

📄 rocketseat-education / bootcamp-launchbase-desafios-01

Code Issues 1 Pull requests 2 Actions Projects Security Insights

Join GitHub today

Dismiss

GitHub is home to over 50 million developers working together to host and review code, manage projects, and build software together.

Sign up

🔑 master ▾

...

bootcamp-launchbase-desafios-01 / desafios / 01-4-aplicacao-operacoes-bancarias.md

 vinifraga ✎ improved delivery text



👤 1 contributor

Raw Blame



121 lines (80 sloc) 3.94 KB



LaunchBase

BOOTCAMP

Desafio 1-4: Aplicação: Operações bancárias

“Não compare o seu bastidor com o palco do outro!”

[Sobre o desafio](#) | [Entrega](#) | [Licença](#)

Sobre o desafio

Desafios para fortalecer alguns conceitos, entre eles:

- Booleanos;
- Organização;
- Padronização;
- Escrita.

Intro

Crie um programa para realizar operações bancárias na conta de um usuário.

Comece criando um objeto com o nome do usuário, suas transações e saldo.

As transações (transactions) devem iniciar como um array vazio `[]` e o saldo (balance) em `0` (zero).

```
const user = {  
  name: "Mariana",  
  transactions: [],  
  balance: 0  
};
```

Adicionar transações

Crie uma função `createTransaction` para adicionar uma nova transação no array de transações de um usuário, essa função deve receber como parâmetro um objeto de transação que tem o seguinte formato:

```
{  
  type: 'credit',  
  value: 50.5  
}
```

O `type` pode ser `credit` para créditos e `debit` para débitos da conta do usuário.

Quanto uma transação for do tipo `credit` ela deve também somar o valor do crédito no saldo (balance) do usuário.

Se for uma transação do tipo `debit` ela deve subtrair o valor do débito no saldo (balance) do usuário.

Dica.: Você pode usar o método `user.transactions.push(transaction)` para adicionar um novo item no array de transações.

Relatórios

- Crie uma função chamada `getHigherTransactionByType` que recebe como parâmetro o tipo de transação `credit/debit`, percorre as transações do usuário e retorna o **objeto** da transação de maior valor com aquele tipo:

```
getHigherTransactionByType("credit"); // { type: 'credit', value: 120 }
```

- Crie uma função chamada `getAverageTransactionValue` que retorna o valor médio das transações de um usuário independente do seu tipo:

```
getAverageTransactionValue(); // 83.3
```

- Crie uma função chamada `getTransactionsCount` que retorna o número de transações de cada tipo `credit/debit`, o retorno da função deve ser um objeto e seguir exatamente como o modelo apresentado abaixo:

```
getTransactionsCount(); // { credit: 5, debit: 3 }
```

Exemplo de resultado final do projeto:

```
createTransaction({ type: "credit", value: 50 });
createTransaction({ type: "credit", value: 120 });
createTransaction({ type: "debit", value: 80 });
createTransaction({ type: "debit", value: 30 });

console.log(user.balance); // 60

getHigherTransactionByType("credit"); // { type: 'credit', value: 120 }
getHigherTransactionByType("debit"); // { type: 'debit', value: 80 }

getAverageTransactionValue(); // 70

getTransactionsCount(); // { credit: 2, debit: 2 }
```



Entrega

Esse desafio **não precisa ser entregue** e não receberá correção, mas você pode ver um exemplo de solução [aqui](#). Após concluí-lo, adicionar esse código ao seu Github é uma boa forma de demonstrar seus conhecimentos para oportunidades futuras.



Licença

Esse projeto está sob a licença MIT. Veja o arquivo [LICENSE](#) para mais detalhes.

Feito com 💜 by [Rocketseat](#) 🙌 [Entre na nossa comunidade!](#)