Escola de Mobile

Dart: Primeiros Passos

Material de Apoio

alura

O que você vai encontrar aqui:

Aula 1

1.1 Introdução

Porque Usar Dart?

1.2 Onde Programar

Aprenda mais sobre IDE's e SDK's.

1.3 Funções Iniciais

Primeiro contato com o Dart e as funções

void *main(){}* e *print()*.

1.1 Introdução

Porque usar Dart?

- História do Dart

- Vantagens da Linguagem

- Mercado

1.2 Onde Programar

1.2 Onde Programar

- IDE (Integrated Development Environment)
  - IntelliJ
  - VSCode
  - DartPad

- SDK (Software Development Kit)

1.3 Funções Iniciais

1.3 Funções Iniciais

- **`void`** *`main(){}`*
  - ***void*** *→ Vazio, não retorna nada;*
  - *() → recebe argumentos de fora;*
  - *{} → Nosso código fica aqui dentro.*

- *`print();`*
  - *função que imprime informações no console;*
  - *() → recebe a informação;*
  - *; → necessário para terminar a frase.*

Aula 2

2.1 Numbers

Quantificar objetos e criar valores numéricos.

2.2 Boolean

Valor simples de **verdadeiro** ou **falso**.

2.3 String

Criar e escrever frases inteiras.

2.1 Numbers

2.1 Numbers

- **Necessários para quantificar;**

- Tipo → `int;`
  - Aceita números inteiros;
  - 2^**53** números.

- Tipo → `double;`
  - Aceita números fracionados;
  - Até 53 casas decimais.

2.2 Boolean

2.2 Boolean

- Resposta simples ideal para condições:
  - **`true`** → Verdadeiro / Sim;
  - **`false`** → Falso / Não;
  - Comparações resultam em bool;
    - Cinco é menor que 10? (Sim).
      - `(5<10) == true;`

2.3 String

2.3 String

- Necessário estar entre *aspas ‘’*:
  - Podemos usar *aspas duplas “”*.

- Pode receber informações de objetos criados!
  - `‘Eu tenho $idade anos de idade’` .

- É possível **concatenar** (juntar) frases:
  - `‘Eu amo’ + ‘Dart’.`

Aula 3

3.1 Listas Únicas

Ideal para manipular vários objetos iguais juntos.

3.2 Listas Dinâmicas

Listas que aceitam objetos diferentes entre si.

3.3 Var, Const e Final

Definir se o objeto pode mudar ou ficar fixo.

3.1 Listas Únicas

3.1 Listas Únicas

- `List<String> nomes = [];`
  - List → Comando para definir o objeto;
  - <String> → Define o tipo de valor;
  - nomes → identificação da Lista;
  - [] → Lista vazia.

- `List<String> nomes = [‘Kako’,’Ricarth’,’Natália’];`

3.2 Listas Dinâmicas

3.2 Listas Dinâmicas

- `List<dynamic> personas = [];`
  - List → Comando para definir o objeto
  - <dynamic> → tipo dinâmico (variável)
  - personas → identificação da Lista
  - [] → Lista vazia

- `List<dynamic> personas = [‘Kako’,27,true];`

3.3 Var, Const e Final

3.3 Var, Const e Final

- **Var → Tipo variável:**
  - o Dart define o tipo;
  - usar quando não souber o tipo do objeto.
- **Const → Define como constante:**
  - o Dart bloqueia o objeto de qualquer alteração;
  - melhora a performance do código.
- **Final → Variáveis não iniciadas e constantes:**
  - não sabe o valor inicial do objeto;
  - uma vez que souber, impedir mudanças.

Aula 4

4.1 If / Else

Condição de escolha baseado em booleanos.

4.2 For

Criação de loops de repetição simples.

4.3 While / Do

Loops de repetição sem limites definidos.

4.1 If / Else

4.1 If / Else

- `if(){} else {}`
  - **if** → comando que inicia a condição;
  - **()** → comparação com resultado booleano;
  - **{}** → ação tomada caso valor verdadeiro;
  - **else** → comando para indicar ação caso falso;
  - **{}** → ação tomada caso valor seja falso.

- `if(idade>=18){“maior de idade”}else{“menor de idade”}`

Para melhor visualizar:

```
if(idade>=18){
    print(“maior de idade”);
}else{
    print(“menor de idade”);
}
```

4.2 For

4.2 For

- `for(int i=0; i<10;i++){}`
  - **for** → comando para iniciar o loop
  - **(int i=0; i<10;i++)** → argumento de condição
    - **int i=0** → iniciar objeto de comparação
    - **i<10** → comparação para condição
    - **i++** → ação pós-loop

- `for(int a=0;a>=6;a=a+2){print(“ Repetição $a”);}`

Para melhor visualizar:

```
for(int a=0;a>=6;a=a+2){
    print(“Repetição $a”);
}
```

4.3 While / Do

4.3 While / Do

- `while(){}`
  - **while** → comando para iniciar a repetição;
  - () → condição;
  - {} → ação tomada caso verdadeiro.

- `do{}while()`
  - **do** → comando para iniciar ação;
  - {} → ação;
  - **while**() → comando para verificar repetição.

Aula 5

5.1 Comentários

Entender e aplicar comentários não lidos pelo Dart.

5.2 Documentação

Explorando a Documentação.

5.3 Aplicação

Podemos aplicar o que aprendemos?

5.1 Comentários

5.1 Comentários

- Importantes para **guiar quem vai ler** seu código;

- // ou /**/
  - Comentários simples e cinzas, ideais para não poluir o código;

- /// ou /****/
  - Comentários de Documentação, verdes e podem poluir o código.

5.2 Documentação

5.2 Documentação

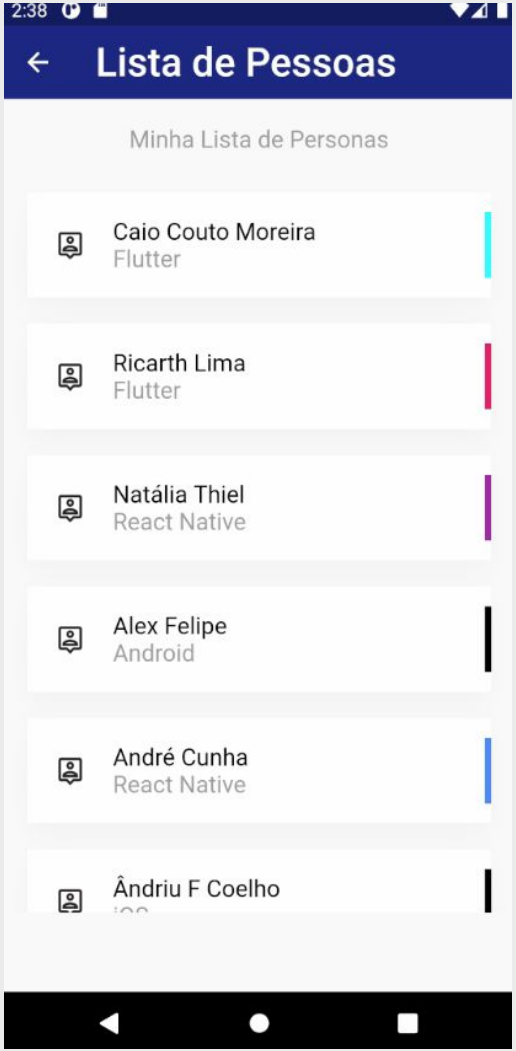
- Aprender a navegar pela documentação do Dart;

- Ideal para tirar dúvidas iniciais.

5.3 Aplicação

5.3 Aplicação na Vida Real

- Aprendemos a ver variáveis em um aplicativo!
- Vimos como condições são usadas em Apps;
- **Não aprendemos a fazer o App neste curso ainda!**



alura

Obrigado!

“Profissionais da TI não vão para o céu. Eles são armazenados na Nuvem.”

KAKO
Instrutor