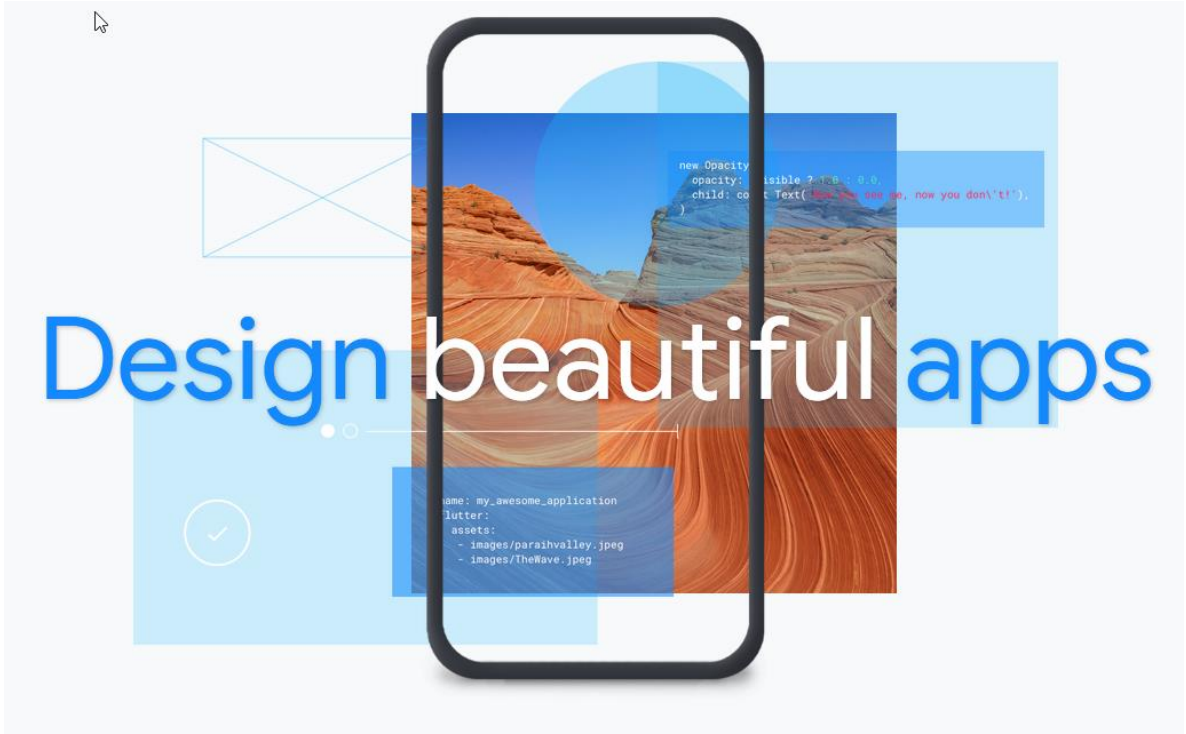
Transformando o futuro das pessoas
e as pessoas para o futuro.

#Senacfaz75

Senac

Desenvolvimento Mobile: Flutter



O que é o Flutter?

Flutter é um kit de desenvolvimento de interface de usuário (UI toolkit), de código aberto, criado pelo Google, que possibilita a criação de aplicativos compilados nativamente. Atualmente pode compilar para Android, iOS, Windows, Mac, Linux, Google *Fuchsia e Web.

O que é o Dart?

Os aplicativos Flutter são escritos na linguagem de programação Dart e fazem uso de muitos dos recursos mais avançados da linguagem.

No Windows, macOS e Linux, por meio do projeto semi-oficial Flutter Desktop Embedding, o Flutter é executado na máquina virtual Dart, que possui um mecanismo de compilação que ocorre em tempo de execução. Ao escrever e depurar um aplicativo, o Flutter usa a compilação JIT, permitindo o "hot reload", com a qual as modificações nos arquivos de origem podem ser injetadas em um aplicativo em execução. O Flutter estende isso com suporte para hot reload de widgets stateful, onde na maioria dos casos as alterações no código-fonte podem ser refletidas imediatamente no aplicativo em execução, sem a necessidade de uma reinicialização ou perda do Estado.

As versões de lançamento dos aplicativos Flutter são compiladas com a compilação antecipada (AOT) no Android e no iOS, possibilitando o alto desempenho do Flutter em dispositivos móveis.

*Fuchsia é um sistema operacional atualmente sendo desenvolvido pelo Google. Ao contrário de sistemas operacionais anteriores desenvolvidos pelo Google, como o Chrome OS e o Android, que são baseados no kernel Linux, Fuchsia é baseado em um novo microkernel chamado Zircon (o nome anterior era Magenta), derivado do Little Kernel, que foi destinado para sistemas embarcados e é principalmente escrito em C. Fuchsia foi projetado para ser executado em uma infinidade de dispositivos, incluindo telefones celulares e computadores pessoais.

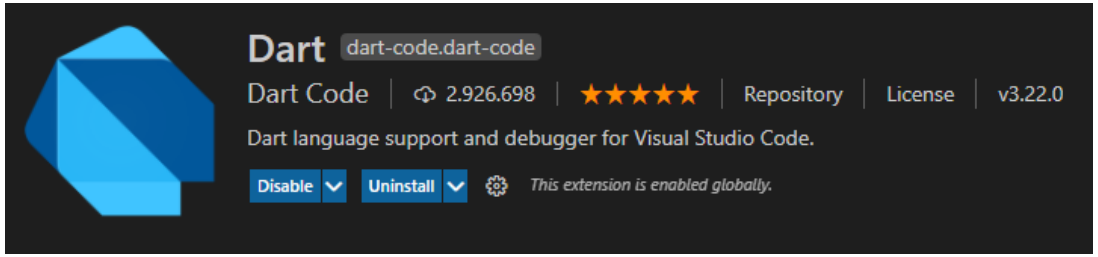
Instalações:

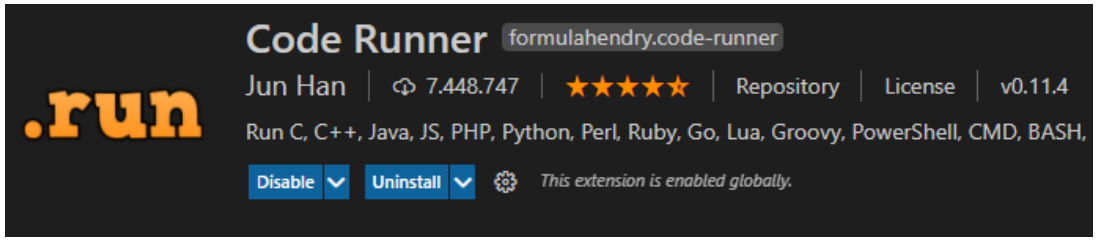
Java SE
https://www.oracle.com/java/technologies/javase/javase-jdk8-downloads.html

Flutter
https://flutter.dev/docs/get-started/install/windows

OBS: Colocar a pasta bin do Flutter no Path do Windows

Extensões VS Code





Null Safety

A partir do Dart 2.12, temos a possibilidade de usar o Null Safety para facilitar nossas aplicações, dificultando erros de valores nulos. Aprenderemos a identificar erros de valores nulos, aplicar no nosso projeto do ByteBank, atualizar o Dart para a nova versão e migrar um projeto inteiro! Assim podemos criar códigos complexos com mais facilidade!

Erros de valor Null

Existem infinitos erros que podem acontecer por conta de um valor nulo:

- Você pode estar esperando um dado do seu back-end, mas ele não existe ainda...
- Você pode criar uma lista que muda de tamanho de acordo com a quantidade de produtos...

Fonte: Kako(Caio couto Moreira). Alura. Disponível em: https://www.alura.com.br/artigos/flutter-null-safety

Aula 1 – Hello dart e tipos de variáveis!

hellodart.dart > ...

```dart
//É preciso criar uma classe main para rodar o código Dart
Run | Debug
main() {
  //print é o comando de saída em Dart
  print('Hello dart!');
  print('Como vai?');

  //Tipos de variáveis em Dart
  //Dart faz inferência de tipo (Genérico)
  var nome = 'John Doe';
  //nome é do tipo String
  //Se eu tentar colocar outro valo nessa variável que não seja String
  //vai dar erro
  //var nome = 10; //Tipo já foi definido como String (ERRO)

  /**
   * Podemos também definir uma variável
   * já com o seu tipo
   */

  //Variável do tipo String
  String novoNome = 'Jane Doe'; //Estou tipando minha variável
  String CPF = '999.999.999.99-99';

  //Definindo um tipo numérico inteiro
  int ano = 1970;

  //Definindo um número decimal
  double altura = 1.77;

  //Definindo um tipo booleano (Verdadeiro ou falso)
  bool vf = true;
  bool fv = false;

  //Saída Interpolada, ao contrário do javascript
  //Não é reciso as {} e `` para interpolar valores
  print('Seu nome: $novoNome'); //Lembra PHP
  print('Nascimento: $ano');
  print('Sua altura: $altura');
  print('Resultado VF: $vf');
  print('Resultado FV: $fv');
}
```

```
Hello dart!
Como vai?
Seu nome: Jane Doe
Nascimento: 1970
Sua altura: 1.77
Resultado VF: true
Resultado FV: false

[Done] exited with code=0 in 1.359 seconds
```

Aula 1 – Estruturas de Dados Arrays (arrays.dart)

```dart
Run | Debug
void main() {
  //Arays são estruturas que armazenam mais de um valor
  //a uma variável

  //Criando um array
  var compras = ['Macarrão', 'Feijão', 'Pão', 'Manteiga'];

  //Exibindo o array
  print('Lista de compras: $compras');

  //Exibindo os itens pelo índice
  print('Nome do primeiro produto: ${compras[0]}');
  print('Nome do segundo produto: ${compras[1]}');
  print('Nome do terceiro produto: ${compras[3]}');

  //Item out of range
  //print('Nome do primeiro produto: ${compras[5]}');

  //Acessando e alterando um valor do array
  compras[0] = 'Arroz';
  //Exibindo os itens pelo índice
  print('Nome do primeiro produto: ${compras[0]}');

  //Criando array numérico
  var pares = [0, 2, 4, 6];

  //Exibindo o array
  print('Lista de números pares $pares');
  print('------------------------------------------------------');

  //Métodos utilizados em arrays
  /**
   * first() : Retorna o primeiro elemento do Array
   * last() : Retorna o último elemento do Array
   * isEmpty() : Retorna true se a lista está vazia, caso contrário, false.
   * length() : Retorna o tamanho do Array
   */

  var listaNomes = ['José Maria', 'Pedro da Silva', 'Cristina Pereira'];

  print('Primeiro nome: ${listaNomes.first}');
  print('Último nome: ${listaNomes.last}');
  print(
      'O Array está vazio? ${listaNomes.isEmpty}');
  print('Tamanho do Array: ${listaNomes.length}');
}
```

```
Lista de compras: [Macarrão, Feijão, Pão, Manteiga]
Nome do primeiro produto: Macarrão
Nome do segundo produto: Feijão
Nome do terceiro produto: Manteiga
Nome do primeiro produto: Arroz
Lista de números pares [0, 2, 4, 6]
------------------------------------------------------
Primeiro nome: José Maria
Último nome: Cristina Pereira
O Array está vazio? false
Tamanho do Array: 3
Exited
```

Aula 1 – Operdaores Aritméticos (oparitimeticos.dart)

```dart
Run | Debug
void main() {
  //Operadores aritméticos
  // + - * / %

  int a = 20;
  int b = 5;

  //Operador de Soma
  var soma = a + b;

  //Operador de Subtração
  var subt = a - b;

  //Operador de Multiplicação/Produto
  var produto = a * b;

  //Operador de divisão
  var divisao = a / b;

  //operador resto da divisão
  var restoDiv = a % b;

  //Saída
  print('Soma de $a + $b = $soma');
  print('Subtração de $a - $b = $subt');
  print('Produto de $a * $b = $produto');
  print('Divisão de $a / $b = $divisao');
  print('Resto da Divisão de $a % $b = $restoDiv');

  //Operdador resumido
  int num = 100;
  num += 20;
  num -= 10; //Pode ser * /

  //Saída
  print('Número: $num');

  //Incremento e decremento
  num++;
  num--;

  //Saída
  print('Número: $num');
}
```

```
Soma de 20 + 5 = 25
Subtração de 20 - 5 = 15
Produto de 20 * 5 = 100
Divisão de 20 / 5 = 4.0
Resto da Divisão de 20 % 5 = 0
Número: 110
Número: 110

[Done] exited with code=0 in 1.345 seconds
```

Aula 1 – Operadores Relacionais (oprelaconais.dart)

```dart
Run | Debug
void main() {
  //Operdaores Relacionais
  /**
   * ==, != (Igual e diferente)
   * >, < (Maior e menor)
   * >=, <= (Maio igual e Menor igual)
   */

  //Declarando variáveis
  int a = 20;
  int b = 5;

  //Verificando as variáveis
  print('$a = $b? Resultado: ${a == b}');
  print('$a ≠ $b? Resultado: ${a != b}');
  print('$a > $b? Resultado: ${a > b}');
  print('$a < $b? Resultado: ${a < b}');
  print('$a ≥ $b? Resultado: ${a >= b}');
  print('$a ≤ $b? Resultado: ${a <= b}');

  //Podemos atribuir esses resultados a uma variável
  bool igual = a == b;
  print('Verificação de igualdade: $igual');
}

```

```
20 = 5? Resultado: false
20 ≠ 5? Resultado: true
20 > 5? Resultado: true
20 < 5? Resultado: false
20 ≥ 5? Resultado: true
20 ≤ 5? Resultado: false
Verificação de igualdade: false

[Done] exited with code=0 in 1.508 seconds
```

Aula 1 – Operadores Lógicos (oplogicos.dart)

```dart
Run | Debug
void main() {
  //Operadores Lógicos
  /**
   * && (E) --> V && V = V
   * || (OU) --> F || F = F
   * ! Negação
   */

  //Declarando as variáveis
  int a = 20;
  int b = 5;
  int c = 7;

  //Verificando Verdadeiro
  bool proposicao1 = a > b;
  bool proposicao2 = b < c;

  //Saída
  //        V         V
  print('$a > $b && $b < $c - Resposta: ${proposicao1 && proposicao2}');
  //        V         F
  print('$a > $b || $b > $c - Resposta: ${proposicao1 || proposicao2}');

  //Verificando o Falso
  bool proposicao3 = a < b;
  bool proposicao4 = b > c;

  //Saída
  print('$a < $b && $b > $c - Resposta: ${proposicao3 && proposicao3}');
  print('$a > $b || $b > $c - Resposta: ${proposicao4 || proposicao4}');

  //Negar um valor
  bool v = true;
  bool f = false;

  //Saída
  print('Negando o V: ${!v}');
  print('Negando o F: ${!f}');
}
```

```
20 > 5 && 5 < 7 - Resposta: true
20 > 5 || 5 > 7 - Resposta: true
20 < 5 && 5 > 7 - Resposta: false
20 > 5 || 5 > 7 - Resposta: false
Negando o V: false
Negando o F: true

[Done] exited with code=0 in 1.404 seconds
```