**Apostila de Dart: Laços de Repetição**

Bem-vindo à apostila sobre laços de repetição (ou *loops*) em Dart. Este guia irá ensiná-lo a executar um mesmo bloco de código várias vezes, uma das tarefas mais fundamentais e poderosas na programação.

**1. O que são Laços de Repetição?**

Imagine que você precisa imprimir os números de 1 a 100 no console. Escrever print(1);, print(2);, print(3);... cem vezes seria extremamente ineficiente e repetitivo.

Laços de repetição resolvem esse problema. Eles permitem que você defina um bloco de código e uma condição que determinará quantas vezes esse bloco será executado. Isso segue o princípio "Não se Repita" (Don't Repeat Yourself - DRY), um dos pilares da boa programação.

Em Dart, temos principalmente quatro tipos de laços:

* for
* while
* do-while
* for-in (para coleções)

Vamos explorar cada um deles.

**2. O Laço for**

O laço for é ideal para quando você **sabe o número de vezes** que deseja repetir uma ação. Sua estrutura é dividida em três partes, separadas por ponto e vírgula.

**Sintaxe:**

for (inicialização; condição; incremento) {

  // Bloco de código a ser repetido

}

* **Inicialização:** Executada apenas uma vez, no início. Geralmente, é onde se declara uma variável de controle (ex: int i = 0).
* **Condição:** Verificada *antes* de cada repetição. Se for true, o bloco de código é executado. Se for false, o laço termina.
* **Incremento:** Executado ao *final* de cada repetição. É usado para atualizar a variável de controle (ex: i++).

**Exemplo Prático: Contando até 5**

void main() {

  for (int i = 1; i <= 5; i++) {

    print("O número é: $i");

  }

}

**Como funciona:**

1. int i = 1; (inicialização): a variável i é criada com o valor 1.
2. i <= 5; (condição): 1 é menor ou igual a 5? Sim (true).
3. O bloco de código print("O número é: $i"); é executado.
4. i++ (incremento): i agora vale 2.
5. O processo se repete até que i seja 6, quando a condição 6 <= 5 se torna false e o laço para.

**Exercícios for**

1. **Tabuada:** Escreva um laço for que imprima a tabuada do 7 (de 7x1 até 7x10).
2. **Contagem Regressiva:** Faça um laço for que conte de 10 até 1.

**3. O Laço while**

O laço while (enquanto) é usado quando você quer repetir um bloco de código **enquanto uma condição for verdadeira**, mas não sabe exatamente quantas vezes isso vai acontecer.

A condição é verificada **antes** de cada repetição. Se a condição for falsa desde o início, o bloco de código nunca será executado.

**Sintaxe:**

while (condicao) {

  // Bloco de código a ser repetido

}

**Atenção:** É crucial que a variável da condição seja atualizada *dentro* do laço, caso contrário, você pode criar um **loop infinito**!

**Exemplo Prático: Sorteando um número**

import 'dart:math';

void main() {

  int numeroSorteado = 0;

  // O laço continua enquanto o número sorteado for diferente de 5

  while (numeroSorteado != 5) {

    numeroSorteado = Random().nextInt(10); // Sorteia um número de 0 a 9

    print("Número sorteado: $numeroSorteado");

  }

  print("Finalmente! O número 5 foi sorteado.");

}

**Exercícios while**

1. **Soma até 100:** Crie uma variável soma com valor 0. Use um laço while para ir somando 5 a ela a cada repetição, até que soma seja maior ou igual a 100. Imprima o valor de soma a cada passo.

**4. O Laço do-while**

O laço do-while é muito parecido com o while, mas com uma diferença fundamental: a condição é verificada **ao final** da repetição.

Isso garante que o bloco de código será executado **pelo menos uma vez**, mesmo que a condição seja falsa desde o início.

**Sintaxe:**

do {

  // Bloco de código a ser repetido

} while (condicao);

**Exemplo Prático: Menu de Opções** Este laço é ótimo para menus onde você precisa que as opções sejam exibidas pelo menos uma vez.

void main() {

  int opcao = 0;

  do {

    print("--- MENU ---");

    print("1. Cadastrar");

    print("2. Ver Perfil");

    print("3. Sair");

    // Em um app real, aqui leríamos a entrada do usuário.

    // Para simplificar, vamos simular a escolha "Sair" (3).

    opcao = 3;

    print("Você escolheu a opção $opcao");

  } while (opcao != 3);

  print("Programa encerrado.");

}

**Exercícios do-while**

1. **Senha Simples:** Crie uma variável senha e uma tentativa. Use um laço do-while para simular a digitação de uma senha. O laço deve continuar enquanto a tentativa for diferente da senha. Imprima "Tentando..." dentro do laço.

**5. O Laço for-in**

Este é um laço especializado para percorrer os itens de uma coleção (como List, Set ou Map). Ele é muito mais limpo e seguro do que usar um laço for tradicional com índices.

**Sintaxe:**

for (var item in colecao) {

  // Use o 'item' diretamente

}

Exemplo Prático: Percorrendo uma Lista de Frutas

void main() {

  List<String> frutas = ["Maçã", "Banana", "Laranja", "Uva"];

  print("Minha lista de frutas:");

  for (var fruta in frutas) {

    print("- $fruta");

  }

}

Note como o código é mais legível. Você não precisa se preocupar com o tamanho da lista ou com índices (frutas[i]).

**Exercícios for-in**

1. **Soma de Notas:** Crie uma lista de notas (List<double>). Use um laço for-in para somar todas as notas e, ao final, imprima o total.

**6. Controlando Laços: break e continue**

Às vezes, você precisa de mais controle sobre a execução de um laço.

* break: Interrompe e **sai do laço imediatamente**.
* continue: Pula a iteração atual e **vai para a próxima**.

**Exemplo com break: Encontrando um nome**

void main() {

  List<String> nomes = ["Ana", "Bruno", "Carlos", "Daniela"];

  for (var nome in nomes) {

    print("Procurando por Carlos... Nome atual: $nome");

    if (nome == "Carlos") {

      print("Achei!");

      break; // Sai do laço, pois já encontrou o que queria

    }

  }

}

Exemplo com continue: Pulando números pares

void main() {

  for (int i = 1; i <= 10; i++) {

    if (i % 2 == 0) { // Se o número for par...

      continue; // ...pule esta iteração e não execute o print

    }

    print("Número ímpar: $i");

  }

}