

# Relatório Simulador de Propósito Geral

Universidade Federal de Santa Catarina

Nathan Reuter Godinho  
Marcos Donaciano Teixeira Júnior  
26 de Outubro de 2017

## SOBRE

O referente trabalho se trata de uma aplicação web feita em javascript para simular uma fila de dois servidores com duas entidades.

Nessa aplicação é possível entrar com diversos dados como *tempo de chega da fila1*, *tempo de chegada da fila 2*, *tempo de serviço do servidor 1*, *tempo de serviço do servidor 2*, *tempo de falha dos servidores* além da *porcentagem de falha* dos servidores.

Cada parâmetro desse pode receber os seguintes parâmetros probabilísticas: *Constante*, *Exponencial*, *Normal*, *Triangular*, *Uniforme* .

# COMO UTILIZAR

Primeiramente execute com o navegador o arquivo **index.html** localizado na pasta do projeto(para melhor usabilidade, estar conectado a internet).

The screenshot shows the main interface of the 'Simulador de Servidor' (Server Simulator). At the top, there is a logo and the title 'Simulador de Servidor'. Below this, the interface is divided into three main sections: 'Configurações Iniciais', 'Condição de parada', and 'Velocidade da Simulação'. The 'Configurações Iniciais' section includes input fields for 'Tempo Chegada Fila 1', 'Tempo Chegada Fila 2', 'Tempo Serviço Servidor 1', 'Tempo Serviço Servidor 2', 'Tempo de Falha Servidores', and 'Porcentagem de Falha'. The 'Condição de parada' section has input fields for 'Máximo de Entidades' and 'Tempo de Simulação (Default: 200s)'. The 'Velocidade da Simulação' section has radio buttons for 'Ultra Rápida', 'Rápida', 'Média', and 'Lenta'. A footer bar at the bottom contains the text 'Desenvolvido por: [nome] - 2023'.

Ao executar, uma interface como a imagem acima deve aparecer.

Na primeira linha aparece as configurações básica e essenciais para rodar a simulação.

This screenshot is a close-up of the 'Configurações Iniciais' (Initial Configurations) section. It shows the same input fields as the previous image. On the right side, there is a dropdown menu for 'Exponencial' with options: 'Constante', 'Exponencial', 'Normal' (selected), 'Triangular', and 'Uniforme'. Below the dropdown, there are input fields for 'Constante 1' and 'Constante 2'. The 'Condição de parada' section is also visible at the bottom.

Na coluna mais da direita é possível escolher a função probabilística desejada em cada parâmetro.

Mais abaixo é esperado entradas para a condição de parada para simulação. A regra leva primeiramente em conta o *tempo*, que por padrão é 200 segundos de simulação.

Porem é possível parar por *número de entidades*, desde de que o tempo setado seja grande suficiente para simulação parar nessa condição.

Por fim é possível escolher a velocidade de execução do programa.

Com as opções de *ultra rápido*, *rápido*, *média* e *lenta* é possível ver mais detalhadamente ou não a execução do mesmo.

No final, depois das modificações feitas, os parâmetros devem estar preenchidos como a imagem abaixo.

## Simulador de Servidor

### Configurações Iniciais

Tempo Chegada Fila 1	20	Exponencial
Tempo Chegada Fila 2	20	Constante
Tempo Serviço Servidor 1	1 2 3	Triangular
Tempo Serviço Servidor 2	10	Constante
Tempo de Restauração	10	Constante
Porcentagem de Falha	50	Constante

### Condição de parada

Máximo de Entidades	1000
Tempo de Simulação (Opcional, 200s)	2000

### Velocidade da Simulação

☐ Ultra-rápida
☒ Rápida
☐ Média
☐ Lenta

Começar

E agora está tudo pronto, basta clicar em **Começar**.

E se tudo correr bem os dados da simulação devem aparecer como a imagem abaixo:



Enquanto o Status estiver escrito **Running** a simulação estará em execução ainda.

Ao fim da simulação, o status ficara como **finished** e será possível analisar os dados da simulação, ou rodar uma nova no botão vermelho de *voltar*.

*(Obs: Infelizmente os botões para controle de simulação de pausa e continuação não foram terminados de implementar)*

The screenshot displays the 'Simulador de Servidor' (Server Simulator) web application. At the top, there is a circular icon with a server rack and the title 'Simulador de Servidor'. Below this, the status is indicated as 'Status: FINISHED' in red text. The interface is divided into two main sections: 'Dados Gerais' (General Data) on the left and 'Controles Simulação' (Simulation Controls) on the right. The 'Dados Gerais' section lists various statistics: Total de Entidades: 200, Total de Entidades Tipo 1: 100, Total de Entidades Tipo 2: 99, Total de Entidades Disponíveis: 201, Total de Entidades Disponíveis Completadas: 201, Total de Tempo de Simulação: 2208.9529995000005 segundos, Instances na fila do Servidor 1: 0, and Entidades na fila do Servidor 2: 1. Below this, it shows 'Falhas Servidor 1: 20', 'Falhas Servidor 2: 57', and 'Falhas Totais: 77'. Further down, it lists 'Tempo total Falha Servidor 1: 180 segundos', 'Tempo total Falha Servidor 2: 570 segundos', and 'Falhas totais:'. At the bottom, it shows 'Tempo Médias Entidades no Sistema: 26.308403+7203542+ segundos' and 'Número de Trocas entre Servidores: 50'. The 'Controles Simulação' section contains three buttons: a red button with a circular arrow (likely 'Voltar'), a grey button with a play/pause symbol, and a green button with a play symbol. The footer of the application reads 'Desenvolvido por: Rafael Costa - 2023'.

# COMPONENTES DO PROGRAMA

O programa é compilado inteiramente em um único arquivo chamado de **app.js**, onde este é interpretado pelo navegador ao carregar a página no arquivo **index.html**

Dentro do pasta enviada com este trabalho, tem o diretório **src** com os arquivos que montam a lógica do programa, eles são os seguintes:

- **app.js** - É a parte principal do programa, que invoca todos os outros componentes e controla o início da simulação.
- **probFunctions.js** - Arquivo com a implementação das funções de probabilidades utilizadas no programa
- **eventList.js** - Arquivo que implementa o objeto que manipula a lista de eventos do simulador.
- **simulation.js** - Arquivo que contem a Simulação em si. Nele estão todas as variáveis necessárias para a simulação, implementação das entidades, chamada e parada do loop de eventos entre outros.
- **view.js** - Responsável pela manipulação da parte gráfica ou DOM do navegador. Toda questão visual passa por ele.
- **config.js** - Responsável por portar algumas configurações básicas como elementos utilizados na View e configurações iniciais de simulação como tempo mínimo de simulação.

Configurações finais:

Caso queira fazer modificações no programa e compila-lo, é necessário ter **Nodejs** instalado na maquina e rodar os seguintes comandos

1. `npm install && npm install -g browserify`
2. To play: run **\*\*index.js\*\*** or `npm start`.
3. To compile changes run: `npm run build`