Manual Técnico e Acadêmico de Uso dos Scripts Bônus

---

1. Introdução

Os scripts bônus apresentados são projetados para integrar sistemas interativos e realizar análises em tempo real, com destaque para monitoramento, visualização e automação de tarefas. Este manual oferece um guia técnico detalhado para cada arquivo, suas funcionalidades, como utilizá-los e expandi-los academicamente.

Os sistemas utilizam:

- **Backend (Flask):** Processamento de dados e comunicação com o frontend via Socket.IO.
- **Frontend (HTML e JavaScript):** Visualizações dinâmicas e controle de execução.
- **Scripts de automação (JavaScript):** Monitoramento de multiplicadores e automação de cliques.

---

2. Estrutura Geral dos Arquivos

2.1. Arquivo: `ctsv1.py`

- **Descrição:**
  Backend Flask que realiza o processamento principal, incluindo cálculo de multiplicadores, RSI, médias móveis, e comunicação em tempo real com o frontend.
- **Funcionalidades principais:**
  - Geração de dados em tempo real usando seeds fixas.
  - Cálculo e transmissão de métricas (RSI, médias móveis, maior sequência de perdas/ganhos).
  - Controle remoto via comandos enviados do frontend.
- **Tecnologias utilizadas:**
  - Flask (backend web)
  - Flask-SocketIO (comunicação em tempo real)
  - Pandas (cálculos estatísticos)
  - Threading (simulação em paralelo)

Configurações personalizáveis:

- **`target`:** Multiplicador-alvo.
- **`speed`:** Velocidade de execução dos dados (1.0x padrão).
- **`janela_media`:** Tamanho da janela para cálculo da média móvel.
- **`janela_rsi`:** Tamanho da janela para cálculo do RSI.

Como executar:

1. Instale as dependências:

   ```
   bash
   CopiarEditar
   ```

```
pip install flask flask-socketio flask-cors pandas
```

2. Execute o script:

```
bash
CopiarEditar
python ctsv1.py
```

3. Acesse o frontend na URL fornecida no terminal (por padrão: `http://127.0.0.1:5000`).

2.2. Arquivo: `display.html`

- **Descrição:**
  Um frontend básico que exibe dados transmitidos pelo backend. Permite interagir com o sistema por meio de botões (pausar, retomar, etc.).
- **Funcionalidades principais:**
  - Exibição de métricas como multiplicador, percentual, RSI e maior sequência de perdas/ganhos.
  - Controles interativos para alterar o estado do simulador (pausar, acelerar, etc.).

Configurações personalizáveis:

- Os controles são fixos no layout, mas podem ser modificados diretamente no arquivo, como ajustar os botões ou alterar os limites de interação.

Como utilizar:

1. Abra o arquivo `display.html` no navegador enquanto o backend está rodando.
2. Interaja com o sistema usando os botões disponíveis:
   - Pausar/Retomar: Controla o fluxo de dados.
   - Acelerar/Desacelerar: Ajusta a velocidade da simulação.

2.3. Arquivo: `index.html`

- **Descrição:**
  Um frontend avançado que inclui gráficos dinâmicos para exibição dos dados em tempo real, como gráficos de linha e RSI.
- **Funcionalidades principais:**
  - Gráfico de linha para percentuais e médias móveis.
  - Gráfico de RSI com demarcação de zonas (sobrecompra, sobrevenda).
  - Customização visual (cores e espessuras das linhas).

Configurações personalizáveis:

- Customizações gráficas:
  - Cor e espessura das linhas (gráfico de linha e RSI).
  - Zonas de sobrecompra e sobrevenda no RSI.
- As configurações podem ser ajustadas na seção de controles ou diretamente no código.

Como utilizar:

1. Abra o arquivo `index.html` no navegador com o backend rodando.
2. Utilize os controles para customizar os gráficos.
3. Visualize as atualizações em tempo real no gráfico.

2.4. Arquivo: `l.js`

- **Descrição:**
  Um script de monitoramento que detecta mudanças em elementos DOM de terceiros, calculando perdas consecutivas e parando automaticamente apostas em caso de sequência negativa.
- **Funcionalidades principais:**
  - Monitoramento contínuo dos valores exibidos.
  - Detecção de perdas consecutivas abaixo de um valor-alvo.
  - Automação para pausar apostas quando necessário.

Configurações personalizáveis:

- **`target`:** Multiplicador-alvo para detectar perdas.
- **`perdasConsecutivas`:** Número de perdas consecutivas para disparar a parada automática.

Como utilizar:

1. Insira o script em uma página que contenha o DOM compatível.
2. Execute o script diretamente no console do navegador ou injete em extensões como Tampermonkey.
3. O monitor irá começar automaticamente e logará as ações no console.

2.5. Arquivo: `laV2.js`

- **Descrição:**
  Uma versão expandida do monitor de multiplicadores, com recursos avançados de análise por blocos e registro detalhado.
- **Funcionalidades principais:**
  - Divisão em blocos de 300 multiplicadores para análise granular.
  - Registro de maiores sequências de perdas por bloco.
  - Reset automático após completar um número configurável de blocos.

Configurações personalizáveis:

- **`target`:** Multiplicador-alvo.
- **`perdasConsecutivas`:** Número de perdas consecutivas para registro.
- **`maxBlocos`:** Número máximo de blocos antes do reset automático.

Como utilizar:

1. Insira o script na página compatível, como em `l.js`.

2. O monitor irá logar no console os detalhes sobre cada bloco e as maiores sequências de perdas.
3. Utilize comandos no console para expandir os registros ou multiplicadores:
    - `monitor.toggleRegistros()`: Mostra ou oculta os registros de perdas.
    - `monitor.toggleMultiplicadores()`: Mostra ou oculta os multiplicadores capturados.

---

3. Aplicações Acadêmicas

Esses scripts têm grande potencial para uso acadêmico em áreas como:

1. **Teoria dos Jogos e Probabilidade:**
    - Estudo de padrões em sistemas determinísticos e suas distribuições.
2. **Visualização de Dados:**
    - Gráficos interativos para explorar séries temporais e indicadores financeiros.
3. **Automação e Detecção:**
    - Desenvolvimento de sistemas de monitoramento baseados em eventos.