

Desvendando os segredos dos Streams.

Gustavo Henrique



Quem sou eu

Gu Henriques (Gustavo Henrique), 28 anos

- Desenvolvedor Backend
- Engenheiro da Computação
- Community Manager da Thasfin
- Embaixador Rockeseat
- Fundador da Yoler Digital



Curiosidade



Introdução.

No mundo da tecnologia, os Streams, também conhecidos como "fluxos de dados", são como rios que transportam informações de forma contínua e eficiente. Imagine um rio fluindo sem parar, carregando consigo dados em vez de água. Essa é a essência dos Streams!

Em vez de armazenar e processar todos os dados de uma vez, os Streams os tratam em pedaços menores, linha por linha, como se estivessem "bebendo" da fonte contínua.

INICIAR →



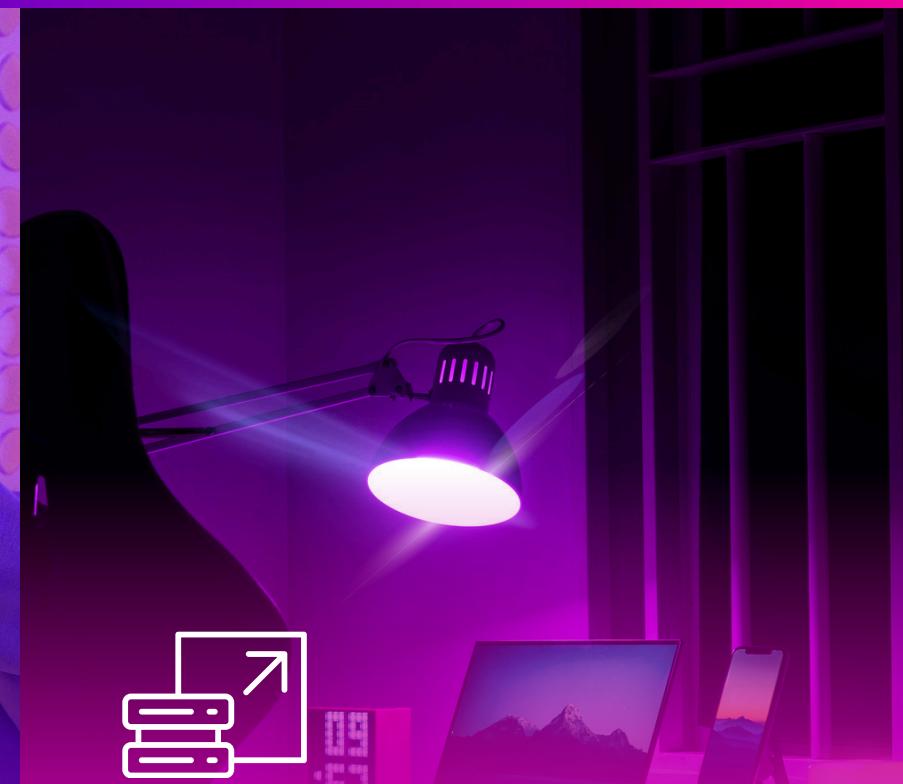


Importância dos Streams

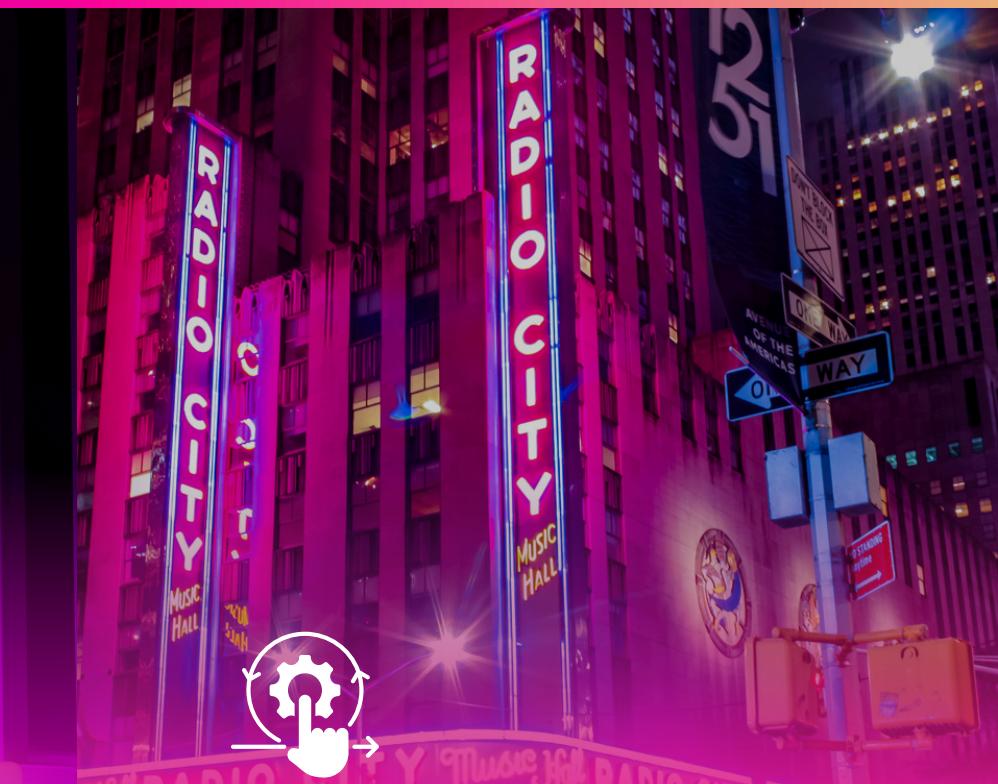
Os Streams são ferramentas essenciais para lidar com grandes volumes de dados de forma eficiente e escalável. Ela permite que informações sejam enviadas e recebidas de forma contínua. Eles são amplamente utilizados em diversas aplicações, como vídeos ao vivo, música online e atualizações de redes sociais.



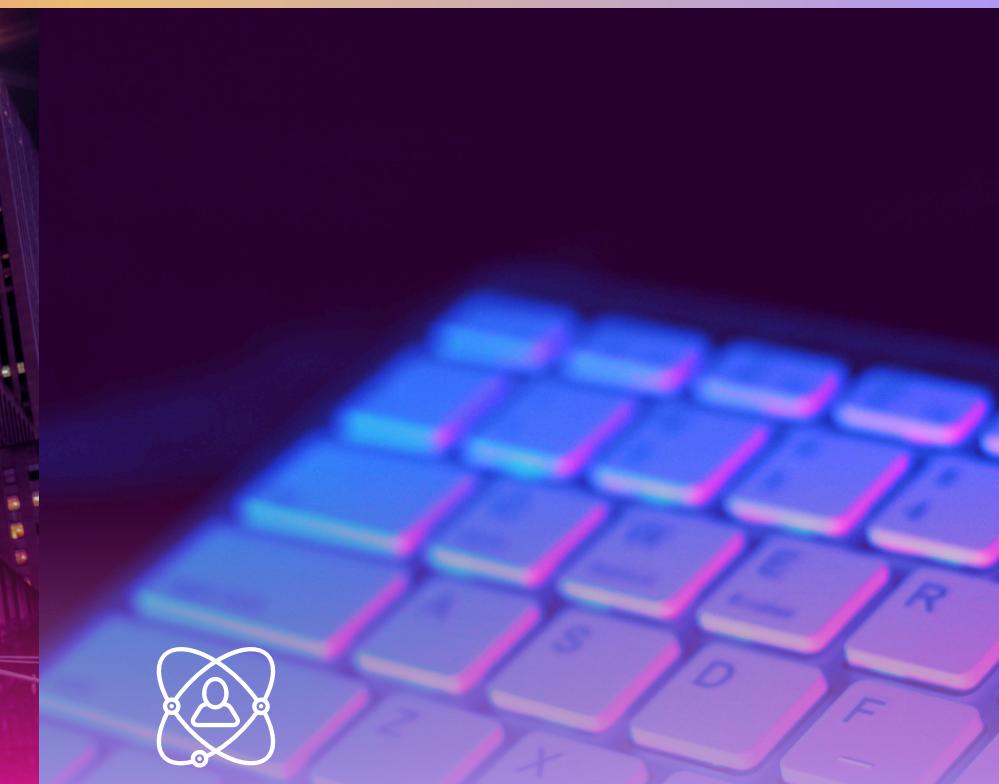
Processamento
Contínuo de dados



Eficiência e
Escalabilidade



Simplicidade e
Flexibilidade



Experiência do
Usuário

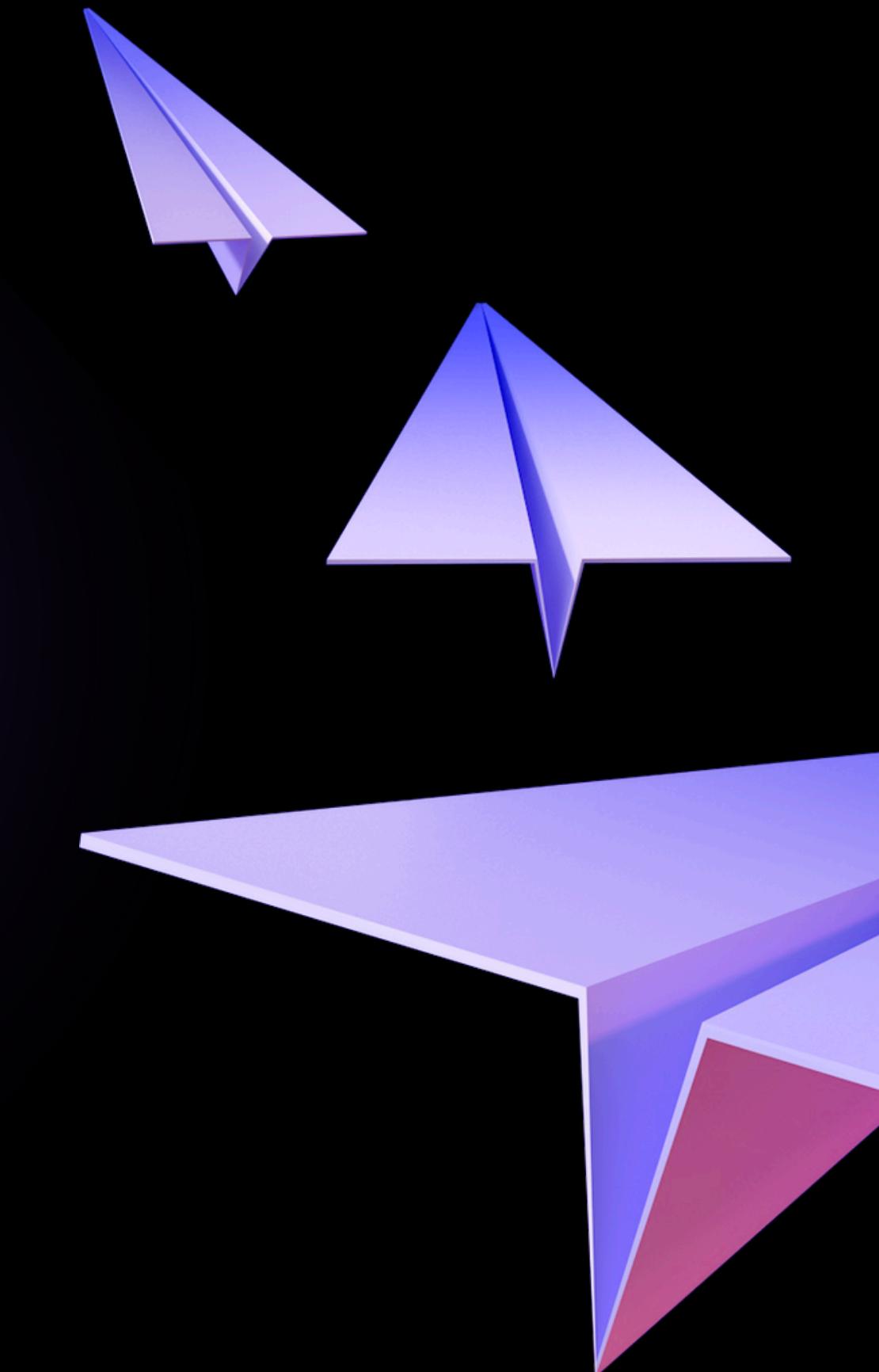
Processamento Contínuo de dados

- * Adeus ao gargalo: Diga adeus ao gargalo do processamento tradicional, onde os dados precisam ser armazenados e processados de forma completa antes da próxima etapa. Com os Streams, você processa os dados linha por linha, sem precisar esperar pelo conjunto completo.
- * Ideal para grandes volumes: Seja lidando com logs de acesso, feeds de mídia social ou sensores de IoT, os Streams permitem processar terabytes de dados com fluidez e sem travamentos.
- * Eventos em tempo real: Torne seus sistemas responsivos e dinâmicos com a capacidade de reagir a eventos instantaneamente, sem atrasos.



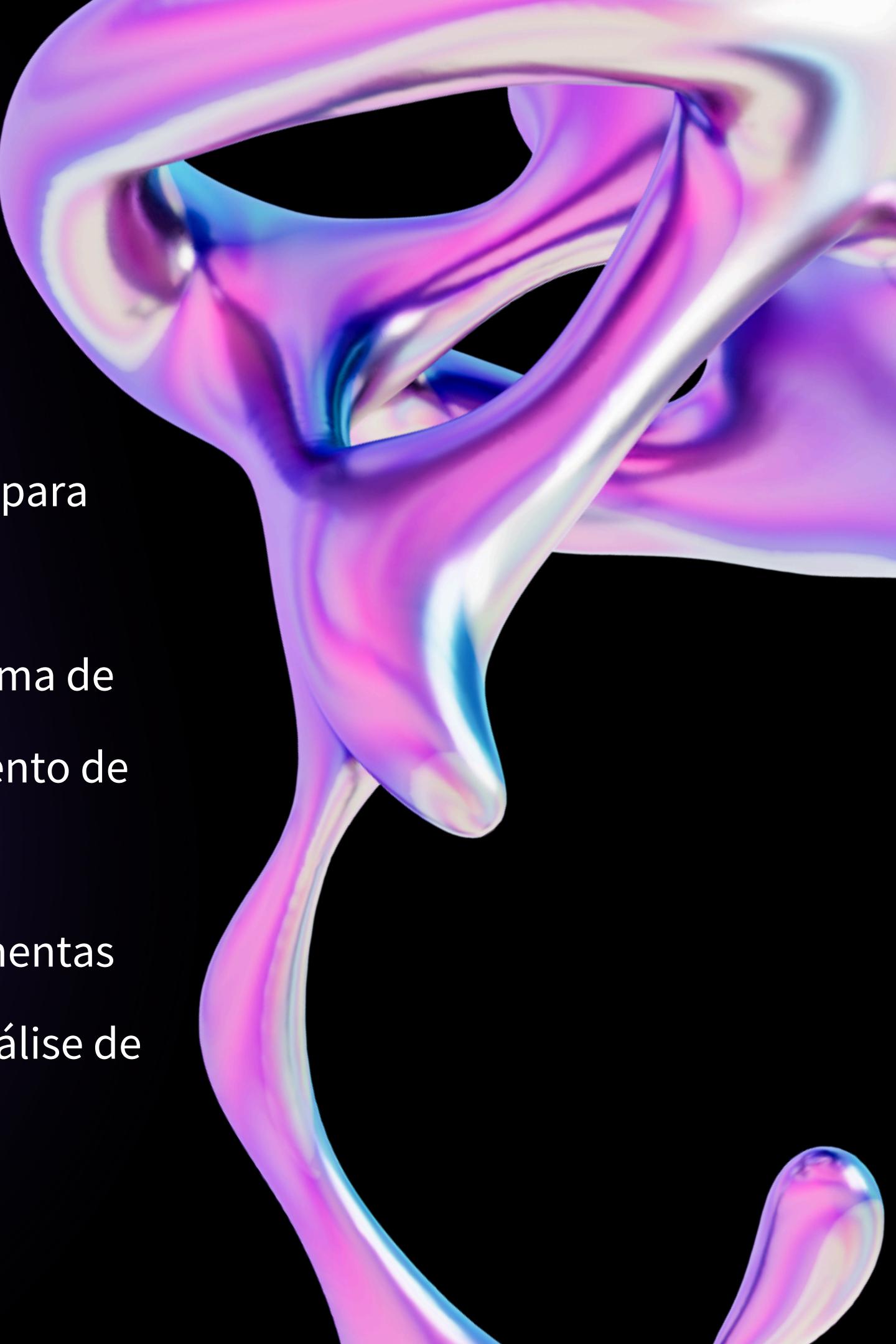
Eficiência e Escalabilidade

- * **Menos memória:** Os Streams utilizam menos memória do que os métodos tradicionais, pois processam os dados em pedaços menores, liberando espaço na memória para outras operações.
- * **Menos recursos de CPU:** O processamento contínuo também significa menor uso de CPU, otimizando o desempenho geral do seu sistema.
- * **Escalabilidade sob demanda:** Prepare-se para lidar com picos de demanda sem precisar se preocupar com infraestrutura complexa. Os Streams se adaptam automaticamente ao volume de dados, garantindo um desempenho consistente.



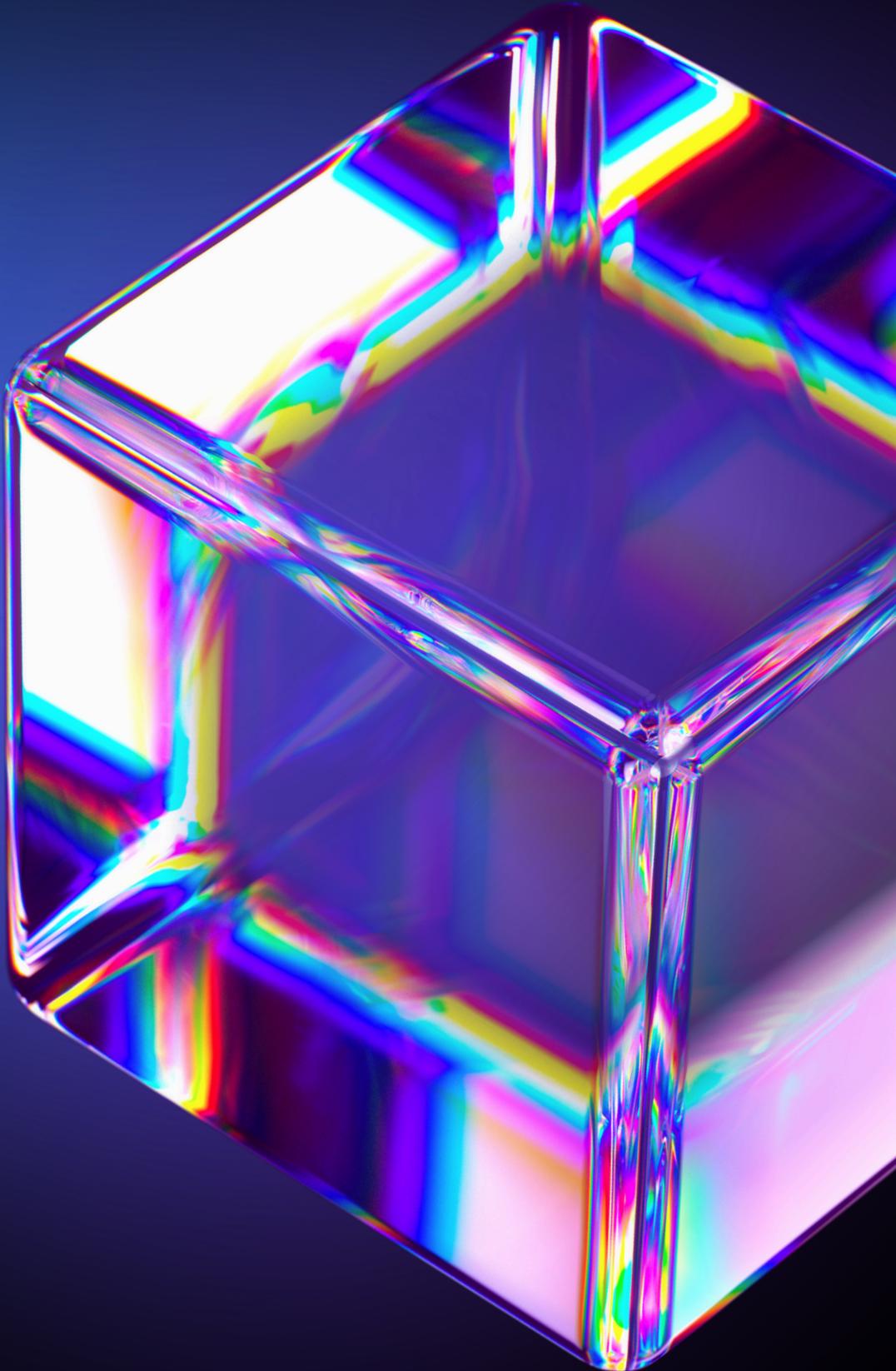
Simplicidade e Flexibilidade

- * **Fácil de usar:** A API dos Streams é intuitiva e fácil de aprender, mesmo para iniciantes.
- * **Flexível para diversas aplicações:** Utilize os Streams em uma ampla gama de cenários, desde APIs RESTful e serviços em tempo real até processamento de arquivos e análise de logs.
- * **Combine com outras ferramentas:** Integre os Streams com suas ferramentas favoritas, como bancos de dados, frameworks web e bibliotecas de análise de dados.



Experiência de Usuário

- * Aplicações mais responsivas: Crie interfaces de usuário mais responsivas e fluidas com a capacidade de processar e exibir dados em tempo real.
- * Transmissões de dados sem interrupções: Ofereça transmissões de vídeo e áudio sem travamentos ou buffering, proporcionando uma ótima experiência para seus usuários.
- * Atualizações instantâneas: Mantenha seus usuários informados em tempo real com atualizações instantâneas de feeds, notificações e dashboards.



Tipos de Streams.

1.

Readable

2.

Writable

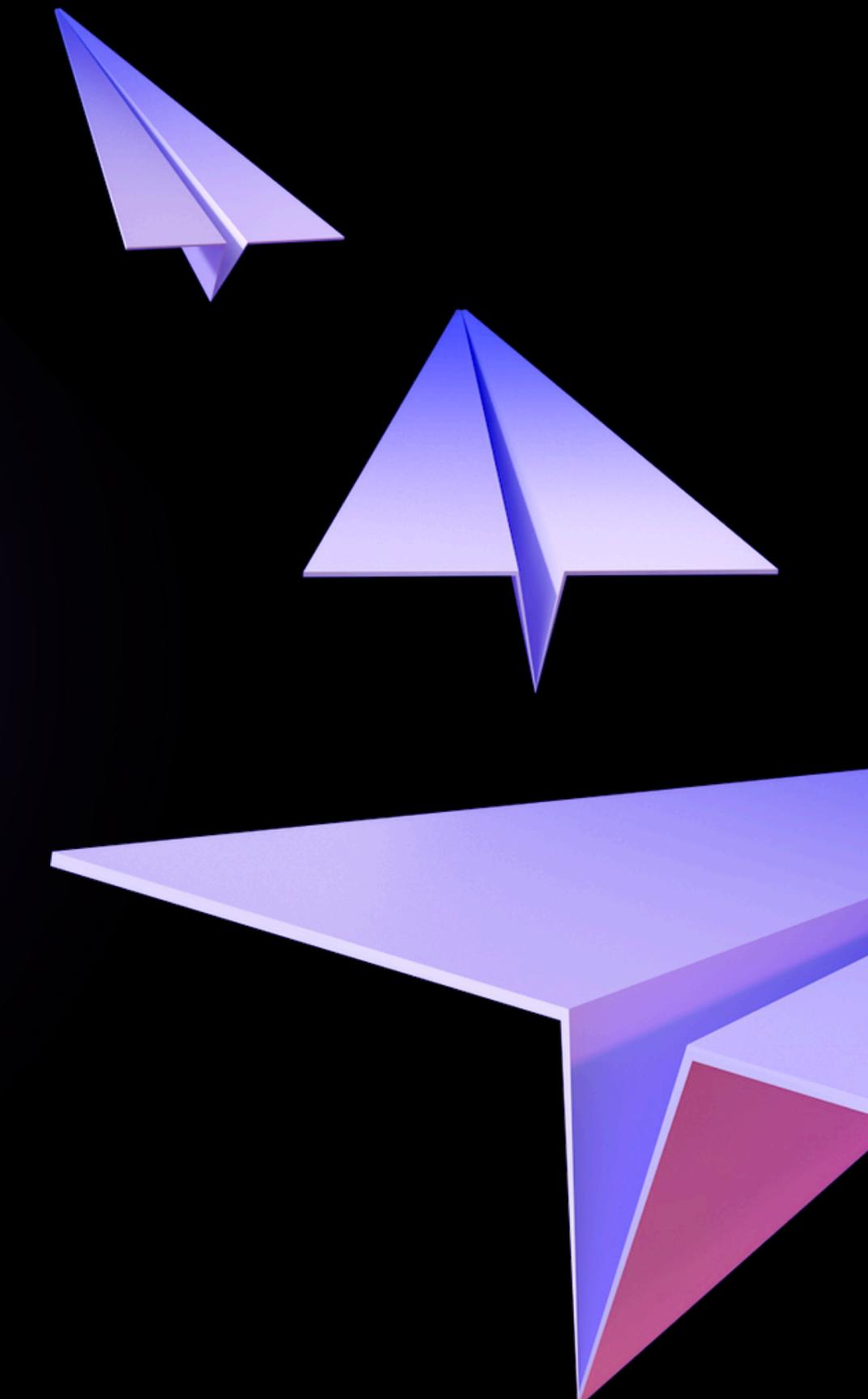
3.

Duplex

Readable

Os streams legíveis são um fluxo constante de dados que podem ser lidos e processados pelo seu programa. Exemplos comuns incluem:

- * **Leitura de arquivos:** Ler o conteúdo de um arquivo linha por linha, sem precisar carregá-lo todo na memória de uma só vez.
- * **Respostas HTTP:** Receber dados de um servidor web em partes, à medida que são enviados.
- * **Processos:** Acessar a saída de um processo como um stream, permitindo que você processe os dados em tempo real.



Writable

Os streams gravados permitem que você escreva e armazene dados de forma eficiente. Exemplos comuns incluem:

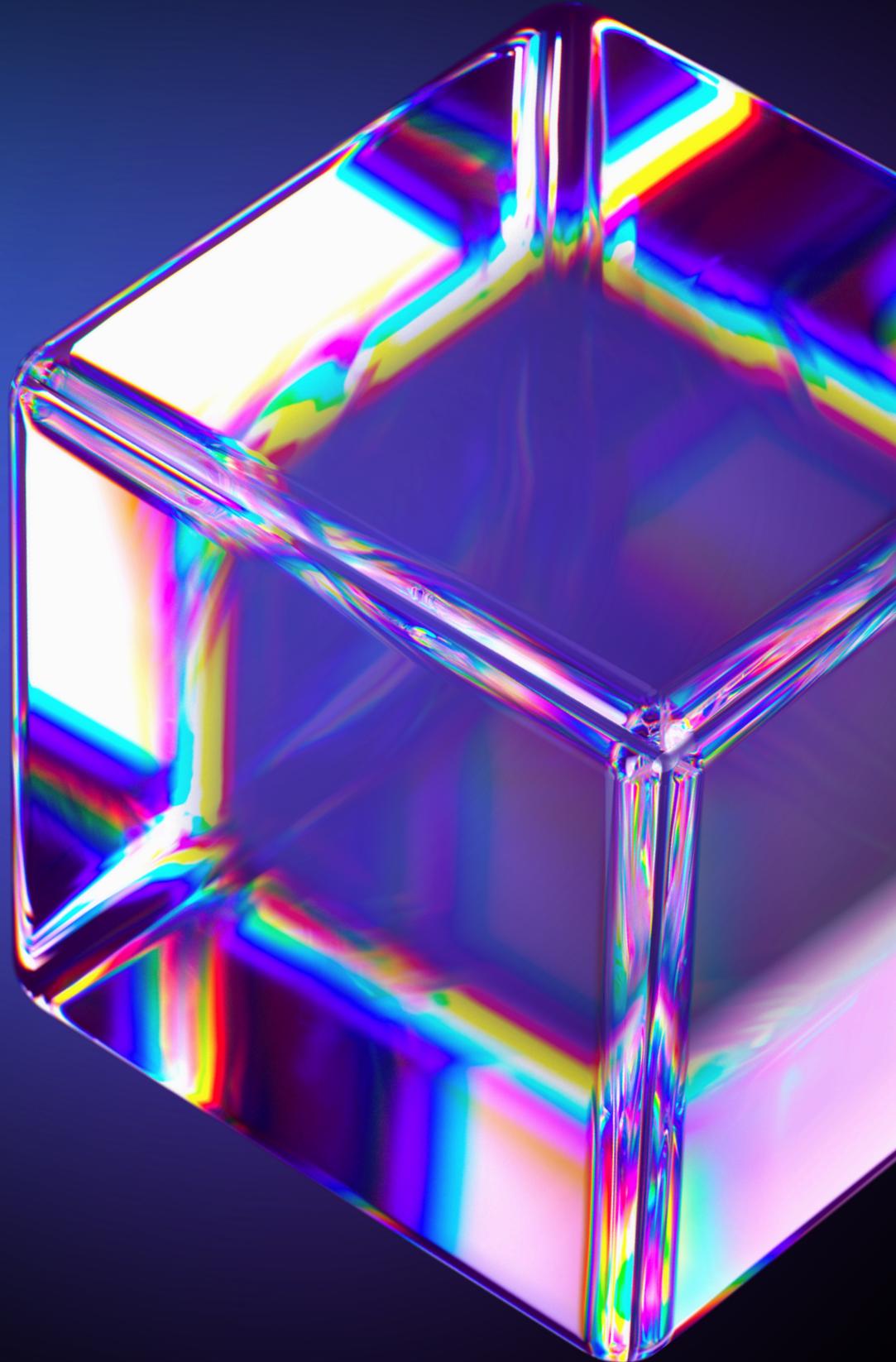
- * **Gravação em arquivos:** Salvar dados em um arquivo linha por linha, sem precisar armazená-los todos na memória de uma só vez.
- * **Requisições HTTP:** Enviar dados para um servidor web em partes, otimizando o tempo de envio.
- * **Consoles:** Escrever mensagens no console da sua aplicação, exibindo informações e logs para depuração.



Duplex

Os streams duplex combinam as características de streams legíveis e gravados, permitindo que você leia e escreva dados no mesmo stream. Exemplos comuns incluem:

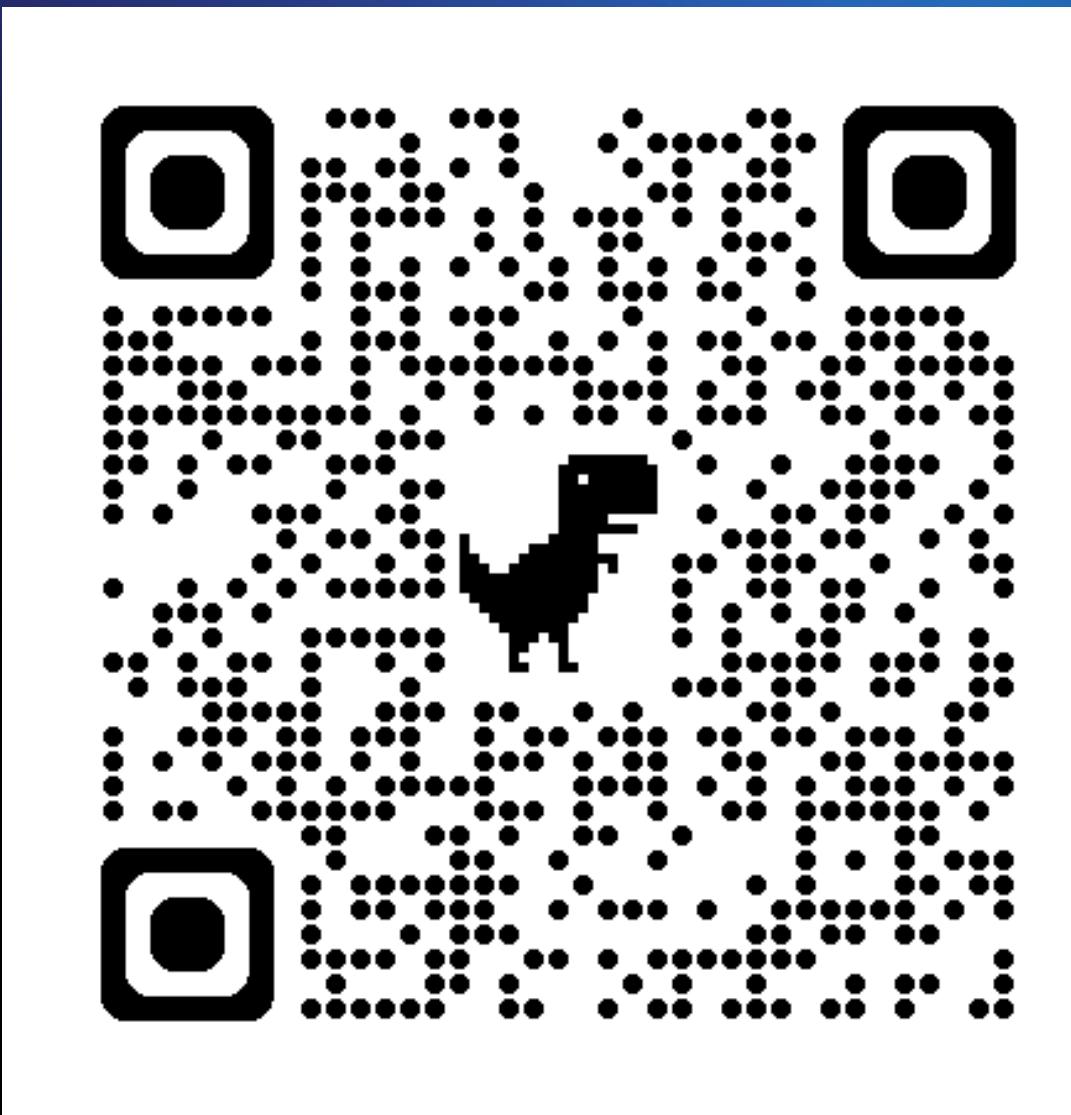
- * **Sockets TCP**: Estabelecer uma conexão bidirecional com outro computador, permitindo a troca de dados em tempo real.
- * **Criar pipes**: Conectar duas streams para que os dados fluam de uma para a outra de forma transparente.
- * **Combinar streams**: Combinar o output de várias streams em um único stream.



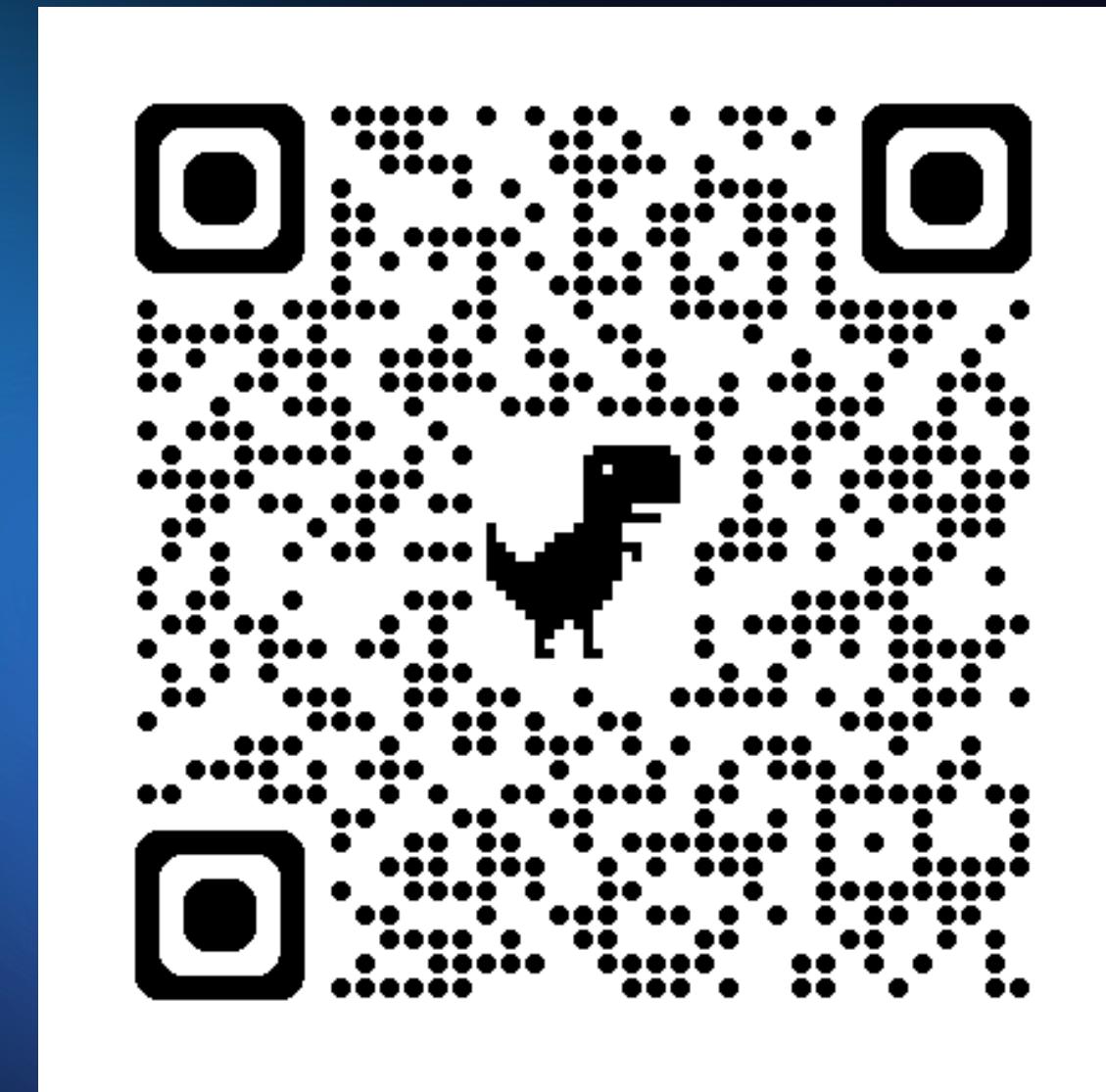


Informações.

Repositório



Fale Comigo



Muito Obrigado!

