshot.connectionState == ConnectionState.done) {
  children = <Widget>[
  SizedBox(width: MediaQuery.of(context).size.width-8,
  height: MediaQuery.of(context).size.width-8,
    child: CameraPreview(_controller)),
    ElevatedButton(onPressed: ()
    async { tirafoto(); }, child: const Icon(Ionicons.camera_outline)),];
    }
    else if (snapshot.hasError) {
   .......
  }
```

```
Future<void> tirafoto() async
{

    final image = await _controller.takePicture();
    print(image.path);
}
```

Utilização da câmara

- Verificação de permissões:

```
Future<bool> _checkPermissions() async {
    List<Permission> permissions = [];

    var cameraStatus = await Permission.camera.status;
    if (!cameraStatus.isGranted) permissions.add(Permission.camera);

    var storageStatus = await Permission.storage.status;
    if (!storageStatus.isGranted) permissions.add(Permission.storage);

    if (permissions.isEmpty) {
      return true;
    } else {
      try {
        Map<Permission, PermissionStatus> statuses =
            await permissions.request();
        return statuses[Permission.camera] == PermissionStatus.granted &&
            statuses[Permission.storage] == PermissionStatus.granted;
      } on Exception catch (_) {
        return false;
      }
    }
}
```

Utilização da câmara

- Declaração de variáveis:

```
late CameraController _controller;
late Future<void> _initializeControllerFuture;
```

Tarefa

- Desenvolva uma aplicação com três *routes*.
- A route principal contém um botão que direciona para uma em que é possível tirar um foto (segunda route). Na *route* principal tem de ser selecionada a primeira câmara. A informação sobre a câmara tem de ser passada para a segunda route.
- A segunda *route* permite pre-visualizar a fotografia e tem um botão para capturar a fotografia. Após a capturar a imagem, o botão redirecciona para a terceira *route*.
- A terceira *route* mostra a fotografia capturada. A imagem deve ser mostrada através de:

```
Image.file(File(caminhoimagem)) // Sendo caminhoimagem o path para a imagem.
```

Bibliografia

- https://api.flutter.dev/flutter/widgets/WidgetsFlutterBinding-class.html
- https://pub.dev/packages/camera
- https://pub.dev/packages/camerawesome
- https://docs.flutter.dev/cookbook/plugins/picture-using-camera