{reprograma}

#ReforçoDasManas

```
let aprender = "esforço, dedicação, paciência"
if (aprender > 0) {
    return console.log(
        "BORA REPROGRAMAR O MUNDO"
        )
} else {
    console.log(
        "baixo astral aqui nao!!! simbora tentar dnv"
        )
}
```

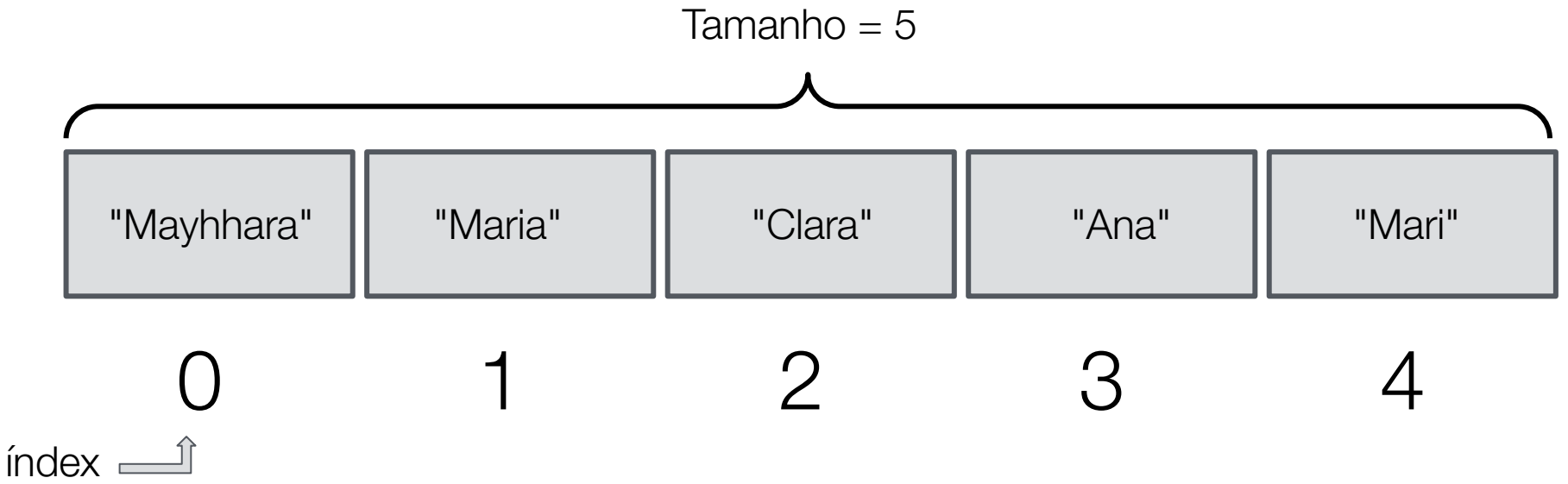
Lógica de Programação

Arrays

Também conhecidos como vetores ou listas

É um conjunto de dados ordenados guardados numa mesma variável

Ex: Array de nomes



Bora resumir?

- **Arrays** permitem o armazenamento ordenado de dados semelhantes
- **index** é a posição de cada item na lista, começando sempre em zero
- O último item da lista tem sempre **index = length - 1**
- **.push()** adiciona um item ao fim da lista
- **.pop()** remove um item do fim da lista
- **.unshift()** adiciona um item ao início da lista
- **.shift()** remove um item do início da lista

Lógica de Programação

For Loop e While Loop

Sintaxe

```
for (let item of array) {
  Enquanto a condição
  acima for verdadeira, o
  código entre as chaves se
  repetirá em loop
}
```

Código

```
for (let nome of nomes) {
    console.log(nome);
}
```

Sintaxe

```
for (setup; condição; passo) {
  Enquanto a condição
  acima for verdadeira, o
  código entre as chaves se
  repetirá em loop
}
```

Código

```
for (let i = 0; i < 11; i++) {
    console.log("Contando
    de 0 a 10: " + i);
}
```

Sintaxe

```
while (condição) {
  Enquanto a condição
  acima for verdadeira, o
  código entre as chaves se
  repetirá em loop
}
```

Código

```
while (5 > 3) {
    console.log("Não faça
    isso. Isso é um loop
    infinito");
}
```

Bora resumir?

- **while** permite executar um código em loop, enquanto determinada condição for verdadeira.
- **break** interrompe a execução do loop imediatamente
- **contiue** interrompe a execução da iteração atual, passando para a próxima iteração do loop

.

Lógica de Programação

Métodos de Strings

Métodos de Strings

- indexOf() e includes()
- lastIndexOf()
- slice()
- substr()
- replace()
- toUpperCase()
- toLowerCase()
- trim()
- split()
- padStart()
- padEnd()

indexOf(text)

Encontra uma string dentro de outra string.

O método retorna o índice da primeira ocorrência de text em uma string.

Exemplo

```
1
2   const frase =  “Eu amo a praia
    pernambucana, e amo muito!”;
3
4   const index = frase.indexOf("amo");
5
6   console.log(index);
7
```

slice(start, end)

Extrai parte de uma string, retornando uma nova string

O método recebe o índice de início e fim (excluso) da extração, e retorna uma nova string.

Exemplo

```
1
2   const texto = "Esse texto é muito
    grande e por isso não cabe no meu
    card. Vou pegar só os primeiros 10
    caracteres para não ficar poluído
    na minha linda página que eu
    aprendi a fazer na Reprograma.”;
3
4   const resumo = texto.slice(0,10);
5   console.log(resumo);
6
```

substr(start, length)

Extrai parte de uma string, retornando uma nova string.

O método recebe o índice de início e tamanho da extração, retornando uma nova string.

Exemplo

```
1
2   const texto = "Um texto qualquer";
3
4   const parte = texto.substr(3,5);
5   console.log(texto);
6   console.log(parte);
7
```

replace(text, newText)

Substitui text por newText em uma string.

O método procura por text numa string e substitui a primeira ocorrência por newText, ao encontrar.

Exemplo

```
1
2   const texto =  “Eu amo a praia pernambucana,
    amo muito”;
3
4   const parte = texto.replace("amo","adoro");
5   console.log(parte);
6
```

toUpperCase()

Coloca todas as letras de uma determinada string em maiúsculas.

O método toLowerCase() é o oposto, e coloca todas as letras de uma string em minúsculas.

Exemplo

```
1
2   const texto = “Eu amo a praia pernambucana,
    amo muito”;
3
4   const gritando = texto.toUpperCase();
5   console.log(gritando);
6
```

trim()

Remove espaços em branco tanto no início quanto no fim de uma string.

Exemplo

```
1
2   const email = "      mayhhara@gmail.com    ";
3   const formatado = email.trim();
4   console.log(`_${formatado}_`);
5
```

padStart(length, text)

Preenche uma string, colocando *text* no início dela (várias vezes, se necessário) até que a string atinja o tamanho *length*

Exemplo

```
const ultimos4dig = "2345";

const numCartao = ultimos4dig.padStart(19, "**** ");
console.log(numCartao);

//**** **** **** 2345
```

split(text)

Transforma uma string num array de strings menores, separando-a em todos os lugares onde encontrar o argumento *text*.

Exemplo

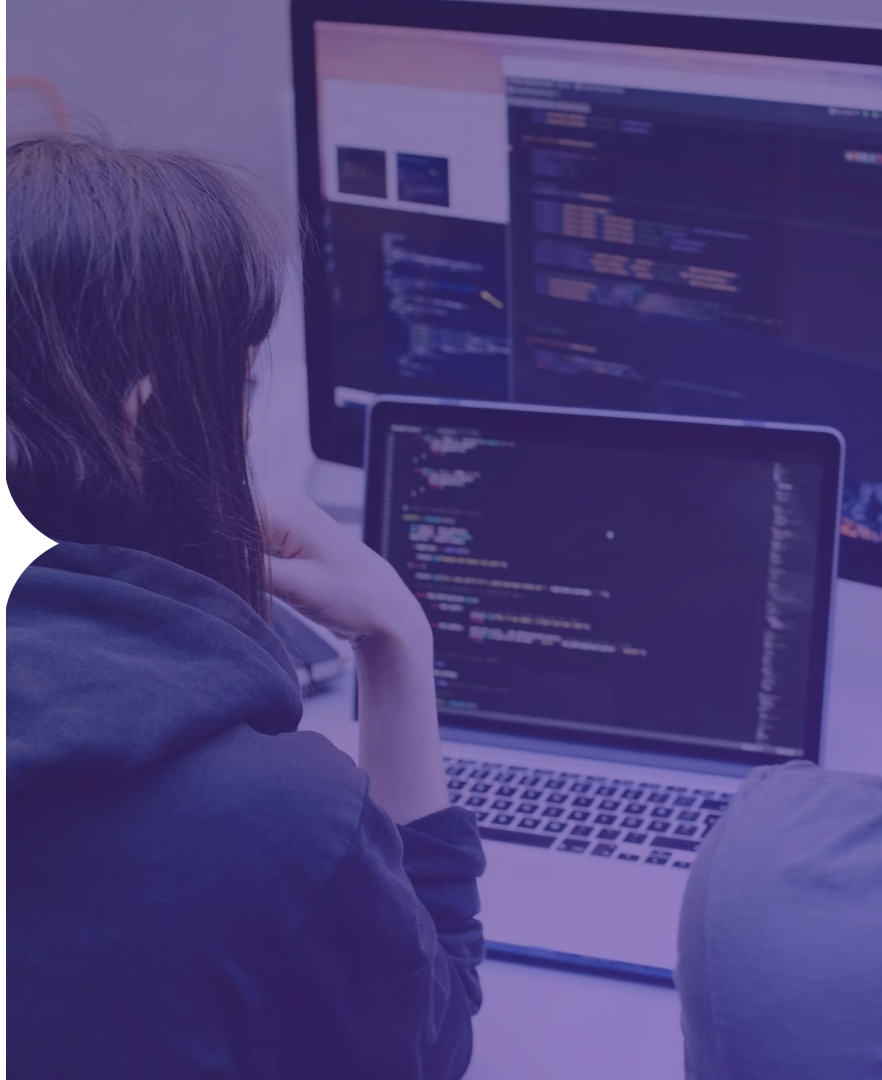
```
1   const nomeCompleto = "Jose Messias Pereira"
2   const array = nomeCompleto.split(" ");
3   const nomeDoMeio = array[1];
4
5   console.log(nomeDoMeio);
6   //Messias
```

Lógica de Programação

Métodos de Arrays

Já vimos antes:

- length (propriedade)
- .push() - insere no fim
- .pop()- remove do fim
- .unshift() - insere o inicio
- .shift() - remove do início



Métodos de Arrays

- indexOf() e includes()
- concat()
- splice()
- join()
- reverse()
- slice()
- toString()

indexOf()

Encontra um item dentro de um array, retornando o índice da primeira ocorrência do item procurado.

Doc completa aqui

Exemplo

```
1   const lista = ["Eu", "Tu", "Eles"];
2   const indice = lista.indexOf("Tu");
3   console.log(indice); //1
```

includes()

Verifica se existe um item dentro de um array, retornando true ou false.

Doc completa aqui

Exemplo

```
1
2   const lista = ["Eu", "Tu", "Eles"];
3   const boolean = lista.includes("Tu");
4   console.log(boolean); //true
```

reverse()

Inverte os itens de um array.

Os últimos serão os primeiros.

Doc completa agui

Exemplo

```
2   const opcoes = ["Eu", "você", "ele"];
3   const resultado = opcoes.reverse();
4   console.log(resultado);
5   //[ 'ele', 'você', 'Eu' ]
```

join()

Junta todos os elementos de um array em uma string e retorna esta string.

Opcionalmente, pode-se passar uma string como argumento para ser colocada entre os itens.

Doc completa aqui

Exemplo

```
2   const opcoes = ["Eu", "você", "ele"];
3   const resultado = opcoes.join(" ou ");
4   console.log(resultado);
5   //Eu ou você ou ele
```

concat()

Retorna um novo único array que é a concatenação de todos os arrays passados como argumento.

Doc completa aqui

Exemplo

```
2   const a = [1,2];
3   const b = [3,4];
4   const c = [5,6];
5   const resultado = a.concat(b, c);
6
7   console.log(resultado);
8   //[ 1, 2, 3, 4, 5, 6 ]
```

slice()

Faz uma cópia de parte de um array, retornando essa cópia.

O array original não é modificado.

Doc completa aqui

Exemplo

```
2   const opcoes = ["Eu", "você", "ele", "nós"];
3   const resultado = opcoes.slice(1,3);
4   console.log(resultado);
5   //[ 'você', 'ele' ]
```

splice()

Altera o conteúdo de uma lista, removendo zero ou mais itens de um array num determinado índice.

Opcionalmente, pode-se inserir um ou mais itens no local onde outros foram deletados.

Doc completa aqui

Exemplo

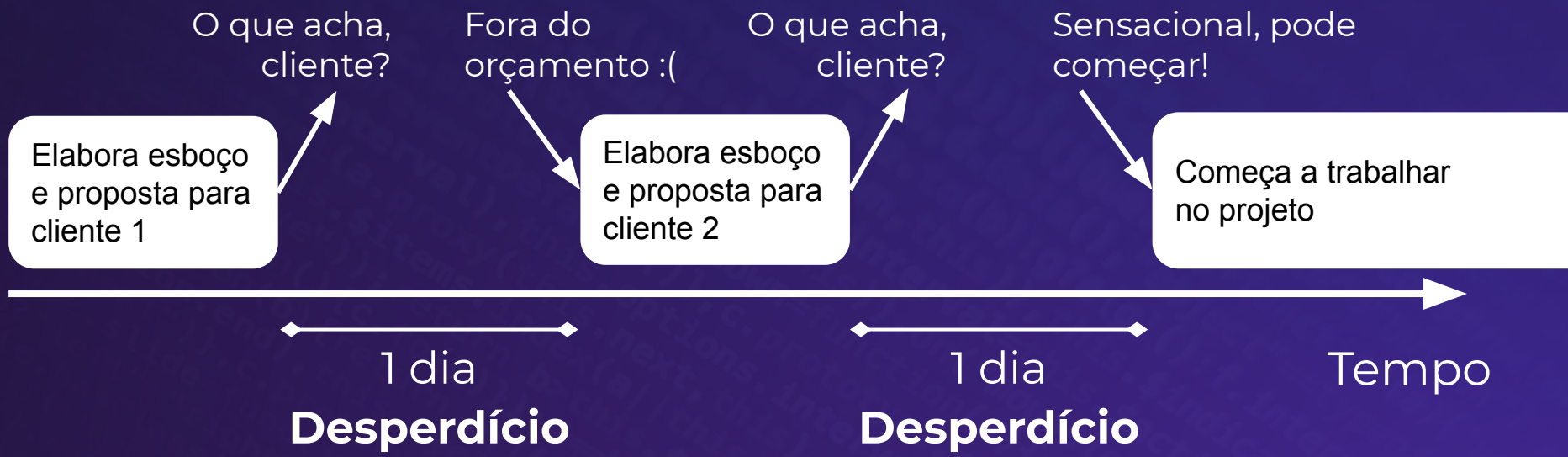
```
1
2   const lista = ["a","b","c","d","e"];
3   lista.splice(1, 2, "f");
4   console.log(lista);
5   //[ 'a', 'f', 'd', 'e' ]
```

Assincronicidade

Conheça Josefa

Web designer freelancer ✨

- Possíveis clientes entram em contato
- Ela elabora e envia um esboço de design + proposta de valor e prazo
- Ela espera resposta dos clientes antes de começar o projeto

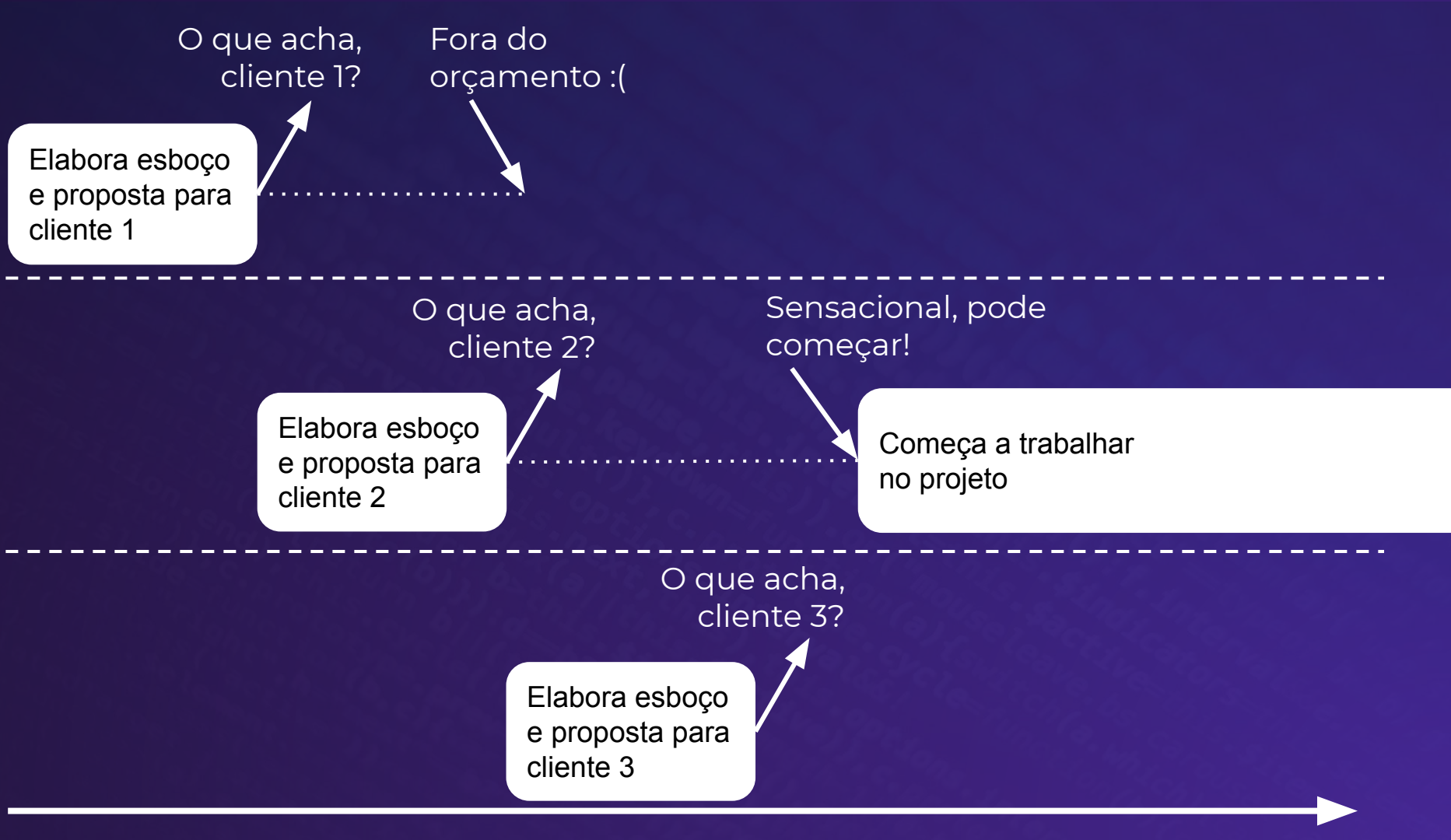

O que acha, cliente?
Fora do orçamento :(
O que acha, cliente?
Sensacional, pode começar!
Elabora esboço e proposta para cliente 1
Elabora esboço e proposta para cliente 2
Começa a trabalhar no projeto
1 dia
Desperdício
1 dia
Desperdício
Tempo

Josefa tá chateada

Não fazer nada enquanto espera resposta de clientes está desperdiçando tempo

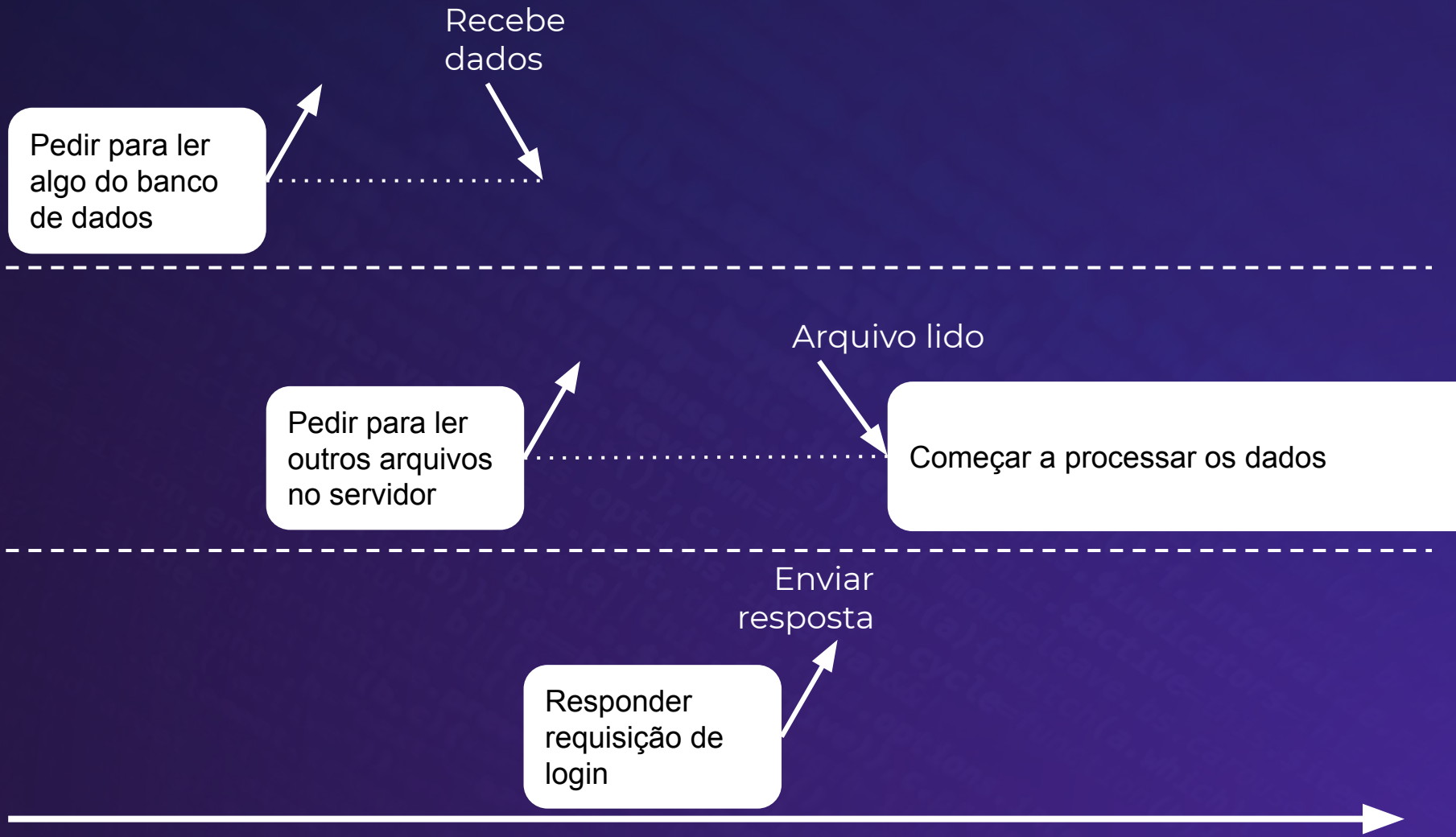
Como resolver a situação? 🤔

✨ Assincronicidade ✨


O que acha, cliente 1?
Fora do orçamento :(
Elabora esboço e proposta para cliente 1
O que acha, cliente 2?
Sensacional, pode começar!
Elabora esboço e proposta para cliente 2
Começa a trabalhar no projeto
O que acha, cliente 3?
Elabora esboço e proposta para cliente 3

Ou seja:

assincronicidade é quando a gente usa o tempo que seria desperdiçado esperando para outras tarefas


Recebe dados
Pedir para ler algo do banco de dados
Arquivo lido
Pedir para ler outros arquivos no servidor
Começar a processar os dados
Enviar resposta
Responder requisição de login

Lógica de Programação

filter()

Docs aqui

Retorna um novo array filtrando elementos do array original de acordo com algum critério a ser definido

filter()

- Retorna um novo array (não modifica o original)
- O callback tem 3 argumentos: o item atual, o índice e o array
- Quando o callback retorna **true** o elemento fica
- Quando o callback retorna **false** o elemento sai
- O callback é chamado para cada item que existia inicialmente. Elementos adicionados posteriormente não são chamados.

Lógica de Programação

map()

Docs aqui

Retorna um novo array fazendo modificações em cada um dos itens

map()

- Retorna um novo array (não modifica o original)
- O callback tem 3 argumentos: o item atual, o índice e o array
- O callback só é chamado para elementos que forem setados.
- O callback é chamado para cada item que existia inicialmente, antes da primeira chamada. Elementos adicionados posteriormente não são chamados.

Lógica de Programação

forEach

Docs aqui

É uma forma mais simples de varrer um array utilizando uma callback

Lógica de Programação

find()

Docs aqui

É uma método de arrays que encontra um determinado elemento num array, retornando-o

Observação importante

É retornado apenas o primeiro elemento que atende a condição buscada

Lógica de Programação

findIndex()

Docs aqui

É uma método de arrays que encontra um determinado elemento num array, retornando o índice desse elemento.

Lógica de Programação

every()

Docs aqui

É um método de arrays que verifica se **todos** os elementos atendem a condição passada no callback

Lógica de Programação

some()

Docs aqui

É uma método de arrays que verifica se existe algum elemento que atende a condição passada no callback

Lógica de Programação

sort()

É uma método de arrays que coloca seus elementos em ordem. Para isso, é possível passar uma função callback de comparação.

Observação importante

Caso não seja passada uma função de comparação será utilizada uma função padrão que ordena pelos valores UNICODE.

Lógica de Programação

localeCompare()

Docs aqui

É uma método de comparação que ordena alfabeticamente levando em consideração o idioma e seus caracteres especiais.

Lógica de Programação

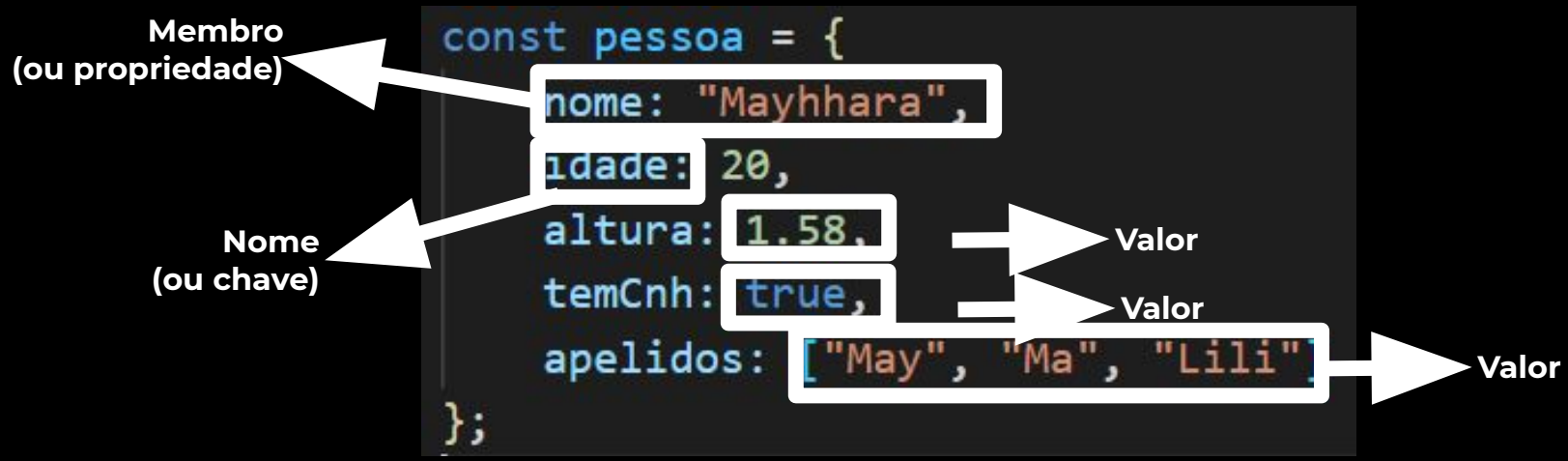
Objetos e Funções

Lógica de Programação

Objetos

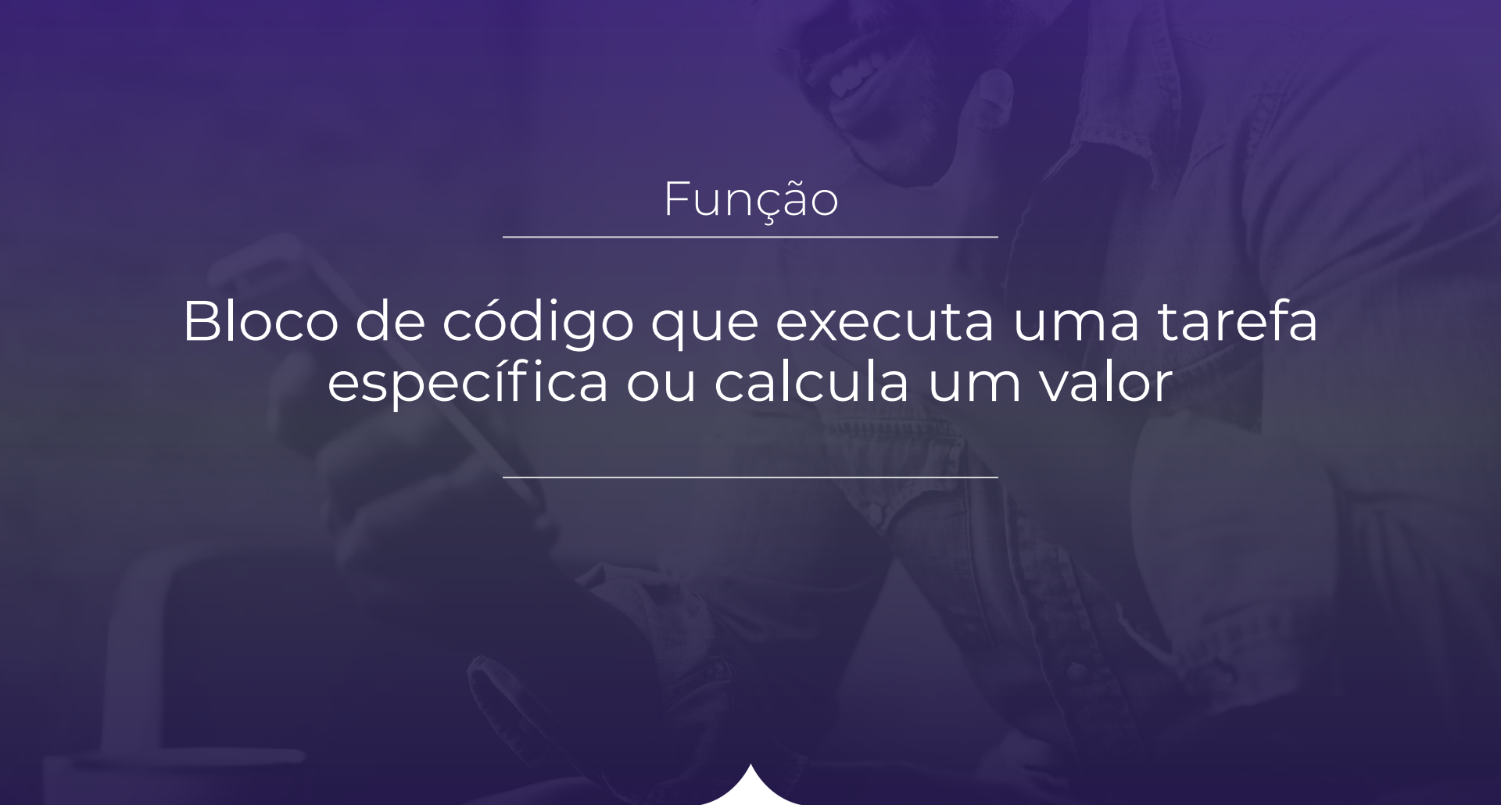
Objeto

É um **conjunto de dados** ou funcionalidades relacionadas


Membro
(ou propriedade)
Nome
(ou chave)
const pessoa = {
nome: "Mayhhara",
idade: 20,
altura: 1.58,
temCnh: true,
apelidos: ["May", "Ma", "Lili"]
};
Valor
Valor
Valor

Lógica de Programação

Funções



Função

Bloco de código que executa uma tarefa específica ou calcula um valor

Sintaxe

```
function nomeDaFunc (params) {
  //seu código aqui
}
```

Código

```
function imprimirOla ( ) {
    console.log("Olá!");
}
```