Módulo 23

Exercícios - Eventos em Node.js

Lembre-se de que os exercícios podem ser resolvidos com diferentes abordagens, e nosso gabarito é apenas uma forma de implementação.

Nesta etapa do curso, você encontrará uma lista de exercícios/desafios que foram projetados para estimular sua curiosidade e promover uma abordagem prática no aprendizado de programação. Embora alguns dos conceitos abordados nesses exercícios ainda não tenham sido explicados detalhadamente, essa abordagem faz parte da nossa metodologia de ensino.

Objetivo dos Desafios:

- **Simular a Vida de Programador:** Ao se deparar com problemas que envolvem conceitos ainda não estudados, você terá a oportunidade de desenvolver habilidades de pesquisa e resolução de problemas, semelhantes às que um programador enfrenta no seu dia a dia.
- **Incentivar a Pesquisa:** A ideia é que você procure soluções, entenda novos conceitos e aprenda a aplicar técnicas que ainda serão discutidas nas aulas futuras. Isso ajudará a solidificar seu entendimento e prepará-lo para os tópicos que serão abordados em breve.
- **Preparação para o Futuro:** Ao resolver os desafios e pesquisar os conceitos, você estará se preparando para as explicações detalhadas que virão posteriormente no curso. Cada exercício é uma oportunidade para explorar e aprender de forma independente.

Como Abordar os Exercícios:

1. **Pesquise e Experimente:** Antes de procurar a solução completa, tente entender o problema e busque informações sobre os conceitos que estão relacionados. Use recursos como documentação, fóruns e tutoriais.
2. **Resolva com Criatividade:** A abordagem não precisa ser perfeita. O objetivo é experimentar e ver o que funciona, e o que não funciona.
3. **Reflita e Revise:** Após encontrar e entender a solução, revise o exercício e compare com o que foi discutido nas aulas subsequentes. Isso ajudará a reforçar seu aprendizado.

O que Esperar Após os Desafios:

- **Explicação Detalhada:** Após concluir os desafios, você receberá uma explicação detalhada dos conceitos envolvidos e das melhores práticas para resolver os problemas. Isso garantirá que você compreenda completamente o que foi proposto e como aplicar o conhecimento em situações futuras.

Prepare-se para um desafio enriquecedor e lembre-se: a pesquisa e a prática são ferramentas essenciais para se tornar um programador habilidoso. Boa sorte e divirta-se explorando!

Exercício 1: Crie um programa Node.js onde você tenha um EventEmitter. Emita um evento chamado `mensagemRecebida` e ouça esse evento para exibir a mensagem "Mensagem recebida com sucesso!".

Exercício 2: Altere o código anterior para que o evento `mensagemRecebida` aceite um argumento contendo a mensagem recebida e exiba essa mensagem no console.

Exercício 3: Crie uma classe chamada `Conversa` que herde de `EventEmitter`. A classe deve ter um método chamado `enviarMensagem` que emita o evento `mensagemEnviada`. O evento deve aceitar um argumento com a mensagem enviada e exibir no console quando a mensagem for enviada.

Exercício 4: Modifique o código da classe `Conversa` para herdar a emissão de eventos e adicione um novo evento chamado `mensagemRecebida`. Ao receber a mensagem, exiba "Nova mensagem recebida!" e a própria mensagem.

Exercício 5: Modifique o código da classe `Conversa` para adicionar dois ouvintes diferentes para o evento `mensagemRecebida`. O primeiro ouvinte deve exibir o conteúdo da mensagem, e o segundo ouvinte deve contar quantas mensagens foram recebidas.

Exercícios 6 ao 10 - Desafios

Exercício 6: Crie um sistema de `login` que herda de `EventEmitter`. O sistema deve emitir um evento `loginAttempt` toda vez que uma tentativa de login é feita. Se o usuário e senha forem corretos, emita o evento `loginSuccess`, caso contrário, `loginFailure`. Use um callback para simular uma operação assíncrona (como consultar um banco de dados) que leva 1 segundo.

Exercício 7: Crie um sistema de fila que herde de `EventEmitter`. A fila deve permitir adicionar "tarefas" (strings) e processá-las uma por vez a cada 2 segundos, emitindo um evento `taskProcessed` cada vez que uma tarefa for completada. Quando todas as tarefas forem processadas, emita um evento `allTasksProcessed`.

Exercício 8: Crie um sistema de log que herda de `EventEmitter` e registra todos os eventos emitidos, como `login`, `logout` e `error`. O sistema deve armazenar as mensagens de log e, ao final de cada dia (simulado por um intervalo de tempo de 5 segundos), emita um evento `dailyLogReport` com um resumo de todos os logs do dia.

Exercício 9: Crie um EventEmitter que emite o evento `ping` a cada segundo. O evento `ping` deve ser emitido no máximo 5 vezes. Após a quinta emissão, o processo deve ser finalizado emitindo um evento `pingFinished`.

Exercício 10: Crie um programa Node.js que leia o conteúdo de um arquivo de texto e, ao terminar de ler, emita um evento `fileReadSuccess` com o conteúdo lido. Em seguida, escreva esse conteúdo em um novo arquivo e, ao finalizar a escrita, emita um evento `fileWriteSuccess`.