



Flutter Básico - Aula 3

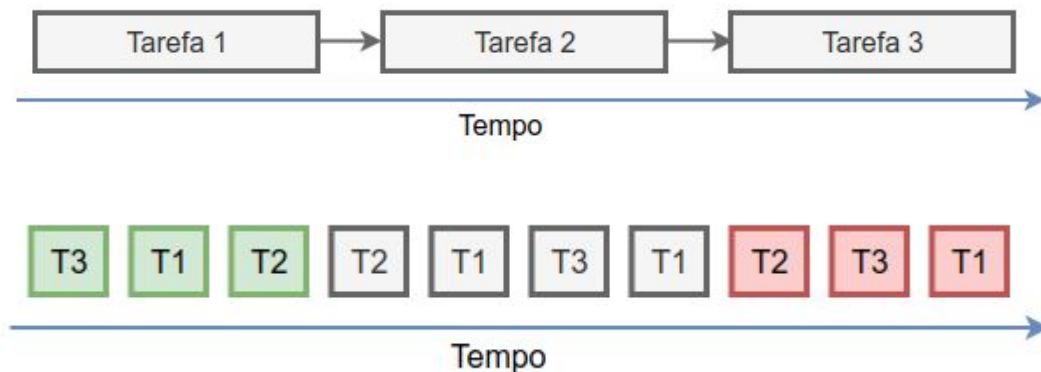
Exercícios 9 a 12



- [iampawan/FlutterExampleApps - GitHub](#); [Ver mais](#)



- Funções assíncronas são executadas fora do fluxo padrão de execução de uma aplicação
- Enquanto aguarda o retorno, libera a aplicação para executar outras funções
- Utilizada para aguardar resultado de fontes externas





Exemplo de uma função assíncrona

```
Future<List> getAllPets() async {  
    final dataList = await DbUtil.getData('pets');  
    _petList = dataList.map((pets) => Pet.fromMap(pets)).toList();  
    return _petList;  
}
```



`async`

O `async` determina que um método será assíncrono, ou seja, não irá retornar algo imediatamente, então o aplicativo pode continuar a execução de outras tarefas enquanto o processamento não é finalizado.

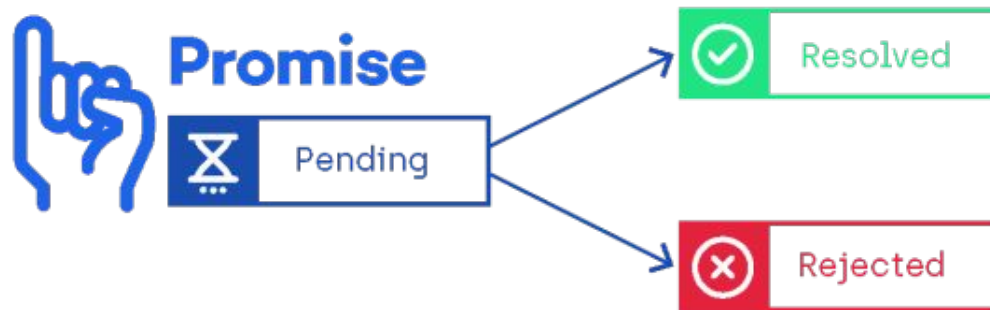
`await`

O `await` serve para determinar que o aplicativo deve esperar uma resposta de uma função antes de continuar a execução. Isso é muito importante pois há casos em que uma função depende do retorno de outra.



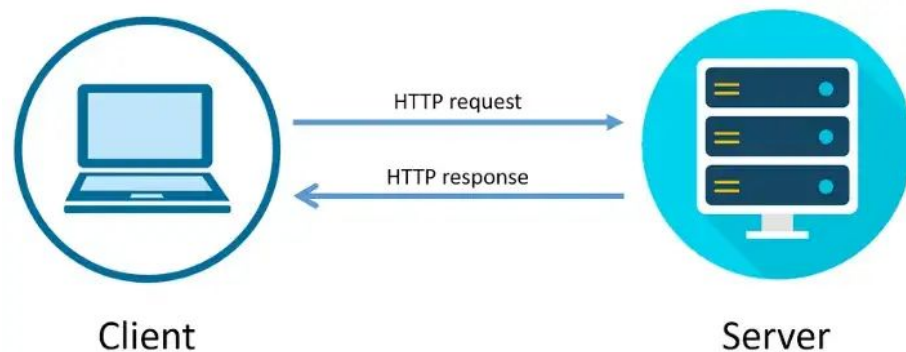
Future

O Future determina que uma função irá retornar algo no “futuro”, ou seja, é uma função que levará um tempo até ser finalizada.





- HTTP: Protocolo de comunicação
- Requisição: Solicitação realizada para o servidor através do navegador, app, etc
- Enviada para o servidor, o servidor processa a requisição e devolve um resultado





- Primeiro, vamos importar o pacote em nosso projeto, através do arquivo pubspec.yaml
- Depois, vamos ao código!

```
import 'package:http/http.dart' as http;

Future<List> Internal_GetWineList() async {
  Uri vrURL = Uri.https("api.sampleapis.com", "/wines/reds");

  http.Response result = await http.get(vrURL);

  return json.decode(result.body);
}
```




- Widget utilizado para construir um layout a partir de uma função assíncrona

```
FutureBuilder(  
  future: this.Internal_GetWineList(),  
  builder: (context, snapshot) {  
    switch (snapshot.connectionState) {  
      case ConnectionState.none:  
      case ConnectionState.waiting:  
        return Container(  
          child: CircularProgressIndicator(),  
        );  
  
      default:  
        return this.Internal_ShowList(snapshot.data!);  
    }  
  }
```

- Vamos criar um app que busque e exiba uma lista de tipos de cerveja
- A lista está na API <https://api.sampleapis.com/beers/ale>
- Você deverá exibir o nome e o preço da cerveja

