

# Manual do Programador - Sentra Partners API

---

## Documentação Técnica da API MT4/MT5

---

**Versão:** 3.0

**Data:** Outubro 2025

**Base URL:** `https://sentrapartners.com/api/mt`

---

## Índice

---

1. [Visão Geral](#)
  2. [Autenticação](#)
  3. [Endpoints](#)
  4. [Modelos de Dados](#)
  5. [Códigos de Erro](#)
  6. [Exemplos de Integração](#)
  7. [Boas Práticas](#)
  8. [Changelog](#)
- 

## Visão Geral

---

A API Sentra Partners permite sincronizar contas MT4/MT5 com a plataforma web através de endpoints REST simples.

## Características

- **Protocolo:** HTTPS (TLS 1.2+)
- **Formato:** JSON
- **Método:** POST
- **Autenticação:** Email do usuário
- **Rate Limit:** 1 request/segundo por conta
- **Timeout:** 10 segundos

## Arquitetura

MT4/MT5 EA → HTTPS POST → API Gateway → Backend → MySQL Database

## Autenticação

A autenticação é feita através do campo `user_email` no body da requisição.

## Requisitos

- Email deve estar cadastrado na plataforma
- Conta MT4/MT5 será criada automaticamente no primeiro heartbeat
- Não há necessidade de tokens ou API keys

## Exemplo

```
{  
  "user_email": "usuario@example.com",  
  "account_number": "12345678"  
}
```

# Endpoints

---

## 1. POST /heartbeat

Sincroniza dados da conta (saldo, equity, margem).

### Request

**URL:** `https://sentrapartners.com/api/mt/heartbeat`

### Headers:

```
Content-Type: application/json
```

### Body:

```
{
  "user_email": "usuario@example.com",
  "account_number": "12345678",
  "broker": "XM Global Limited",
  "server": "XM-Real 50",
  "account_name": "João Silva",
  "balance": 10000.50,
  "equity": 10250.75,
  "currency": "USD",
  "leverage": 500,
  "margin_free": 9500.25,
  "open_positions": 3,
  "platform": "MT4",
  "account_type": "CENT"
}
```

### Campos:

Campo	Tipo	Obrigatório	Descrição
user_email	string	✓	Email cadastrado na plataforma
account_number	string	✓	Número da conta MT4/MT5
broker	string	✗	Nome do broker
server	string	✗	Nome do servidor
account_name	string	✗	Nome do titular da conta
balance	number	✓	Saldo da conta
equity	number	✓	Equity (patrimônio)
currency	string	✗	Moeda da conta (USD, EUR, etc)
leverage	number	✗	Alavancagem
margin_free	number	✗	Margem livre
open_positions	number	✗	Número de posições abertas
platform	string	✗	MT4 ou MT5
account_type	string	✗	CENT ou STANDARD

## Response

### Success (200):

```
{
  "success": true,
  "message": "Heartbeat recebido"
}
```

### Error (400):

```
{
  "success": false,
  "error": "Parâmetros obrigatórios: user_email, account_number"
}
```

### Error (404):

```
{
  "success": false,
  "error": "Usuário não encontrado. Cadastre-se na plataforma primeiro."
}
```

---

## 2. POST /positions

Sincroniza posições abertas (trades flutuantes).

### Request

**URL:** `https://sentrapartners.com/api/mt/positions`

### Headers:

Content-Type: application/json

### Body:

```
{
  "user_email": "usuario@example.com",
  "account_number": "12345678",
  "positions": [
    {
      "ticket": "123456789",
      "symbol": "EURUSD",
      "type": "buy",
      "volume": 0.10,
      "open_price": 1.08500,
      "current_price": 1.08650,
      "profit": 15.00,
      "swap": -0.50,
      "commission": -1.00,
      "open_time": "2025-10-29 10:30:00"
    }
  ]
}
```

**Campos do Array `positions`:**

Campo	Tipo	Obrigatório	Descrição
ticket	string	✓	Número do ticket
symbol	string	✓	Par de moedas
type	string	✓	"buy" ou "sell"
volume	number	✓	Volume em lotes
open_price	number	✓	Preço de abertura
current_price	number	✓	Preço atual
profit	number	✓	Lucro/prejuízo atual
swap	number	✗	Swap acumulado
commission	number	✗	Comissão
open_time	string	✓	Data/hora de abertura (ISO 8601)

## Response

### Success (200):

```
{
  "success": true,
  "message": "3 posições sincronizadas"
}
```

## 3. POST /trades

Sincroniza histórico de trades fechados.

### Request

**URL:** `https://sentrapartners.com/api/mt/trades`

### Headers:

Content-Type: application/json

Body:

```
{
  "user_email": "usuario@example.com",
  "account_number": "12345678",
  "trades": [
    {
      "ticket": "987654321",
      "symbol": "GBPUSD",
      "type": "sell",
      "volume": 0.05,
      "open_price": 1.25000,
      "close_price": 1.24800,
      "open_time": "2025-10-28 14:00:00",
      "close_time": "2025-10-28 16:30:00",
      "profit": 10.00,
      "commission": -0.50,
      "swap": -0.25
    }
  ]
}
```

Campos do Array trades :

Campo	Tipo	Obrigatório	Descrição
ticket	string	✓	Número do ticket
symbol	string	✓	Par de moedas
type	string	✓	"buy" ou "sell"
volume	number	✓	Volume em lotes
open_price	number	✓	Preço de abertura
close_price	number	✓	Preço de fechamento
open_time	string	✓	Data/hora de abertura
close_time	string	✓	Data/hora de fechamento
profit	number	✓	Lucro/prejuízo final
commission	number	✗	Comissão
swap	number	✗	Swap acumulado

## Response

### Success (200):

```
{
  "success": true,
  "message": "150 trades sincronizados"
}
```



## Modelos de Dados

### TradingAccount

Representa uma conta MT4/MT5.

```
interface TradingAccount {
  id: number;
  userId: number;
  terminalId: string;
  accountNumber: string;
  broker: string;
  server: string;
  platform: "MT4" | "MT5";
  accountType: "DEMO" | "REAL" | "CENT";
  isCentAccount: boolean;
  balance: bigint; // Armazenado como inteiro (x100 ou x10000)
  equity: bigint;
  marginFree: bigint;
  marginUsed: bigint;
  marginLevel: number;
  leverage: number;
  openPositions: number;
  currency: string;
  status: "connected" | "disconnected";
  lastHeartbeat: Date;
  createdAt: Date;
  updatedAt: Date;
}
```

### Trade

Representa um trade (aberto ou fechado).



```

interface Trade {
  id: number;
  accountId: number;
  userId: number;
  ticket: string;
  symbol: string;
  type: "buy" | "sell";
  volume: number; // Armazenado como inteiro (x100)
  openPrice: number; // Armazenado como inteiro (x100000)
  openTime: Date;
  closePrice: number | null;
  closeTime: Date | null;
  stopLoss: number | null;
  takeProfit: number | null;
  profit: bigint; // Armazenado como inteiro (x100 ou x10000)
  commission: bigint;
  swap: bigint;
  comment: string;
  magicNumber: number;
  createdAt: Date;
  updatedAt: Date;
}

```

## BalanceHistory

Representa um snapshot de saldo.

```

interface BalanceHistory {
  id: number;
  accountId: number;
  userId: number;
  balance: bigint;
  equity: bigint;
  timestamp: Date;
}

```

## 1234 Conversão de Valores

### Contas CENT vs STANDARD

A API detecta automaticamente o tipo de conta e aplica o divisor correto:

Tipo	Divisor	Exemplo
CENT	10.000	\$100,00 → 1.000.000
STANDARD	100	\$100,00 → 10.000

## Detecção Automática

A API detecta contas CENT através de:

1. Campo `account_type` explícito
2. Nome do servidor contendo "cent" ou "micro"
3. Padrões conhecidos de brokers

## Armazenamento

Todos os valores monetários são armazenados como **BIGINT** no banco de dados:

```
balance BIGINT NOT NULL DEFAULT 0
```

Isso suporta valores até **9.223.372.036.854.775.807** (9 quintilhões).

---

## Códigos de Erro

---

### HTTP Status Codes

Código	Significado	Descrição
200	OK	Requisição bem-sucedida
400	Bad Request	Parâmetros inválidos ou faltando
404	Not Found	Usuário ou conta não encontrada
500	Internal Server Error	Erro no servidor
502	Bad Gateway	Servidor temporariamente indisponível

### Mensagens de Erro Comuns

```
{
  "success": false,
  "error": "Parâmetros obrigatórios: user_email, account_number"
}
```

```
{
  "success": false,
  "error": "Usuário não encontrado. Cadastre-se na plataforma primeiro."
}
```

```
{
  "success": false,
  "error": "Conta não encontrada. Envie heartbeat primeiro."
}
```



## Exemplos de Integração

### MQL4 (MetaTrader 4)

```
//+-----+
//| Enviar Heartbeat |
//+-----+
void SendHeartbeat() {
    string url = "https://sentrapartners.com/api/mt/heartbeat";
    string headers = "Content-Type: application/json\r\n";

    string data = "{";
    data += "\"user_email\": \"usuario@example.com\", ";
    data += "\"account_number\": \"\" + IntegerToString(AccountNumber()) + "\", ";
    data += "\"broker\": \"\" + AccountCompany() + "\", ";
    data += "\"balance\": \"\" + DoubleToStr(AccountBalance(), 2) + "\", ";
    data += "\"equity\": \"\" + DoubleToStr(AccountEquity(), 2) + "\", ";
    data += "\"platform\": \"MT4\", ";
    data += "\"account_type\": \"CENT\"";
    data += "}";

    char post[], result[];
    ArrayResize(post, StringToCharArray(data, post, 0, WHOLE_ARRAY) - 1);

    int res = WebRequest("POST", url, headers, 10000, post, result, headers);

    if(res == 200) {
        Print("✓ Heartbeat enviado com sucesso");
    } else {
        Print("x Erro HTTP ", res);
    }
}
```

## MT5 (MetaTrader 5)

```
//+-----+
//| Enviar Posições Abertas |
//+-----+
void SendPositions() {
    string url = "https://sentrapartners.com/api/mt/positions";
    string headers = "Content-Type: application/json\r\n";

    string data = "{";
    data += "\"user_email\":\"usuario@example.com\",";
    data += "\"account_number\":\"" +
IntegerToString(AccountInfoInteger(ACCOUNT_LOGIN)) + "\",";
    data += "\"positions\":[";

    int total = PositionsTotal();
    for(int i = 0; i < total; i++) {
        ulong ticket = PositionGetTicket(i);
        if(ticket > 0) {
            if(i > 0) data += ",";

            data += "{";
            data += "\"ticket\":\"" + IntegerToString(ticket) + "\",";
            data += "\"symbol\":\"" + PositionGetString(POSITION_SYMBOL) +
"\",";
            data += "\"type\":\"" + (PositionGetInteger(POSITION_TYPE) ==
POSITION_TYPE_BUY ? "buy" : "sell") + "\",";
            data += "\"volume\":\"" +
DoubleToString(PositionGetDouble(POSITION_VOLUME), 2) + "\",";
            data += "\"profit\":\"" +
DoubleToString(PositionGetDouble(POSITION_PROFIT), 2);
            data += "}";
        }
    }

    data += "]]";

    char post[], result[];
    ArrayResize(post, StringToCharArray(data, post, 0, WHOLE_ARRAY) - 1);

    int res = WebRequest("POST", url, headers, 10000, post, result, headers);

    if(res == 200) {
        Print("✓ Posições enviadas com sucesso");
    }
}
```

## Python

```
import requests
import json

def send_heartbeat(email, account_number, balance, equity):
    url = "https://sentrapartners.com/api/mt/heartbeat"

    data = {
        "user_email": email,
        "account_number": str(account_number),
        "broker": "XM Global Limited",
        "balance": balance,
        "equity": equity,
        "platform": "MT4",
        "account_type": "CENT"
    }

    headers = {
        "Content-Type": "application/json"
    }

    response = requests.post(url, json=data, headers=headers, timeout=10)

    if response.status_code == 200:
        print("✓ Heartbeat enviado com sucesso")
        return response.json()
    else:
        print(f"x Erro HTTP {response.status_code}")
        return None

# Uso
send_heartbeat("usuario@example.com", 12345678, 10000.50, 10250.75)
```

## cURL

```
curl -X POST https://sentrapartners.com/api/mt/heartbeat \
-H "Content-Type: application/json" \
-d '{
  "user_email": "usuario@example.com",
  "account_number": "12345678",
  "broker": "XM Global Limited",
  "balance": 10000.50,
  "equity": 10250.75,
  "platform": "MT4",
  "account_type": "CENT"
}'
```



## Boas Práticas

### 1. Rate Limiting

- **Máximo:** 1 request/segundo por conta

- **Heartbeat:** Recomendado 60 segundos
- **Posições:** Apenas quando houver mudanças
- **Histórico:** Enviar em lotes de 100 trades

## 2. Timeout

- Configure timeout de **10 segundos**
- Implemente retry com backoff exponencial
- Máximo de **3 tentativas**

## 3. Logs

- Registre todas as requisições e respostas
- Inclua timestamps
- Não registre dados sensíveis

## 4. Tratamento de Erros

```
int retries = 0;
int maxRetries = 3;
int backoff = 1000; // 1 segundo

while(retries < maxRetries) {
    int res = WebRequest(...);

    if(res == 200) {
        break; // Sucesso
    } else if(res == 502 || res == 503) {
        // Servidor temporariamente indisponível
        Sleep(backoff);
        backoff *= 2; // Backoff exponencial
        retries++;
    } else {
        // Erro permanente
        Print("Erro permanente: ", res);
        break;
    }
}
```

## 5. Segurança

- Sempre use HTTPS
- Não armazene senhas no EA

- Valide todos os inputs
- Use prepared statements no backend

## 6. Performance

- Envie histórico em lotes de 100
  - Comprima JSON se > 1MB
  - Use conexões keep-alive
  - Cache dados quando possível
- 

## Changelog

---

### v3.0 (Outubro 2025)

**Adicionado:** - Suporte a `account_type` (CENT/STANDARD) - Colunas BIGINT para suportar valores grandes - Auto-geração de `terminalId` - Detecção automática de contas CENT

**Corrigido:** - Erro "Out of range value" em contas CENT - Recriação de contas excluídas - Valores incorretos em contas CENT

**Alterado:** - `balance`, `equity`, `profit` agora são BIGINT - API aceita JSON e form-urlencoded

### v2.1 (Setembro 2025)

**Adicionado:** - Endpoint `/positions` para posições abertas - Endpoint `/trades` para histórico - Suporte a arrays de trades

**Corrigido:** - Timeout em históricos grandes - Duplicação de trades

### v2.0 (Agosto 2025)

**Adicionado:** - Suporte a MT5 - Sistema multi-usuário - Histórico de saldo

**Alterado:** - Migração de REST para tRPC (backend)

## v1.0 (Julho 2025)





**Inicial:** - Endpoint `/heartbeat` - Suporte a MT4 - Sincronização básica

---

## Contato

---

### Suporte Técnico

-  **Email:** [dev@sentrapartners.com](mailto:dev@sentrapartners.com)
-  **Docs:** <https://docs.sentrapartners.com>
-  **Discord:** <https://discord.gg/sentrapartners>
-  **Issues:** <https://github.com/sentrapartners/api/issues>

### Contribuir

Contribuições são bem-vindas! Veja nosso [guia de contribuição](#).

---

© 2025 Sentra Partners - Todos os direitos reservados

**Versão da API:** 3.0

**Última Atualização:** Outubro 2025