

Todos os exercícios devem ser realizados no dartpad.

Todos os exercícios devem ser salvos em um diretório no github de vcs chamado: aulas-dart

Todos os códigos devem estar em inglês.

Não esqueça de realizar todos os testes... se é número negativo, se isNAN... Sempre valide os métodos seja de cálculo ou não com valores bons e valores ruins, ou seja, teste o seu código.

Para todas as classes, deverá ser instanciados no mínimo 3 objetos de cada, enviando valores diferente justamente para abranger a todos os cenários.

1 – Crie três classes, sendo elas: pessoa, aluno e professor. A classe pessoa não pode ser instanciada, ou seja, ela não pode ter uma instância concreta. As classes aluno e professor serão extensões da classe pessoa, e elas poderão ter instâncias.

Classe pessoa

Atributos: Código, Nome, sobrenome

Construtor recebendo os três parâmetros

Método: método para retornar o nome completo todo em maiúsculo, método para retornar o nome completo em minúsculo, método toString

Classe aluno

A classe aluno deve estender da classe pessoa

Atributos: Nota 1, nota 2

Construtor recebendo os dois parâmetros mais os 3 do super

Método: Calcular média, toString

Classe professor

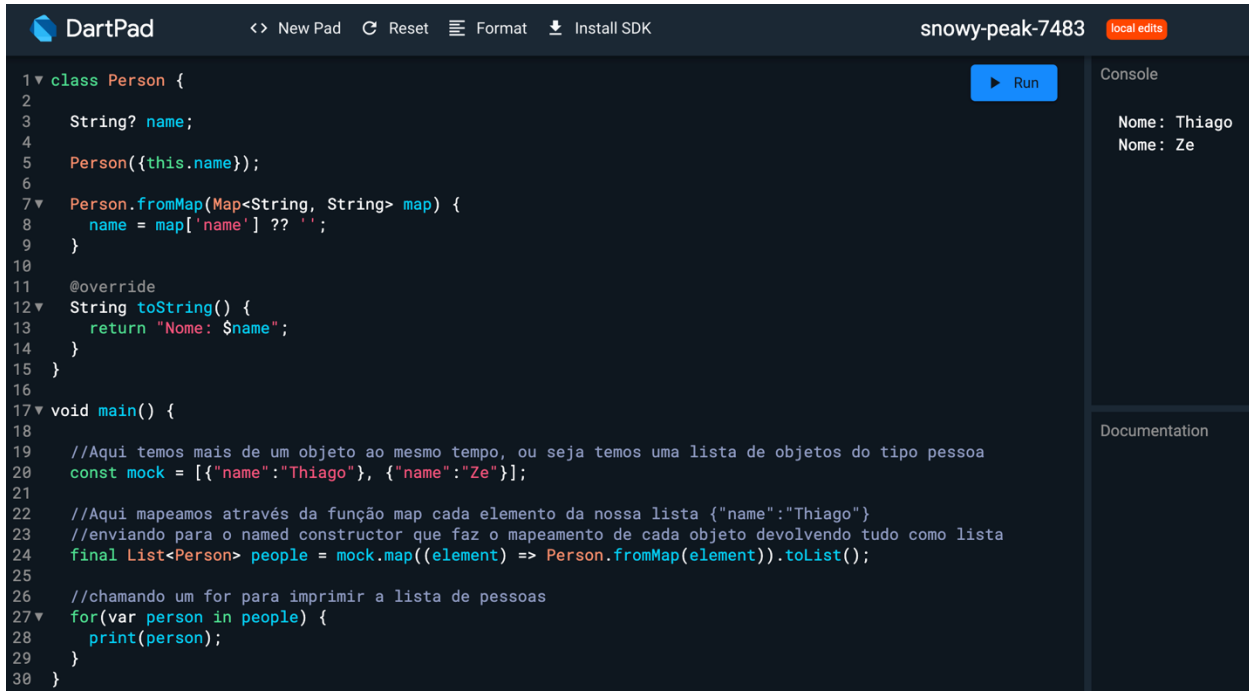
A professor deve estender da classe pessoa

Atributos: valor hora, número de horas trabalhadas

Construtor recebendo os dois parâmetros mais os 3 do super

Método: Calcular salário, toString

2 – Para esse exercício, faça uma classe chamada Person com um named constructor. A ideia é aprender a fazer mocks com “lista de objetos”. Note que na linha 20 temos uma lista (identificada pelos colchetes) de objetos. Já na linha 24 utilizamos um map para enviar objeto por objeto para o named constructor, este faz o mapeamento individual e retorna uma lista de objetos do tipo Person.



```
1 class Person {
2
3   String? name;
4
5   Person({this.name});
6
7   Person.fromMap(Map<String, String> map) {
8     name = map['name'] ?? '';
9   }
10
11   @override
12   String toString() {
13     return "Nome: $name";
14   }
15 }
16
17 void main() {
18
19   //Aqui temos mais de um objeto ao mesmo tempo, ou seja temos uma lista de objetos do tipo pessoa
20   const mock = [{"name": "Thiago"}, {"name": "Ze"}];
21
22   //Aqui mapeamos através da função map cada elemento da nossa lista {"name": "Thiago"}
23   //enviando para o named constructor que faz o mapeamento de cada objeto devolvendo tudo como lista
24   final List<Person> people = mock.map((element) => Person.fromMap(element)).toList();
25
26   //chamando um for para imprimir a lista de pessoas
27   for(var person in people) {
28     print(person);
29   }
30 }
```

Run

Console

Nome: Thiago  
Nome: Ze

Documentation

3 – Adicione ao exercício anterior 4 atributos, são eles: peso, idade e altura. Também adicione um método que verifique se o usuário tem mais de 18 anos, caso sim ele informe: maior de idade, se não ele deverá informar que não é maior de idade. Imprimir todos esses resultados via toString e/ou chamando através de print conforme linha 28 da imagem do exercício número 2.

**ps.: Ao adicionar outras variáveis, provavelmente vai dar erro em algum lugar. Tente descobrir sem perguntar para os devs/tech lead. Dica: vamos ter variáveis de diversos tipos de dados. String, int, double, etc.**

4 – Crie uma classe chamada Animal com um named constructor para receber os atributos \_id, name, diet e status, conforme endpoint a seguir. Faça uma variável de mock com uma lista de 5 objetos, sendo que alguns atributos diet e status devem vir vazios. Caso venha vazio, deve ser informado para o usuário: “não informado”.

Link para a API: <https://api.got.show/api/show/animals>

```
{ "habitat": [], "range": [], "_id": "5cc0746504e71a0010b85853", "name": "Animals and Plants", "diet": "", "status": "", "__v": 0 }
```

5 – Vá até o site:

<https://jsonplaceholder.typicode.com/>

Clique em Run Script. O site vai consumir a API e trazer um JSON com 4 atributos.

Crie uma classe com o nome que achar necessário para consumir esse json através de uma variável mock sendo chamada no método void main().

Basicamente o exercício é só realizar o consumo mockado e conversão de JSON para objeto através de um named constructor.

6 – Crie uma classe para mapear o objeto a seguir através de named constructor.

```
{
  "id": "93f08018-xxx-yyyy-zzzz-516b48bc6060",
  "title": "Projeto novo",
  "description": "Esse é um projeto que visa ensinar a utilização de construtores",
  "start_date": "10/12/2022 09:10:00",
  "expected_end_date": "23/12/2022 10:32:10",
  "amount_people": 1
}
```

7 – Crie um mixin chamado PersonName, onde haverá um método que receba por parâmetro o nome da pessoa e retorne apenas as três primeiras letras em maiúsculo. Se receber null, deve retornar “BAD”.

A seguir crie uma classe chamada Person que utilize esse mixin.

Essa classe deverá receber o nome por construtor e mostrar o resultado do método do mixin através do toString da classe Person.

Ao final de tudo, instanciar um objeto person enviando o nome e imprimir o resultado através de comando print

8 – (by Welinton)

As organizações CSM resolveram dar um aumento de salário aos seus colaboradores e lhe contrataram para desenvolver o programa que calculará os reajustes.

a. Crie uma classe pessoa com os seguintes atributos: nome, data de nascimento, sexo.

**ATENÇÃO:** Essa classe não pode ser instanciada diretamente.

b. Crie uma classe Funcionário que estende da classe Pessoa com os seguintes atributos: id do funcionário, salário atual e cargo; "Lembrando que é necessário enviar as informações para a classe Pessoa no momento de instanciar a classe Funcionário"

Crie uma classe Rh com os seguintes atributos: cnpj, nome da empresa

c. Crie um método "calcularAumentoSalarial" na classe Rh que recebe um Funcionário como parâmetro nomeado e aplica os seguintes critérios:

- Salários até R\$ 280,00 (incluindo): aumento de 20%;
- Salários entre R\$ 280,00 e R\$ 700,00: aumento de 15%;
- Salários entre R\$ 700,00 e R\$ 1500,00: aumento de 10%;
- Salários de R\$ 1500,00 em diante: aumento de 5%

Após o aumento ser realizado; informe na tela;

- a. O salário antes do reajuste;
- b. O percentual de aumento aplicado;
- c. O valor do aumento;
- d. O novo salário, após o aumento.

9 – ... aguarde

10 - ... aguarde