Trade Assistant V2.008 - Funktionsdokumentation (detailliert)

Stand: 30.12.2025

Ziel: Diese PDF erklaert jede Funktion aus den Projektdateien in einem nachvollziehbaren, anfaengerfreundlichen Format. Die Erklaerungen sind aus dem Quelltext abgeleitet und enthalten zudem Code-Referenzen (Datei/Zeile), damit man direkt im Editor mitlesen kann.

Dateien im Scope

Datei	Rolle im Projekt	Encoding
Trade Assistent V2.008.mq5	Haupteinstieg (EA), Lifecycle (OnInit/OnTick/OnDeinit), Koordination von UI/DB/Discord	utf-16
db_state.mqh	Persistenz: DB/Datei-Speicher fuer Trades, Positionen, Linien, Meta-Daten	utf-16
discord.mqh	Discord-Integration: Message-Formatierung, Webhook-URL Handling	utf-8-sig
discord_send.mqh	Low-Level Versand (DiscordSend) via WebRequest	utf-8-sig
event.mqh	Eventhandling (OnChartEvent), Reaktion auf Button-Klicks und Linienbewegung	utf-8-sig
gui_elemente.mqh	UI-Helfer: Linien/Buttons/Labels erzeugen, Layout-Funktionen, Update der Texte	utf-8-sig
trades_panel.mqh	Linkes Trades-Panel (Long/Short Bereiche, Row-Buttons, Rebuild/Show/Hide)	utf-8-sig
ui_registry.mqh	Zentrales Registry fuer alle erzeugten Chart-Objekte (sauberes Loeschen/Adoptieren)	utf-8-sig

Funktionsreferenz

Aufbau pro Funktion: Zweck/Kontext, Parameter, Rueckgabe, Seiteneffekte, Ablauf (mit Abschnittslogik), Kontrollfluss, wichtige Codezeilen, Aufrufe/Verwendungen.

Trade Assistent V2.008.mq5

Haupteinstieg (EA), Lifecycle (OnInit/OnTick/OnDeinit), Koordination von UI/DB/Discord

Funktionen in dieser Datei (36): OnInit, DB_SaveTradeLines, DB_RestoreAll, OnTick, PosSuf, StatusIsClosed, StatusIsActive, SetPosLinesSolid, DeletePosLines, FormatCloseSL_DB, ClearActiveTrend, DB_LoadPrice, setzeTrade, UpdateSabioTP, TPSLReached, OnDeinit, DB_PersistAll, UI_ParseSLHitName, Get_Price_d, Get_Price_s, calcLots, ParsePosNoFromName, ParseTradeLineName, DB_LoadPositionOne, DB_RestoreTradeLines_All, RollbackDraftRow, UI_SyncDirectionFromBaseLines, UI_SetIntEdit, UI_UpdateNextTradePosUI, UI_SetLineMeta, UI_ParseLineMeta, createEntryAndSLLinien, createHLine, UI_IsTradePosLine, UI_CreateOrUpdateLineTag, UI_UpdateAllLineTags

OnInit [EA-Lifecycle / Eventhandler]

```
int OnInit()
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 211 bis 259

Kommentar direkt darueber: ================== ENDE LINE PRICES ==================

Rueckgabe	Parameter (Typ name [default])
int	(keine)

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).
- Liest/schreibt Persistenz (DB/Dateien).
- Kann Discord-Nachrichten versenden oder formatieren.

Ablauf (Schritt fuer Schritt)

1. 1. Start: Persistenz (DB/Datei) lesen/schreiben; Discord-Webhook Nachricht senden/formatieren; Logging/Benachrichtigung. Wichtige Aufrufe: Get_Price_s, DB_RestoreAll, DB_PrintData, OnInit, UI_TradesPanel_Create, DB_Init, checkDiscord, update_Text.
2. 2. Trade-Linien (Pos1..Pos4) aus positions erzeugen: UI/Chart-Objekte anlegen/aktualisieren/verschieben; Persistenz (DB/Datei) lesen/schreiben; Logging/Benachrichtigung. Wichtige Aufrufe: DB_RestoreTradeLines_All, DB_PrintData, Get_Price_d, update_Text.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!checkDiscord())
- if(!DB_Init())

Wichtige Codezeilen (zum Mitlesen im Editor)

```
214: Print("start OnInit()");
216: // Init und Test Discord Api
218: if(!checkDiscord())
219: return -1;
227: if(!DB_Init())
228: return (INIT_FAILED);
229: DB_PrintData();
234: bool restored = DB_RestoreAll(); // liest Meta + Positionen aus SQLite
235: Print("DB restored: ", restored);
```

```
246: //   DB_RestoreTradeLines_All();
247: //   DB_PrintData();
257: ChartRedraw(0);
258: return (INIT_SUCCEEDED);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_Init, DB_PrintData, DB_RestoreAll, DB_RestoreTradeLines_All, Get_Price_d, Get_Price_s, UI_TradesPanel_Create, UI_TradesPanel_RebuildRows, checkDiscord, createEntryAndSLLinien, update_Text	(keine Treffer im Projekt)

Verwendete MQL-Konstanten (UI)

- Chart-Properties: CHART_FOREGROUND

DB_SaveTradeLines [Persistenz (DB/Datei)]

```
void DB_SaveTradeLines(const string suf)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 264 bis 279

Rueckgabe	Parameter (Typ name [default])
void	const string suf

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_Key, DB_SetMetaText.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, SL_Long + suf) >= 0)
- if(ObjectFind(0, Entry_Long + suf) >= 0)
- if(ObjectFind(0, SL_Short + suf) >= 0)
- if(ObjectFind(0, Entry_Short + suf) >= 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
268: if(ObjectFind(0, SL_Long + suf) >= 0)
269: DB_SetMetaText(DB_Key("L_sl"), DoubleToString(ObjectGetDouble(0, SL_Long + suf, OBJPROP_PRI
270: if(ObjectFind(0, Entry_Long + suf) >= 0)
271: DB_SetMetaText(DB_Key("L_entry"), DoubleToString(ObjectGetDouble(0, Entry_Long + suf, OBJPR
275: if(ObjectFind(0, SL_Short + suf) >= 0)
276: DB_SetMetaText(DB_Key("S_sl"), DoubleToString(ObjectGetDouble(0, SL_Short + suf, OBJPROP_PR
277: if(ObjectFind(0, Entry_Short + suf) >= 0)
278: DB_SetMetaText(DB_Key("S_entry"), DoubleToString(ObjectGetDouble(0, Entry_Short + suf, OBJP
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_Key, DB_SetMetaText	trades_panel.mqh:558 DB_SaveTradeLines(suf); trades_panel.mqh:567 DB_SaveTradeLines(suf);

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_PRICE

DB_RestoreAll [Persistenz (DB/Datei)]

```
bool DB_RestoreAll()
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 284 bis 332

Kommentar direkt darueber: | DB Restore (Meta + Positions) | | Minimal für Pyramiden-Logik (TradeNo/PosNo) |

Rueckgabe	Parameter (Typ name [default])
bool	(keine)

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_Key, DB_LoadPositions, DB_SetMetaInt, DB_GetMetaInt, UI_TradesPanel_RebuildRows.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int i = 0; i < n; i++)
- if(rows[i].was_sent != 1)
- if(rows[i].trade_no > max_any)
- if(StringFind(rows[i].status, "CLOSED") == 0)
- if(rows[i].direction == "LONG")
- if(rows[i].trade_no > max_open_long)
- if(rows[i].direction == "SHORT")
- if(rows[i].trade_no > max_open_short)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
287: DB_GetMetaInt(DB_Key("last_trade_no"), meta_last, 0);
288: DB_GetMetaInt(DB_Key("active_long_trade_no"), meta_long, 0);
289: DB_GetMetaInt(DB_Key("active_short_trade_no"), meta_short, 0);
295: DB_PositionRow rows[];
296: int n = DB_LoadPositions(_Symbol, _Period, rows);
298: for(int i = 0; i < n; i++)
300: if(rows[i].was_sent != 1)
304: if(rows[i].trade_no > max_any)
308: if(StringFind(rows[i].status, "CLOSED") == 0)
311: if(rows[i].direction == "LONG")
312: if(rows[i].trade_no > max_open_long)
315: if(rows[i].direction == "SHORT")
327: DB_SetMetaInt(DB_Key("last_trade_no"), last_trade_nummer);
328: DB_SetMetaInt(DB_Key("active_long_trade_no"), active_long_trade_no);
329: DB_SetMetaInt(DB_Key("active_short_trade_no"), active_short_trade_no);
331: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_GetMetaInt, DB_Key, DB_LoadPositions, DB_SetMetaInt, UI_TradesPanel_RebuildRows	Trade Assistent V2.008.mq5:234 bool restored = DB_RestoreAll(); // liest Meta + Positionen aus SQLite

OnTick [EA-Lifecycle / Eventhandler]

```
void OnTick()
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 337 bis 355

Kommentar direkt darueber: | Expert tick function |

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: TPSLReached, CreateLabelsLong, CreateLabelsShort.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(is_long_trade)
- if(is_sell_trade)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
338: {
339: CurrentAskPrice = SymbolInfoDouble(_Symbol, SYMBOL_ASK);
340: CurrentBidPrice = SymbolInfoDouble(_Symbol, SYMBOL_BID);
343: TPSLReached();
346: if(is_long_trade)
351: if(is_sell_trade)
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
CreateLabelsLong, CreateLabelsShort, TPSLReached	(keine Treffer im Projekt)

PosSuf [Hilfsfunktion / Logik]

```
string PosSuf(const int pos_no)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 365 bis 368

Rueckgabe	Parameter (Typ name [default])
string	const int pos_no

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
366: {
367: return "_" + IntegerToString(pos_no);
367: return "_" + IntegerToString(pos_no);
368: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	Trade Assistent V2.008.mq5:391 string suf = PosSuf(pos_no); Trade Assistent V2.008.mq5:417 string suf = PosSuf(pos_no);

StatusIsClosed [Hilfsfunktion / Logik]

```
bool StatusIsClosed(const string s)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 373 bis 376

Rueckgabe	Parameter (Typ name [default])

bool	const string s

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
374: {
375: return (StringFind(s, "CLOSED") == 0); // CLOSED / CLOSED_TP / CLOSED_SL
375: return (StringFind(s, "CLOSED") == 0); // CLOSED / CLOSED_TP / CLOSED_SL
376: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	Trade Assistent V2.008.mq5:636 if(StatusIsClosed(p.status))

StatusIsActive [Hilfsfunktion / Logik]

```
bool StatusIsActive(const string s)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 381 bis 384

Rueckgabe	Parameter (Typ name [default])
bool	const string s

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
382: {
383: return (s == "PENDING" || s == "OPEN");
383: return (s == "PENDING" || s == "OPEN");
384: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	Trade Assistent V2.008.mq5:709 if(StatusIsActive(rows[j].status)) Trade Assistent V2.008.mq5:715 if(StatusIsActive(rows[j].status))

SetPosLinesSolid [Hilfsfunktion / Logik]

```
void SetPosLinesSolid(const string direction, const int pos_no)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 389 bis 410

Rueckgabe	Parameter (Typ name [default])
void	const string direction const int pos_no

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: PosSuf.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(direction == "LONG")
- if(ObjectFind(0, Entry_Long + suf) >= 0)
- if(ObjectFind(0, SL_Long + suf) >= 0)
- if(direction == "SHORT")
- if(ObjectFind(0, Entry_Short + suf) >= 0)
- if(ObjectFind(0, SL_Short + suf) >= 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
393: if(direction == "LONG")
395: if(ObjectFind(0, Entry_Long + suf) >= 0)
396: ObjectSetInteger(0, Entry_Long + suf, OBJPROP_STYLE, STYLE_SOLID);
398: if(ObjectFind(0, SL_Long + suf) >= 0)
399: ObjectSetInteger(0, SL_Long + suf, OBJPROP_STYLE, STYLE_SOLID);
402: if(direction == "SHORT")
404: if(ObjectFind(0, Entry_Short + suf) >= 0)
405: ObjectSetInteger(0, Entry_Short + suf, OBJPROP_STYLE, STYLE_SOLID);
407: if(ObjectFind(0, SL_Short + suf) >= 0)
408: ObjectSetInteger(0, SL_Short + suf, OBJPROP_STYLE, STYLE_SOLID);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
PosSuf	Trade Assistent V2.008.mq5:655 SetPosLinesSolid(p.direction, p.pos_no);

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_STYLE

DeletePosLines [Hilfsfunktion / Logik]

```
void DeletePosLines(const string direction, const int pos_no)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 415 bis 432

Rueckgabe	Parameter (Typ name [default])
void	const string direction const int pos_no

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: PosSuf.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(direction == "LONG")
- if(direction == "SHORT")

Wichtige Codezeilen (zum Mitlesen im Editor)

```
419: if(direction == "LONG")
422: ObjectDelete(0, SL_Long + suf);
423: ObjectDelete(0, Entry_Long + suf);
426: if(direction == "SHORT")
```

```
429: ObjectDelete(0, SL_Short + suf);
430: ObjectDelete(0, Entry_Short + suf);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
PosSuf	Trade Assistent V2.008.mq5:674 DeletePosLines("LONG", p.pos_no); Trade Assistent V2.008.mq5:691 DeletePosLines("SHORT", p.pos_no);

FormatCloseSL_DB [Hilfsfunktion / Logik]

```
string FormatCloseSL_DB(const DB_PositionRow &p)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 437 bis 443

Rueckgabe	Parameter (Typ name [default])
string	const DB_PositionRow &p

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
438: {
439: string msg = "@everyone\n";
440: msg += StringFormat("**Note:** %s Trade %d | Pos %d - has been stopped out\n",
441: p.symbol, p.trade_no, p.pos_no);
442: return msg;
442: return msg;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	Trade Assistent V2.008.mq5:668 SendDiscordMessage(FormatCloseSL_DB(p)); Trade Assistent V2.008.mq5:685 SendDiscordMessage(FormatCloseSL_DB(p));

ClearActiveTrend [Hilfsfunktion / Logik]

```
void ClearActiveTrend(const string direction)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 450 bis 485

Rueckgabe	Parameter (Typ name [default])
void	const string direction

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_Key, DB_SetMetaInt.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(direction == "LONG")
- if(ObjectFind(0, TP_BTN_ACTIVE_LONG) != -1)

- if(direction == "SHORT")
- if(ObjectFind(0, TP_BTN_ACTIVE_SHORT) != -1)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
452: if(direction == "LONG")
458: DB_SetMetaInt(DB_Key("active_long_trade_no"), 0);
460: if(ObjectFind(0, TP_BTN_ACTIVE_LONG) != -1)
462: ObjectSetInteger(0, TP_BTN_ACTIVE_LONG, OBJPROP_COLOR, clrBlack);
463: ObjectSetInteger(0, TP_BTN_ACTIVE_LONG, OBJPROP_BGCOLOR, clrBlack);
464: ObjectSetInteger(0, TP_BTN_ACTIVE_LONG, OBJPROP_BORDER_COLOR, clrBlack);
465: ObjectSetString(0, TP_BTN_ACTIVE_LONG, OBJPROP_TEXT, "");
469: if(direction == "SHORT")
475: DB_SetMetaInt(DB_Key("active_short_trade_no"), 0);
477: if(ObjectFind(0, TP_BTN_ACTIVE_SHORT) != -1)
479: ObjectSetInteger(0, TP_BTN_ACTIVE_SHORT, OBJPROP_COLOR, clrBlack);
480: ObjectSetInteger(0, TP_BTN_ACTIVE_SHORT, OBJPROP_BGCOLOR, clrBlack);
481: ObjectSetInteger(0, TP_BTN_ACTIVE_SHORT, OBJPROP_BORDER_COLOR, clrBlack);
482: ObjectSetString(0, TP_BTN_ACTIVE_SHORT, OBJPROP_TEXT, "");
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_Key, DB_SetMetaInt	Trade Assistent V2.008.mq5:721 ClearActiveTrend("LONG"); Trade Assistent V2.008.mq5:724 ClearActiveTrend("SHORT");

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_BGCOLOR, OBJPROP_BORDER_COLOR, OBJPROP_COLOR, OBJPROP_TEXT

DB_LoadPrice [Persistenz (DB/Datei)]

```
double DB_LoadPrice(const string key, const double fallback)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 491 bis 504

Rueckgabe	Parameter (Typ name [default])
double	const string key const double fallback

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_Key, DB_GetMetaText.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_GetMetaText(DB_Key(key), s, ""))
- if(StringLen(s) == 0)
- if(p <= 0.0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
494: if(!DB_GetMetaText(DB_Key(key), s, ""))
495: return fallback;
496: if(StringLen(s) == 0)
497: return fallback;
500: if(p <= 0.0)
501: return fallback;
503: return p;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_GetMetaText, DB_Key	(keine Treffer im Projekt)

setzeTrade [Hilfsfunktion / Logik]

```
void setzeTrade()
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 512 bis 561

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Kann echte Broker-Orders/Positionen anstossen (Send & Trade).

Ablauf (Schritt fuer Schritt)

1 1. Start: Broker-Trade/Order (echte Ausfuehrung); Logging/Benachrichtigung. Wichtige Aufrufe: calcLots.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!SendOnlyButton)
- if(ui_direction_is_long)
- if(buy_ok)
- if(result == IDOK)
- if(sell_ok)
- if(result == IDOK)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
517: if(!SendOnlyButton)
519: if(ui_direction_is_long)
523: ? trade.BuyStop(lots, Entry_Price, _Symbol, SL_Price, 0.0, ORDER_TIME_GTC)
524: : trade.BuyStop(lots, Entry_Price, _Symbol, SL_Price, TP_Price, ORDER_TIME_GTC);
526: if(buy_ok)
528: ulong order_ticket = trade.ResultOrder();
534: Print(__FUNCTION__,": BuyStop failed. Retcode=", trade.ResultRetcode());
535: int result = MessageBox("BuyStop konnte NICHT platziert werden. TradeNummer bleibt unveränd
536: if(result == IDOK);
543: ? trade.SellStop(lots, Entry_Price, _Symbol, SL_Price, 0.0, ORDER_TIME_GTC)
544: : trade.SellStop(lots, Entry_Price, _Symbol, SL_Price, TP_Price, ORDER_TIME_GTC);
546: if(sell_ok)
548: ulong sell_ticket = trade.ResultOrder();
554: Print(__FUNCTION__,": SellStop failed. Retcode=", trade.ResultRetcode());
555: int result = MessageBox("SellStop konnte NICHT platziert werden. TradeNummer bleibt unverän
556: if(result == IDOK);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
calcLots	(keine Treffer im Projekt)

UpdateSabioTP [Hilfsfunktion / Logik]

```
void UpdateSabioTP()
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 566 bis 588

Kommentar direkt darueber: | Sabio TP berechnen |

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(Entry_Price > CurrentAskPrice)
- if(Entry_Price < CurrentBidPrice)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
567: {
568: if(Entry_Price > CurrentAskPrice)
568: if(Entry_Price > CurrentAskPrice)
569: {
570: string EntryPriceString = ObjectGetString(0, SabioEntry, OBJPROP_TEXT, 0);
578: if(Entry_Price < CurrentBidPrice)
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	event.mqh:412 UpdateSabioTP();

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_TEXT

TPSLReached [Hilfsfunktion / Logik]

```
void TPSLReached()
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 602 bis 725

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).
- Kann Discord-Nachrichten versenden oder formatieren.

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LoadPositions.

2 2. 1) Alle aktiven Positionen prüfen/aktualisieren: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: StatusIsClosed.

3 3. Entry-Hit -> PENDING -> OPEN: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: SetPosLinesSolid, DB_UpsertPosition.

4 4. SL/TP nur wenn OPEN: Persistenz (DB/Datei) lesen/schreiben; Discord-Webhook Nachricht senden/formatieren; Logging/Benachrichtigung. Wichtige Aufrufe: SendDiscordMessage, DeletePosLines, DB_UpsertPosition, FormatCloseSL_DB.

5 5. 2) Prüfen ob Richtung komplett fertig ist -> active_* löschen: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: ClearActiveTrend, StatusIsActive.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(n <= 0)
- for(int i = 0; i < n; i++)
- if(p.was_sent != 1)
- if(p.direction == "LONG")
- if(active_long_trade_no <= 0 || p.trade_no != active_long_trade_no)
- if(p.direction == "SHORT")
- if(active_short_trade_no <= 0 || p.trade_no != active_short_trade_no)
- if(StatusIsClosed(p.status))
- if(p.status == "PENDING")
- if(p.direction == "LONG")
- if(entry_hit)
- if(p.status != "OPEN")
- if(p.direction == "LONG")
- if(p.sl > 0.0 && (CurrentBidPrice <= p.sl))
- if(p.direction == "SHORT")

Wichtige Codezeilen (zum Mitlesen im Editor)

```
604: // Positions aus DB laden
605: DB_PositionRow rows[];
606: int n = DB_LoadPositions(_Symbol, _Period, rows);
607: if(n <= 0)
608: return;
610: // --- 1) Alle aktiven Positionen prüfen/aktualisieren
611: for(int i = 0; i < n; i++)
613: DB_PositionRow p = rows[i];
615: if(p.was_sent != 1)
652: DB_UpsertPosition(p);
668: SendDiscordMessage(FormatCloseSL_DB(p));
672: DB_UpsertPosition(p);
675: Alert(_Symbol + " LONG Trade " + IntegerToString(p.trade_no) + " Pos" + IntegerToString(p.p
685: SendDiscordMessage(FormatCloseSL_DB(p));
689: DB_UpsertPosition(p);
692: Alert(_Symbol + " SHORT Trade " + IntegerToString(p.trade_no) + " Pos" + IntegerToString(p.
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
ClearActiveTrend, DB_LoadPositions, DB_UpsertPosition, DeletePosLines, FormatCloseSL_DB, SendDiscordMessage, SetPosLinesSolid, StatusIsActive, StatusIsClosed	Trade Assistent V2.008.mq5:343 TPSLReached();

OnDeinit [EA-Lifecycle / Eventhandler]

```
void OnDeinit(const int reason)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 730 bis 740

Kommentar direkt darueber: | Expert deinitialization function |

Rueckgabe	Parameter (Typ name [default])
void	const int reason

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben; Logging/Benachrichtigung. Wichtige Aufrufe: UI_TradesPanel_Destroy, UI_Reg_DeleteAll, DB_Close.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
731: {
734: int deleted = UI_Reg_DeleteAll();
735: Print("UI Registry cleanup deleted objects: ", deleted);
735: Print("UI Registry cleanup deleted objects: ", deleted);
738: DB_Close();
738: DB_Close();
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_Close, UI_Reg_DeleteAll, UI_TradesPanel_Destroy	(keine Treffer im Projekt)

DB_PersistAll [Persistenz (DB/Datei)]

```
void DB_PersistAll()
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 746 bis 763

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_Key, DB_SetMetaInt, DB_SaveLinePrices.

2 2. UI-TradeNr aus dem Eingabefeld sichern: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_Key, DB_SetMetaInt.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, TRNB) >= 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
749: DB_SetMetaInt(DB_Key("last_trade_no"), last_trade_nummer);
750: DB_SetMetaInt(DB_Key("active_long_trade_no"), active_long_trade_no);
751: DB_SetMetaInt(DB_Key("active_short_trade_no"), active_short_trade_no);
754: DB_SaveLinePrices();
757: if(ObjectFind(0, TRNB) >= 0)
761: DB_SetMetaInt(DB_Key("trnb_ui"), trn);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_Key, DB_SaveLinePrices, DB_SetMetaInt	(keine Treffer im Projekt)

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_TEXT

UI_ParseSLHitName [UI / Chart-Objekte]

```
bool UI_ParseSLHitName(const string name, string &direction, int &trade_no, int &pos_no)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 774 bis 795

Kommentar direkt darueber: Name parsen

Rueckgabe	Parameter (Typ name [default])
bool	const string name string &direction int &trade_no int &pos_no

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung
2. 2. ["ButtonSLHit", "LONG|SHORT", "trade", "pos"]: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(StringFind(name, SLHIT_PREFIX, 0) != 0)
- if(n < 4)
- if(direction != "LONG" && direction != "SHORT")
- if(trade_no <= 0 || pos_no <= 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
776: if(StringFind(name, SLHIT_PREFIX, 0) != 0)
777: return false;
782: if(n < 4)
783: return false;
789: if(direction != "LONG" && direction != "SHORT")
790: return false;
791: if(trade_no <= 0 || pos_no <= 0)
792: return false;
794: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	(keine Treffer im Projekt)

Get_Price_d [Hilfsfunktion / Logik]

```
double Get_Price_d(string name)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 804 bis 808

Rueckgabe	Parameter (Typ name [default])
double	string name

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
805: {
806: //   return ObjectGetDouble(0, name, OBJPROP_PRICE);
807: return NormalizeDouble(ObjectGetDouble(0, name, OBJPROP_PRICE, 0), _Digits);
807: return NormalizeDouble(ObjectGetDouble(0, name, OBJPROP_PRICE, 0), _Digits);
808: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	Trade Assistent V2.008.mq5:249 Entry_Price = Get_Price_d(PR_HL);
	Trade Assistent V2.008.mq5:250 SL_Price = Get_Price_d(SL_HL);
	Trade Assistent V2.008.mq5:1050 double pr = Get_Price_d(PR_HL);
	Trade Assistent V2.008.mq5:1051 double sl = Get_Price_d(SL_HL);
	Trade Assistent V2.008.mq5:1206 update_Text(SLButton, "SL: " + DoubleToString(((Get_Price_d(PR_HL) - Get_Price_d(SL_HL)) / _Point), 0) + " Points | " + Get_Price_s(SL_HL));
	event.mqh:188 Entry_Price = Get_Price_d(PR_HL);
	event.mqh:189 SL_Price = Get_Price_d(SL_HL);
	event.mqh:266 update_Text(SLButton, "SL: " + DoubleToString(((Get_Price_d(PR_HL) - Get_Price_d(SL_HL)) / _Point), 0) + " Points | " + Get_Price_s(SL_HL));
	... (5 weitere)

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_PRICE

Get_Price_s [Hilfsfunktion / Logik]

```
string Get_Price_s(string name)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 816 bis 819

Rueckgabe	Parameter (Typ name [default])
string	string name

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
817: {
818: return DoubleToString(ObjectGetDouble(0, name, OBJPROP_PRICE), _Digits);
818: return DoubleToString(ObjectGetDouble(0, name, OBJPROP_PRICE), _Digits);
819: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	Trade Assistent V2.008.mq5:1205 update_Text(EntryButton, "Buy Stop @ " + Get_Price_s(PR_HL) + " | Lot: " + DoubleToString(NormalizeDouble(calcLots(SL_Price - Entry_Price), 2), 2));
	Trade Assistent V2.008.mq5:1206 update_Text(SLButton, "SL: " + DoubleToString(((Get_Price_d(PR_HL) - Get_Price_d(SL_HL)) / _Point), 0) + " Points | " + Get_Price_s(SL_HL));
	event.mqh:265 update_Text(EntryButton, "Buy Stop @ " + Get_Price_s(PR_HL) + " | Lot: " + DoubleToString(lots, 2));
	event.mqh:266 update_Text(SLButton, "SL: " + DoubleToString(((Get_Price_d(PR_HL) - Get_Price_d(SL_HL)) / _Point), 0) + " Points | " + Get_Price_s(SL_HL));
	event.mqh:270 update_Text(SabioEntry, "SABIO Entry: " + Get_Price_s(PR_HL));
	event.mqh:271 update_Text(SabioSL, "SABIO SL: " + Get_Price_s(SL_HL));
	event.mqh:287 update_Text(EntryButton, "Sell Stop @ " + Get_Price_s(PR_HL) + " | Lot: " + DoubleToString(lots, 2));
	event.mqh:288 update_Text(SLButton, "SL: " + DoubleToString(((Get_Price_d(SL_HL) - Get_Price_d(PR_HL)) / _Point), 0) + " Points | " + Get_Price_s(SL_HL));
	... (8 weitere)

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_PRICE

calcLots [Hilfsfunktion / Logik]

```
double calcLots(double slDistance)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 827 bis 850

Rueckgabe	Parameter (Typ name [default])
double	double slDistance

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Logging/Benachrichtigung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ticksize == 0 || tickvalue == 0 || lotstep == 0)
- if(moneyLotstep == 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
833: if(ticksize == 0 || tickvalue == 0 || lotstep == 0)
835: Print(__FUNCTION__, "> Lotsize cannot be calculated");
836: return 0;
841: if(moneyLotstep == 0)
843: Print(__FUNCTION__, "> Lotsize cannot be calculated");
844: return 0;
849: return lots;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	Trade Assistent V2.008.mq5:521 double lots = calcLots(Entry_Price - SL_Price);
	Trade Assistent V2.008.mq5:541 double lots = calcLots(SL_Price - Entry_Price);
	Trade Assistent V2.008.mq5:1205 update_Text(EntryButton, "Buy Stop @ " + Get_Price_s(PR_HL) + " | Lot: " + DoubleToString(NormalizeDouble(calcLots(SL_Price - Entry_Price), 2), 2));
	event.mqh:262 double lots = calcLots(Entry_Price - SL_Price);
	event.mqh:284 double lots = calcLots(SL_Price - Entry_Price);
	event.mqh:341 double lots = calcLots(SL_Price - Entry_Price);
	event.mqh:351 double lots = calcLots(Entry_Price - SL_Price);

ParsePosNoFromName [Hilfsfunktion / Logik]

```
int ParsePosNoFromName(const string name, const string prefix)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 855 bis 868

Kommentar direkt darueber: Liefert pos_no aus "TP_Long_3" => 3; aus "TP_Long" => 1

Rueckgabe	Parameter (Typ name [default])
int	const string name const string prefix

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(name == prefix)
- if(StringLen(name) <= baseLen + 1)
- if(StringSubstr(name, baseLen, 1) != "_")

Wichtige Codezeilen (zum Mitlesen im Editor)

```
857: if(name == prefix)
858: return 1;
862: if(StringLen(name) <= baseLen + 1)
863: return 0;
864: if(StringSubstr(name, baseLen, 1) != "_")
865: return 0;
867: return (int)StringToInteger(StringSubstr(name, baseLen + 1));
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	Trade Assistent V2.008.mq5:886 pos_no = ParsePosNoFromName(obj, SL_Long); Trade Assistent V2.008.mq5:893 pos_no = ParsePosNoFromName(obj, Entry_Long); Trade Assistent V2.008.mq5:902 pos_no = ParsePosNoFromName(obj, SL_Short); Trade Assistent V2.008.mq5:909 pos_no = ParsePosNoFromName(obj, Entry_Short);

ParseTradeLineName [Hilfsfunktion / Logik]

```
bool ParseTradeLineName(const string obj,
                        string &direction,
                        string &kind,
                        int &pos_no)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 871 bis 915

Kommentar direkt darueber: Erkennen: ist es eine Trade-Linie? -> direction, kind(ENTRY/SL/TP), pos_no

Rueckgabe	Parameter (Typ name [default])
bool	const string obj string &direction string &kind int &pos_no

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: ParsePosNoFromName.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(obj == SL_Long || StringFind(obj, SL_Long + "_") == 0)
- if(obj == Entry_Long || StringFind(obj, Entry_Long + "_") == 0)
- if(obj == SL_Short || StringFind(obj, SL_Short + "_") == 0)
- if(obj == Entry_Short || StringFind(obj, Entry_Short + "_") == 0)
- if(pos_no < 1 || pos_no > DB_MAX_POS_PER_SIDE)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
882: if(obj == SL_Long || StringFind(obj, SL_Long + "_") == 0)
889: if(obj == Entry_Long || StringFind(obj, Entry_Long + "_") == 0)
898: if(obj == SL_Short || StringFind(obj, SL_Short + "_") == 0)
905: if(obj == Entry_Short || StringFind(obj, Entry_Short + "_") == 0)
912: if(pos_no < 1 || pos_no > DB_MAX_POS_PER_SIDE)
913: return false;
914: return (direction != "" && kind != "");
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
ParsePosNoFromName	(keine Treffer im Projekt)

DB_LoadPositionOne [Persistenz (DB/Datei)]

```
bool DB_LoadPositionOne(const string direction, const int trade_no, const int pos_no, DB_PositionR
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 918 bis 973

Kommentar direkt darueber: Ein einzelnes Row aus positions laden (damit wir sabio/status/flags nicht überschreiben)

Rueckgabe	Parameter (Typ name [default])
bool	const string direction const int trade_no const int pos_no DB_PositionRow &out_row

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: TF_ToString, DB_IsReady.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(rq == INVALID_HANDLE)
- if(DatabaseRead(rq))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
920: if(!DB_IsReady())
921: return false;
934: if(rq == INVALID_HANDLE)
935: return false;
944: if(DatabaseRead(rq))
972: return found;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_IsReady, TF_ToString	(keine Treffer im Projekt)

DB_RestoreTradeLines_All [Persistenz (DB/Datei)]

```
void DB_RestoreTradeLines_All()
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 980 bis 1029

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).
- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LoadPositions.

2 2. falls Preis außerhalb des sichtbaren Bereichs -> Chart-Mitte: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: UI_DrawPriceOrMid, CreateEntryAndSLLines.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(n <= 0)
- for(int i = 0; i < n; i++)
- if(rows[i].was_sent != 1)
- if(rows[i].pos_no < 1 || rows[i].pos_no > 4)
- if(rows[i].direction == "LONG")

- if(active_long_trade_no <= 0)
- if(rows[i].trade_no != active_long_trade_no)
- if(rows[i].direction == "SHORT")
- if(active_short_trade_no <= 0)
- if(rows[i].trade_no != active_short_trade_no)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
 982: DB_PositionRow rows[];
 983: int n = DB_LoadPositions(_Symbol, _Period, rows);
 984: if(n <= 0)
 985: return;
 987: for(int i = 0; i < n; i++)
 990: if(rows[i].was_sent != 1)
 993: // nur Positions 1..4  (BUGFIX)
 994: if(rows[i].pos_no < 1 || rows[i].pos_no > 4)
1001: if(rows[i].direction == "LONG")
1003: if(active_long_trade_no <= 0)
1005: if(rows[i].trade_no != active_long_trade_no)
1014: if(rows[i].direction == "SHORT")
1016: if(active_short_trade_no <= 0)
1018: if(rows[i].trade_no != active_short_trade_no)
1028: ChartRedraw(0);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
CreateEntryAndSLLines, DB_LoadPositions, UI_DrawPriceOrMid	(keine Treffer im Projekt)

RollbackDraftRow [Hilfsfunktion / Logik]

```
void RollbackDraftRow(const DB_PositionRow &row)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 1038 bis 1042

Kommentar direkt darueber: 3.2 RollbackDraft(int idx) ersetzen (DB-rollback statt TradeInfo reset)

Rueckgabe	Parameter (Typ name [default])
void	const DB_PositionRow &row

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_DeletePosition.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
1039: {
1040: // Draft wieder entfernen, damit PosNo nicht “verbrannt” ist
1041: DB_DeletePosition(row.symbol, (ENUM_TIMEFRAMES)_Period, row.direction, row.trade_no, row.po
1041: DB_DeletePosition(row.symbol, (ENUM_TIMEFRAMES)_Period, row.direction, row.trade_no, row.po
1042: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_DeletePosition	discord_send.mqh:114 RollbackDraftRow(row);

UI_SyncDirectionFromBaseLines [UI / Chart-Objekte]

```
void UI_SyncDirectionFromBaseLines()
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 1047 bis 1055

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: Get_Price_d.

2 2. SL unter TP (klassisch). Wenn SL über TP -> SHORT: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
1048: {
1050: double pr = Get_Price_d(PR_HL);
1051: double sl = Get_Price_d(SL_HL);
1053: // BUY: SL unter TP (klassisch). Wenn SL über TP -> SHORT
1054: ui_direction_is_long = (sl < pr);
1055: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
Get_Price_d	(keine Treffer im Projekt)

UI_SetIntEdit [UI / Chart-Objekte]

```
void UI_SetIntEdit(const string obj, int v)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 1061 bis 1064

Rueckgabe	Parameter (Typ name [default])
void	const string obj int v

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: update_Text.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
1062: {
1063: update_Text(obj, IntegerToString(v));
1064: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
update_Text	Trade Assistent V2.008.mq5:1084 UI_SetIntEdit(TRNB, trade_no); Trade Assistent V2.008.mq5:1085 UI_SetIntEdit(POSNB, pos_no);

UI_UpdateNextTradePosUI [UI / Chart-Objekte]

```
void UI_UpdateNextTradePosUI()
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 1069 bis 1086

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: UI_SetIntEdit, DB_GetNextPosNo.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(trade_no <= 0)
- if(pos_no < 1)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
1070: {
1071: bool isLong = ui_direction_is_long;
1072: string dir = (isLong ? "LONG" : "SHORT");
1077: if(trade_no <= 0)
1080: int pos_no = DB_GetNextPosNo(_Symbol, (ENUM_TIMEFRAMES)_Period, dir, trade_no);
1081: if(pos_no < 1)
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

DB_GetNextPosNo, UI_SetIntEdit	discord_send.mqh:190 UI_UpdateNextTradePosUI();
	event.mqh:420 UI_UpdateNextTradePosUI();
	event.mqh:564 UI_UpdateNextTradePosUI();
	trades_panel.mqh:781 UI_UpdateNextTradePosUI();
	trades_panel.mqh:826 UI_UpdateNextTradePosUI();

UI_SetLineMeta [UI / Chart-Objekte]

```
void UI_SetLineMeta(const string lineName, int trade_no, int pos_no, const string dir, const strin
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 1090 bis 1094

Rueckgabe	Parameter (Typ name [default])
void	const string lineName int trade_no int pos_no const string dir const string kind

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
1091: {
1092: string meta = StringFormat("T=%d;P=%d;D=%s;K=%s", trade_no, pos_no, dir, kind);
1093: ObjectSetString(0, lineName, OBJPROP_TOOLTIP, meta);
1093: ObjectSetString(0, lineName, OBJPROP_TOOLTIP, meta);
1094: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	(keine Treffer im Projekt)

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_TOOLTIP

UI_ParseLineMeta [UI / Chart-Objekte]

```
bool UI_ParseLineMeta(const string meta, int &trade_no, int &pos_no, string &dir, string &kind)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 1099 bis 1125

Rueckgabe	Parameter (Typ name [default])
bool	const string meta int &trade_no int &pos_no string &dir string &kind

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(c < 4)
- for(int i = 0; i < c; i = i + 1)
- if(StringSplit(parts[i], '=', kv) != 2)
- if(kv[0] == "T")
- if(kv[0] == "P")
- if(kv[0] == "D")
- if(kv[0] == "K")

Wichtige Codezeilen (zum Mitlesen im Editor)

```
1107: if(c < 4)
1108: return false;
1110: for(int i = 0; i < c; i = i + 1)
1113: if(StringSplit(parts[i], '=', kv) != 2)
1115: if(kv[0] == "T")
1117: if(kv[0] == "P")
1119: if(kv[0] == "D")
1121: if(kv[0] == "K")
1124: return (trade_no > 0 && pos_no > 0 && dir != "" && kind != "");
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	(keine Treffer im Projekt)

createEntryAndSLLinien [Hilfsfunktion / Logik]

```
void createEntryAndSLLinien()
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 1140 bis 1213

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: createHL, Get_Price_s, createButton, getChartWidthInPixels, calcLots, SabioEdit, SetPriceOnObject, SendButton.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(Sabioedit)
- if(!SendOnlyButton)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
1173: ObjectMove(0, EntryButton, 0, dt_prc, price_prc);
1179: if(Sabioedit)
1188: if(!SendOnlyButton)
1190: ObjectSetString(0, SENDTRADEBTN, OBJPROP_TEXT, "T & S"); // label
1191: ObjectSetInteger(0, SENDTRADEBTN, OBJPROP_BGCOLOR, TSButton_bgcolor);
1192: ObjectSetInteger(0, SENDTRADEBTN, OBJPROP_COLOR, TSButton_font_color);
1200: ObjectMove(0, SLButton, 0, dt_sl, price_sl);
1212: ChartRedraw(0);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
Get_Price_d, Get_Price_s, SabioEdit, SendButton, SetPriceOnObject, calcLots, createButton, createHL, createHLine, getChartHeightInPixels, getChartWidthInPixels, update_Text	Trade Assistent V2.008.mq5:237 createEntryAndSLLinien(); // erzeuge den EntryButton und den SL Button und deren HL Linien:

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_BGCOLOR, OBJPROP_COLOR, OBJPROP_TEXT
- Chart-Properties: CHART_EVENT_MOUSE_MOVE

createHLine [Hilfsfunktion / Logik]

```
bool createHLine(const string name, const double price, color clr)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 1218 bis 1240

Rueckgabe	Parameter (Typ name [default])
bool	const string name const double price color clr

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben; Logging/Benachrichtigung. Wichtige Aufrufe: UI_Reg_Add.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, name) >= 0)
- if(!ObjectCreate(0, name, OBJ_HLINE, 0, 0, price))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
1220: if(ObjectFind(0, name) >= 0)
1221: return true;
1223: if(!ObjectCreate(0, name, OBJ_HLINE, 0, 0, price))
1225: Print("createHLine: create failed for ", name, " err=", GetLastError());
1226: return false;
1228: ObjectSetInteger(0, name, OBJPROP_SELECTABLE, true);
1229: ObjectSetInteger(0, name, OBJPROP_HIDDEN, false);
1230: //ObjectSetInteger(0, objName, OBJPROP_TIME, time1);
1231: ObjectSetDouble(0, name, OBJPROP_PRICE, price);
1232: ObjectSetInteger(0, name, OBJPROP_COLOR, clr);
1233: ObjectSetInteger(0, name, OBJPROP_BACK, false);
1234: ObjectSetInteger(0, name, OBJPROP_STYLE, STYLE_SOLID);
1235: ObjectSetInteger(0, name, OBJPROP_ZORDER, 10);
1239: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_Add	Trade Assistent V2.008.mq5:1154 createHLine(PR_HL, price_prc,color_EntryLine);

Verwendete MQL-Konstanten (UI)

- Objekt-Typen: OBJ_HLINE
- Objekt-Properties: OBJPROP_BACK, OBJPROP_COLOR, OBJPROP_HIDDEN, OBJPROP_PRICE, OBJPROP_SELECTABLE, OBJPROP_STYLE, OBJPROP_TIME, OBJPROP_ZORDER

UI_IsTradePosLine [UI / Chart-Objekte]

```
bool UI_IsTradePosLine(const string name)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 1250 bis 1262

Kommentar direkt darueber: --- Trade-Line Erkennung/Parsing ----------

Rueckgabe	Parameter (Typ name [default])
bool	const string name

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(L >= 4 && StringSubstr(name, L-4) == "_TAG")

Wichtige Codezeilen (zum Mitlesen im Editor)

```
1251: {
1252: // Tags nicht als Linien behandeln
1253: int L = (int)StringLen(name);
1254: if(L >= 4 && StringSubstr(name, L-4) == "_TAG") return false;
1254: if(L >= 4 && StringSubstr(name, L-4) == "_TAG") return false;
1256: return (
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	Trade Assistent V2.008.mq5:1337 if(UI_IsTradePosLine(name)) event.mqh:157 if(UI_IsTradePosLine(sparam)) event.mqh:175 if(sparam == PR_HL || sparam == SL_HL || UI_IsTradePosLine(sparam)) event.mqh:177 if(UI_IsTradePosLine(sparam)) gui_elemente.mqh:126 if(!UI_IsTradePosLine(name)) gui_elemente.mqh:183 if(!UI_IsTradePosLine(name))

UI_CreateOrUpdateLineTag [UI / Chart-Objekte]

```
bool UI_CreateOrUpdateLineTag(const string line_name)
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 1265 bis 1328

Kommentar direkt darueber: Erzeugt/updated das Tag-Label für genau DIESE Linie

Rueckgabe	Parameter (Typ name [default])
bool	const string line_name

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung
2. 2. T/P aus Suffix ziehen: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: UI_Reg_Add.
3. 3. Abschnitt: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, line_name) < 0)
- if(n >= 2)
- if(StringFind(line_name, Entry_Long, 0) == 0)
- if(StringFind(line_name, SL_Long, 0) == 0)
- if(StringFind(line_name, Entry_Short, 0) == 0)
- if(StringFind(line_name, SL_Short, 0) == 0)
- if(ObjectFind(0, tag_name) < 0)
- if(!ObjectCreate(0, tag_name, OBJ_LABEL, 0, 0, 0))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
1267: if(ObjectFind(0, line_name) < 0)
1268: return false;
1290: if(n >= 2)
1297: if(StringFind(line_name, Entry_Long, 0) == 0)  kind = "ENTRY LONG";
1308: if(!ObjectCreate(0, tag_name, OBJ_LABEL, 0, 0, 0))
1309: return false;
1313: ObjectSetInteger(0, tag_name, OBJPROP_CORNER, CORNER_LEFT_UPPER);
1314: ObjectSetInteger(0, tag_name, OBJPROP_FONTSIZE, 9);
1315: ObjectSetInteger(0, tag_name, OBJPROP_COLOR, line_clr);
1318: ObjectSetInteger(0, tag_name, OBJPROP_SELECTABLE, false);
1319: ObjectSetInteger(0, tag_name, OBJPROP_SELECTED,   false);
1320: ObjectSetInteger(0, tag_name, OBJPROP_BACK,       false);
1323: ObjectSetInteger(0, tag_name, OBJPROP_XDISTANCE, x);
1324: ObjectSetInteger(0, tag_name, OBJPROP_YDISTANCE, y_adj-5);
1325: ObjectSetString(0,  tag_name, OBJPROP_TEXT,      txt);
1327: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_Add	Trade Assistent V2.008.mq5:1338 UI_CreateOrUpdateLineTag(name); event.mqh:42 UI_CreateOrUpdateLineTag(g_tp_drag_name); event.mqh:143 UI_CreateOrUpdateLineTag(sparam); event.mqh:159 UI_CreateOrUpdateLineTag(sparam); event.mqh:178 UI_CreateOrUpdateLineTag(sparam); trades_panel.mqh:555 UI_CreateOrUpdateLineTag(Entry_Long + suf); trades_panel.mqh:557 UI_CreateOrUpdateLineTag(SL_Long + suf); trades_panel.mqh:564 UI_CreateOrUpdateLineTag(Entry_Short + suf); ... (1 weitere)

Verwendete MQL-Konstanten (UI)

- Objekt-Typen: OBJ_LABEL
- Objekt-Properties: OBJPROP_BACK, OBJPROP_COLOR, OBJPROP_CORNER, OBJPROP_FONTSIZE, OBJPROP_PRICE, OBJPROP_SELECTABLE, OBJPROP_SELECTED, OBJPROP_TEXT, OBJPROP_XDISTANCE, OBJPROP_YDISTANCE
- Chart-Properties: CHART_WIDTH_IN_PIXELS

UI_UpdateAllLineTags [UI / Chart-Objekte]

```
void UI_UpdateAllLineTags()
```

Ort im Code: Trade Assistent V2.008.mq5 Zeile 1331 bis 1340

Kommentar direkt darueber: Update für alle Trade-Linien per Scan

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_CreateOrUpdateLineTag, UI_IsTradePosLine.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int i = total - 1; i >= 0; --i)
- if(UI_IsTradePosLine(name))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
1332: {
1333: const int total = ObjectsTotal(0, -1, -1);
1334: for(int i = total - 1; i >= 0; --i)
1334: for(int i = total - 1; i >= 0; --i)
1335: {
1337: if(UI_IsTradePosLine(name))
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

UI_CreateOrUpdateLineTag, UI_IsTradePosLine	event.mqh:422 UI_UpdateAllLineTags();
	event.mqh:521 UI_UpdateAllLineTags();
	trades_panel.mqh:689 UI_UpdateAllLineTags();

db_state.mqh

Persistenz: DB/Datei-Speicher fuer Trades, Positionen, Linien, Meta-Daten

Funktionen in dieser Datei (32): TF_ToString, DB_IsReady, DB_DefaultFile, DB_Key, DB_LogErr, DB_Open, DB_Close, DB_Exec, DB_ExecPreparedNonQuery, DB_CreateSchema, DB_Init, DB_SetMetaText, DB_GetMetaText, DB_SetMetaInt, DB_GetMetaInt, DB_GetMetaInt, DB_UpsertPosition, DB_LoadPositions, DB_PrintData, SetPriceOnObject, UI_SnapObjectBottomToPrice, SetObjectPriceAny, RestoreOne, DB_RestoreTradeLines, DB_GetNextPosNo, GetObjectPriceAny, SaveOne, DB_DeletePositionKey, DB_GetPosition, DB_UpdatePositionStatus, DB_DeletePosition, DB_GetPosNoCount

TF_ToString [Persistenz (DB/Datei)]

```
string TF_ToString(ENUM_TIMEFRAMES tf)
```

Ort im Code: db_state.mqh Zeile 20 bis 47

Kommentar direkt darueber: Helpers

Rueckgabe	Parameter (Typ name [default])
string	ENUM_TIMEFRAMES tf

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
21: {
22: switch(tf)
23: {
24: case PERIOD_M1:  return "M1";
25: case PERIOD_M2:  return "M2";
26: case PERIOD_M3:  return "M3";
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	Trade Assistent V2.008.mq5:923 string tf_txt = TF_ToString((ENUM_TIMEFRAMES)Period());
	db_state.mqh:56 string tf = TF_ToString((ENUM_TIMEFRAMES)Period());
	db_state.mqh:64 string tf = TF_ToString((ENUM_TIMEFRAMES)Period());
	db_state.mqh:360 string tf_txt = TF_ToString(tf);
	db_state.mqh:436 string tf = TF_ToString((ENUM_TIMEFRAMES)Period());
	db_state.mqh:610 string tf_txt = TF_ToString(tf);
	db_state.mqh:715 out_row.tf = TF_ToString(tf);
	db_state.mqh:733 string tf_txt = TF_ToString(tf);
	... (8 weitere)

DB_IsReady [Persistenz (DB/Datei)]

```
bool DB_IsReady()
```

Ort im Code: db_state.mqh Zeile 49 bis 52

Rueckgabe	Parameter (Typ name [default])
bool	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
50: {
51: return (g_db != INVALID_HANDLE);
51: return (g_db != INVALID_HANDLE);
52: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	Trade Assistent V2.008.mq5:920 if(!DB_IsReady()) db_state.mqh:82 if(DB_IsReady()) db_state.mqh:108 if(!DB_IsReady()) db_state.mqh:120 if(!DB_IsReady()) db_state.mqh:203 if(!DB_IsReady()) db_state.mqh:225 if(!DB_IsReady()) db_state.mqh:311 if(!DB_IsReady()) db_state.mqh:357 if(!DB_IsReady()) ... (9 weitere)

DB_DefaultFile [Persistenz (DB/Datei)]

```
string DB_DefaultFile()
```

Ort im Code: db_state.mqh Zeile 54 bis 58

Rueckgabe	Parameter (Typ name [default])
string	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: TF_ToString.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
55: {
56: string tf = TF_ToString((ENUM_TIMEFRAMES)Period());
57: return StringFormat("DowHowState_%s_%s.sqlite", _Symbol, tf);
57: return StringFormat("DowHowState_%s_%s.sqlite", _Symbol, tf);
58: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
TF_ToString	db_state.mqh:85 g_db_file = (file_name == "" ? DB_DefaultFile() : file_name);

DB_Key [Persistenz (DB/Datei)]

```
string DB_Key(const string key)
```

Ort im Code: db_state.mqh Zeile 61 bis 66

Rueckgabe	Parameter (Typ name [default])
string	const string key

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: TF_ToString.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
62: {
63: // Namespaced key to avoid collisions between charts/symbols/timeframes
64: string tf = TF_ToString((ENUM_TIMEFRAMES)Period());
65: return StringFormat("%s|%s|%s", _Symbol, tf, key);
65: return StringFormat("%s|%s|%s", _Symbol, tf, key);
66: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
TF_ToString	Trade Assistent V2.008.mq5:269 DB_SetMetaText(DB_Key("L_sl"), DoubleToString(ObjectGetDouble(0, SL_Long + suf, OBJPROP_PRICE), _Digits)); Trade Assistent V2.008.mq5:271 DB_SetMetaText(DB_Key("L_entry"), DoubleToString(ObjectGetDouble(0, Entry_Long + suf, OBJPROP_PRICE), _Digits)); Trade Assistent V2.008.mq5:276 DB_SetMetaText(DB_Key("S_sl"), DoubleToString(ObjectGetDouble(0, SL_Short + suf, OBJPROP_PRICE), _Digits)); Trade Assistent V2.008.mq5:278 DB_SetMetaText(DB_Key("S_entry"), DoubleToString(ObjectGetDouble(0, Entry_Short + suf, OBJPROP_PRICE), _Digits)); Trade Assistent V2.008.mq5:287 DB_GetMetaInt(DB_Key("last_trade_no"), meta_last, 0); Trade Assistent V2.008.mq5:288 DB_GetMetaInt(DB_Key("active_long_trade_no"), meta_long, 0); Trade Assistent V2.008.mq5:289 DB_GetMetaInt(DB_Key("active_short_trade_no"), meta_short, 0); Trade Assistent V2.008.mq5:327 DB_SetMetaInt(DB_Key("last_trade_no"), last_trade_nummer); ... (24 weitere)

DB_LogErr [Persistenz (DB/Datei)]

```
void DB_LogErr(const string where, const string extra = "")
```

Ort im Code: db_state.mqh Zeile 68 bis 75

Rueckgabe	Parameter (Typ name [default])
void	const string where const string extra = ""

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Logging/Benachrichtigung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(extra == "")

Wichtige Codezeilen (zum Mitlesen im Editor)

```
69: {
70: int err = GetLastError();
71: if(extra == "")
71: if(extra == "")
72: Print(where, ": DB error=", err);
74: Print(where, ": DB error=", err, " | ", extra);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	db_state.mqh:93 DB_LogErr(__FUNCTION__, "DatabaseOpen failed | file=" + g_db_file);
	db_state.mqh:126 DB_LogErr(__FUNCTION__, sql);
	db_state.mqh:153 DB_LogErr(__FUNCTION__, "PreparedNonQuery");
	db_state.mqh:209 DB_LogErr(__FUNCTION__, "Prepare failed");
	db_state.mqh:231 DB_LogErr(__FUNCTION__, "Prepare failed");
	db_state.mqh:325 DB_LogErr(__FUNCTION__, "Prepare failed");
	db_state.mqh:375 DB_LogErr(__FUNCTION__, "Prepare failed");
	db_state.mqh:447 DB_LogErr(__FUNCTION__, "Prepare meta failed");
	... (7 weitere)

DB_Open [Persistenz (DB/Datei)]

```
bool DB_Open(const string file_name = "")
```

Ort im Code: db_state.mqh Zeile 80 bis 104

Kommentar direkt darueber: DB Open/Close

Rueckgabe	Parameter (Typ name [default])
bool	const string file_name = ""

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LogErr, DB_DefaultFile, DB_IsReady.
2. 2. Pragmas (wenn SQLite das nicht mag, einfach weglassen): Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(DB_IsReady())
- if(g_db == INVALID_HANDLE)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
 82: if(DB_IsReady())
 83: return true;
 85: g_db_file = (file_name == "" ? DB_DefaultFile() : file_name);
 91: if(g_db == INVALID_HANDLE)
 93: DB_LogErr(__FUNCTION__, "DatabaseOpen failed | file=" + g_db_file);
 94: return false;
103: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_DefaultFile, DB_IsReady, DB_LogErr	db_state.mqh:193 if(!DB_Open(file_name))

DB_Close [Persistenz (DB/Datei)]

```
void DB_Close()
```

Ort im Code: db_state.mqh Zeile 106 bis 113

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_IsReady.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())

Wichtige Codezeilen (zum Mitlesen im Editor)

```
107: {
108: if(!DB_IsReady())
108: if(!DB_IsReady())
109: return;
109: return;
111: DatabaseClose(g_db);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_IsReady	Trade Assistent V2.008.mq5:738 DB_Close();

DB_Exec [Persistenz (DB/Datei)]

```
bool DB_Exec(const string sql)
```

Ort im Code: db_state.mqh Zeile 118 bis 130

Kommentar direkt darueber: Exec helpers

Rueckgabe	Parameter (Typ name [default])
bool	const string sql

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LogErr, DB_IsReady.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(!DatabaseExecute(g_db, sql))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
120: if(!DB_IsReady())
121: return false;
```

```
124: if(!DatabaseExecute(g_db, sql))
126: DB_LogErr(__FUNCTION__, sql);
127: return false;
129: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_IsReady, DB_LogErr	db_state.mqh:165 ok &= DB_Exec("CREATE TABLE IF NOT EXISTS meta (" db_state.mqh:170 ok &= DB_Exec("CREATE TABLE IF NOT EXISTS positions ("

DB_ExecPreparedNonQuery [Persistenz (DB/Datei)]

```
bool DB_ExecPreparedNonQuery(const int request)
```

Ort im Code: db_state.mqh Zeile 133 bis 156

Kommentar direkt darueber: Prepared statements werden über DatabaseRead() ausgeführt:contentReference[oaicite:5]{index=5}.

Rueckgabe	Parameter (Typ name [default])
bool	const int request

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LogErr.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(DatabaseRead(request))
- if(err == ERR_DATABASE_NO_MORE_DATA)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
139: if(DatabaseRead(request))
142: return true;
146: if(err == ERR_DATABASE_NO_MORE_DATA)
150: return true;
153: DB_LogErr(__FUNCTION__, "PreparedNonQuery");
155: return false;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

DB_LogErr	db_state.mqh:216 bool ok = DB_ExecPreparedNonQuery(rq);
	db_state.mqh:348 bool ok = DB_ExecPreparedNonQuery(rq);
	db_state.mqh:701 bool ok = DB_ExecPreparedNonQuery(rq);
	db_state.mqh:841 bool ok = DB_ExecPreparedNonQuery(rq);
	db_state.mqh:879 bool ok = DB_ExecPreparedNonQuery(rq);

DB_CreateSchema [Persistenz (DB/Datei)]

```
bool DB_CreateSchema()
```

Ort im Code: db_state.mqh Zeile 161 bis 189

Kommentar direkt darueber: Schema

Rueckgabe	Parameter (Typ name [default])
bool	(keine)

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_Exec.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
162: {
163: bool ok = true;
165: ok &= DB_Exec("CREATE TABLE IF NOT EXISTS meta ("
165: ok &= DB_Exec("CREATE TABLE IF NOT EXISTS meta ("
170: ok &= DB_Exec("CREATE TABLE IF NOT EXISTS positions ("
188: return ok;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_Exec	db_state.mqh:195 return DB_CreateSchema();

DB_Init [Persistenz (DB/Datei)]

```
bool DB_Init(const string file_name = "")
```

Ort im Code: db_state.mqh Zeile 191 bis 196

Rueckgabe	Parameter (Typ name [default])
bool	const string file_name = ""

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_Open, DB_CreateSchema.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_Open(file_name))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
192: {
193: if(!DB_Open(file_name))
193: if(!DB_Open(file_name))
194: return false;
194: return false;
195: return DB_CreateSchema();
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_CreateSchema, DB_Open	Trade Assistent V2.008.mq5:227 if(!DB_Init())

DB_SetMetaText [Persistenz (DB/Datei)]

```
bool DB_SetMetaText(const string key, const string value)
```

Ort im Code: db_state.mqh Zeile 201 bis 219

Kommentar direkt darueber: META key/value

Rueckgabe	Parameter (Typ name [default])
bool	const string key const string value

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LogErr, DB_IsReady, DB_ExecPreparedNonQuery.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(rq == INVALID_HANDLE)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
203: if(!DB_IsReady())
204: return false;
207: if(rq == INVALID_HANDLE)
209: DB_LogErr(__FUNCTION__, "Prepare failed");
210: return false;
216: bool ok = DB_ExecPreparedNonQuery(rq);
218: return ok;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

DB_ExecPreparedNonQuery, DB_IsReady, DB_LogErr	Trade Assistent V2.008.mq5:269 DB_SetMetaText(DB_Key("L_sl"), DoubleToString(ObjectGetDouble(0, SL_Long + suf, OBJPROP_PRICE), _Digits));
	Trade Assistent V2.008.mq5:271 DB_SetMetaText(DB_Key("L_entry"), DoubleToString(ObjectGetDouble(0, Entry_Long + suf, OBJPROP_PRICE), _Digits));
	Trade Assistent V2.008.mq5:276 DB_SetMetaText(DB_Key("S_sl"), DoubleToString(ObjectGetDouble(0, SL_Short + suf, OBJPROP_PRICE), _Digits));
	Trade Assistent V2.008.mq5:278 DB_SetMetaText(DB_Key("S_entry"), DoubleToString(ObjectGetDouble(0, Entry_Short + suf, OBJPROP_PRICE), _Digits));
	db_state.mqh:260 return DB_SetMetaText(key, IntegerToString(value));
	db_state.mqh:667 DB_SetMetaText(DB_Key(key), DoubleToString(p, _Digits));
	gui_elemente.mqh:112 DB_SetMetaText(DB_Key("price_entry"), DoubleToString(p, _Digits));
	gui_elemente.mqh:118 DB_SetMetaText(DB_Key("price_sl"), DoubleToString(p, _Digits));
	... (4 weitere)

DB_GetMetaText [Persistenz (DB/Datei)]

```
bool DB_GetMetaText(const string key, string &out_value, const string default_value = "")
```

Ort im Code: db_state.mqh Zeile 221 bis 256

Rueckgabe	Parameter (Typ name [default])
bool	const string key string &out_value const string default_value = ""

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LogErr, DB_IsReady.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(rq == INVALID_HANDLE)
- if(DatabaseRead(rq)) // muss vor DatabaseColumnXXX()
- if(DatabaseColumnText(rq, 0, v))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
225: if(!DB_IsReady())
226: return false;
229: if(rq == INVALID_HANDLE)
231: DB_LogErr(__FUNCTION__, "Prepare failed");
232: return false;
238: if(DatabaseRead(rq)) // muss vor DatabaseColumnXXX() passieren:contentReference[oaicite:6]{
241: if(DatabaseColumnText(rq, 0, v))
255: return found;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_IsReady, DB_LogErr	Trade Assistent V2.008.mq5:494 if(!DB_GetMetaText(DB_Key(key), s, "")) db_state.mqh:272 if(!DB_GetMetaText(key, s, "")) db_state.mqh:580 if(!DB_GetMetaText(DB_Key(key), s, "")) return false;

DB_SetMetaInt [Persistenz (DB/Datei)]

```
bool DB_SetMetaInt(const string key, const int value)
```

Ort im Code: db_state.mqh Zeile 258 bis 261

Rueckgabe	Parameter (Typ name [default])
bool	const string key const int value

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_SetMetaText.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
259: {
260: return DB_SetMetaText(key, IntegerToString(value));
260: return DB_SetMetaText(key, IntegerToString(value));
261: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_SetMetaText	Trade Assistent V2.008.mq5:327 DB_SetMetaInt(DB_Key("last_trade_no"), last_trade_nummer); Trade Assistent V2.008.mq5:328 DB_SetMetaInt(DB_Key("active_long_trade_no"), active_long_trade_no); Trade Assistent V2.008.mq5:329 DB_SetMetaInt(DB_Key("active_short_trade_no"), active_short_trade_no); Trade Assistent V2.008.mq5:458 DB_SetMetaInt(DB_Key("active_long_trade_no"), 0); Trade Assistent V2.008.mq5:475 DB_SetMetaInt(DB_Key("active_short_trade_no"), 0); Trade Assistent V2.008.mq5:749 DB_SetMetaInt(DB_Key("last_trade_no"), last_trade_nummer); Trade Assistent V2.008.mq5:750 DB_SetMetaInt(DB_Key("active_long_trade_no"), active_long_trade_no); Trade Assistent V2.008.mq5:751 DB_SetMetaInt(DB_Key("active_short_trade_no"), active_short_trade_no); ... (8 weitere)

DB_GetMetaInt [Persistenz (DB/Datei)]

```
bool DB_GetMetaInt(const string key, int &out_value)
```

Ort im Code: db_state.mqh Zeile 264 bis 267

Rueckgabe	Parameter (Typ name [default])
bool	const string key int &out_value

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_GetMetaInt.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
265: {
266: return DB_GetMetaInt(key, out_value, 0);
266: return DB_GetMetaInt(key, out_value, 0);
267: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	Trade Assistent V2.008.mq5:287 DB_GetMetaInt(DB_Key("last_trade_no"), meta_last, 0); Trade Assistent V2.008.mq5:288 DB_GetMetaInt(DB_Key("active_long_trade_no"), meta_long, 0); Trade Assistent V2.008.mq5:289 DB_GetMetaInt(DB_Key("active_short_trade_no"), meta_short, 0);

DB_GetMetaInt [Persistenz (DB/Datei)]

```
bool DB_GetMetaInt(const string key, int &out_value, const int default_value)
```

Ort im Code: db_state.mqh Zeile 269 bis 279

Rueckgabe	Parameter (Typ name [default])
bool	const string key int &out_value const int default_value

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_GetMetaText.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_GetMetaText(key, s, ""))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
270: {
271: string s = "";
272: if(!DB_GetMetaText(key, s, ""))
272: if(!DB_GetMetaText(key, s, ""))
275: return false;
278: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_GetMetaText	Trade Assistent V2.008.mq5:287 DB_GetMetaInt(DB_Key("last_trade_no"), meta_last, 0); Trade Assistent V2.008.mq5:288 DB_GetMetaInt(DB_Key("active_long_trade_no"), meta_long, 0); Trade Assistent V2.008.mq5:289 DB_GetMetaInt(DB_Key("active_short_trade_no"), meta_short, 0);

DB_UpsertPosition [Persistenz (DB/Datei)]

```
bool DB_UpsertPosition(const DB_PositionRow &p)
```

Ort im Code: db_state.mqh Zeile 309 bis 351

Rueckgabe	Parameter (Typ name [default])
bool	const DB_PositionRow &p

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LogErr, DB_IsReady, DB_ExecPreparedNonQuery.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(rq == INVALID_HANDLE)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
311: if(!DB_IsReady())
312: return false;
323: if(rq == INVALID_HANDLE)
```

```
325: DB_LogErr(__FUNCTION__, "Prepare failed");
326: return false;
348: bool ok = DB_ExecPreparedNonQuery(rq);
350: return ok;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

DB_ExecPreparedNonQuery, DB_IsReady, DB_LogErr	Trade Assistent V2.008.mq5:652 DB_UpsertPosition(p);
	Trade Assistent V2.008.mq5:672 DB_UpsertPosition(p);
	Trade Assistent V2.008.mq5:689 DB_UpsertPosition(p);
	discord_send.mqh:100 if(!DB_UpsertPosition(row))
	discord_send.mqh:127 DB_UpsertPosition(row);
	gui_elemente.mqh:154 DB_UpsertPosition(row);
	gui_elemente.mqh:202 DB_UpsertPosition(row);

DB_LoadPositions [Persistenz (DB/Datei)]

```
int DB_LoadPositions(const string symbol, ENUM_TIMEFRAMES tf, DB_PositionRow &out_rows[])
```

Ort im Code: db_state.mqh Zeile 353 bis 422

Rueckgabe	Parameter (Typ name [default])
int	const string symbol ENUM_TIMEFRAMES tf DB_PositionRow &out_rows[]

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LogErr, TF_ToString, DB_IsReady.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(rq == INVALID_HANDLE)
- while(DatabaseRead(rq))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
357: if(!DB_IsReady())
358: return 0;
373: if(rq == INVALID_HANDLE)
375: DB_LogErr(__FUNCTION__, "Prepare failed");
376: return 0;
382: while(DatabaseRead(rq))
384: DB_PositionRow r;
421: return ArraySize(out_rows);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

DB_IsReady, DB_LogErr, TF_ToString	Trade Assistent V2.008.mq5:296 int n = DB_LoadPositions(_Symbol, _Period, rows);
	Trade Assistent V2.008.mq5:606 int n = DB_LoadPositions(_Symbol, _Period, rows);
	Trade Assistent V2.008.mq5:983 int n = DB_LoadPositions(_Symbol, _Period, rows);
	event.mqh:435 int n = DB_LoadPositions(_Symbol, _Period, rows);
	trades_panel.mqh:5 #include "db_state.mqh" // DB_LoadPositions(), DB_PositionRow
	trades_panel.mqh:394 int n = DB_LoadPositions(_Symbol, (ENUM_TIMEFRAMES)_Period, rows);
	trades_panel.mqh:528 int n = DB_LoadPositions(_Symbol, (ENUM_TIMEFRAMES)_Period, rows);

DB_PrintData [Persistenz (DB/Datei)]

```
void DB_PrintData()
```

Ort im Code: db_state.mqh Zeile 428 bis 514

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Persistenz (DB/Datei) lesen/schreiben; Logging/Benachrichtigung. Wichtige Aufrufe: TF_ToString, DB_IsReady.
2. 2. META (nur aktueller Namespace) ---: Persistenz (DB/Datei) lesen/schreiben; Logging/Benachrichtigung. Wichtige Aufrufe: DB_LogErr.
3. 3. POSITIONS (aktuelles Symbol/TF) ---: Persistenz (DB/Datei) lesen/schreiben; Logging/Benachrichtigung. Wichtige Aufrufe: DB_LogErr.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(rq == INVALID_HANDLE)
- while(DatabaseRead(rq))
- if(rq == INVALID_HANDLE)
- while(DatabaseRead(rq))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
430: if(!DB_IsReady())
432: Print("DB_PrintData: DB not ready");
433: return;
439: Print("========== DB DUMP START ==========");
440: Print("DB file: ", g_db_file);
441: Print("Namespace prefix: ", prefix);
445: if(rq == INVALID_HANDLE)
447: DB_LogErr(__FUNCTION__, "Prepare meta failed");
448: return;
459: Print("META | ", k, " = ", v);
463: Print("META rows: ", metaCount);
473: DB_LogErr(__FUNCTION__, "Prepare positions failed");
474: return;
501: Print("POS | ", dir,
511: Print("POSITIONS rows: ", posCount);
513: Print("========== DB DUMP END ==========");
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_IsReady, DB_LogErr, TF_ToString	Trade Assistent V2.008.mq5:229 DB_PrintData();

SetPriceOnObject [Persistenz (DB/Datei)]

```
bool SetPriceOnObject(const string name, const double price)
```

Ort im Code: db_state.mqh Zeile 517 bis 530

Rueckgabe	Parameter (Typ name [default])
bool	const string name const double price

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung
2. 2. Abschnitt: UI/Chart-Objekte anlegen/aktualisieren/verschieben

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, name) < 0)
- if(type == OBJ_HLINE)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
518: {
519: if(ObjectFind(0, name) < 0)
520: return false;
524: if(type == OBJ_HLINE)
525: return ObjectSetDouble(0, name, OBJPROP_PRICE, price);
529: return ObjectMove(0, name, 0, t, price);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	Trade Assistent V2.008.mq5:1155 SetPriceOnObject(PR_HL, price_prc);

Verwendete MQL-Konstanten (UI)

- Objekt-Typen: OBJ_HLINE
- Objekt-Properties: OBJPROP_PRICE, OBJPROP_TYPE

UI_SnapObjectBottomToPrice [Persistenz (DB/Datei)]

```
bool UI_SnapObjectBottomToPrice(const string objName, const double price)
```

Ort im Code: db_state.mqh Zeile 532 bis 549

Kommentar direkt darueber: --- UI helper: snap a screen object (button/edit) so its BOTTOM edge aligns with a given chart price

Rueckgabe	Parameter (Typ name [default])
bool	const string objName const double price

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, objName) < 0)
- if(!ChartTimePriceToXY(0, subwin, t, price, x, y))
- if(ydist < 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
534: if(ObjectFind(0, objName) < 0)
535: return false;
541: if(!ChartTimePriceToXY(0, subwin, t, price, x, y))
542: return false;
546: if(ydist < 0) ydist = 0;
548: return ObjectSetInteger(0, objName, OBJPROP_YDISTANCE, ydist);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	(keine Treffer im Projekt)

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_YDISTANCE, OBJPROP_YSIZE

SetObjectPriceAny [Persistenz (DB/Datei)]

```
bool SetObjectPriceAny(const string name, const double price)
```

Ort im Code: db_state.mqh Zeile 558 bis 571

Rueckgabe	Parameter (Typ name [default])
bool	const string name const double price

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, name) < 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
559: {
560: if(ObjectFind(0, name) < 0)
561: return false;
564: bool ok0 = ObjectSetDouble(0, name, OBJPROP_PRICE, 0, price);
568: ObjectSetDouble(0, name, OBJPROP_PRICE, 1, price);
570: return ok0;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	db_state.mqh:586 bool ok = SetObjectPriceAny(obj_name, p);

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_PRICE

RestoreOne [Persistenz (DB/Datei)]

```
bool RestoreOne(const string key, const string obj_name)
```

Ort im Code: db_state.mqh Zeile 577 bis 589

Rueckgabe	Parameter (Typ name [default])
bool	const string key const string obj_name

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben; Logging/Benachrichtigung. Wichtige Aufrufe: DB_Key, DB_GetMetaText, SetObjectPriceAny.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_GetMetaText(DB_Key(key), s, ""))
- if(StringLen(s) == 0)
- if(p <= 0.0)
- if(!ok) Print("RestoreOne FAILED: obj=", obj_name, " key=", key, " p=", DoubleToString(p,_Digits), " err=", GetLastError())

Wichtige Codezeilen (zum Mitlesen im Editor)

```
578: {
580: if(!DB_GetMetaText(DB_Key(key), s, "")) return false;
581: if(StringLen(s) == 0) return false;
584: if(p <= 0.0) return false;
587: if(!ok) Print("RestoreOne FAILED: obj=", obj_name, " key=", key, " p=", DoubleToString(p,_D
588: return ok;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

DB_GetMetaText, DB_Key, SetObjectPriceAny	db_state.mqh:594 RestoreOne("L_entry", Entry_Long + suf);
	db_state.mqh:595 RestoreOne("L_sl", SL_Long + suf);
	db_state.mqh:599 RestoreOne("S_entry", Entry_Short + suf);
	db_state.mqh:600 RestoreOne("S_sl", SL_Short + suf);

DB_RestoreTradeLines [Persistenz (DB/Datei)]

```
void DB_RestoreTradeLines(const string suf)
```

Ort im Code: db_state.mqh Zeile 591 bis 603

Rueckgabe	Parameter (Typ name [default])
void	const string suf

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: RestoreOne.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
592: {
593: // LONG
594: RestoreOne("L_entry", Entry_Long + suf);
595: RestoreOne("L_sl",    SL_Long    + suf);
598: // SHORT
602: ChartRedraw(0);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
RestoreOne	(keine Treffer im Projekt)

DB_GetNextPosNo [Persistenz (DB/Datei)]

```
int DB_GetNextPosNo(const string symbol, ENUM_TIMEFRAMES tf, const string direction, const int tra
```

Ort im Code: db_state.mqh Zeile 605 bis 640

Rueckgabe	Parameter (Typ name [default])
int	const string symbol ENUM_TIMEFRAMES tf const string direction const int trade_no

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LogErr, TF_ToString, DB_IsReady.
2. 2. NICHT cappen – der Caller muss ">" erkennen können.: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(rq == INVALID_HANDLE)
- if(DatabaseRead(rq))
- if(nextpos < 1)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
607: if(!DB_IsReady())
608: return 1;
617: if(rq == INVALID_HANDLE)
619: DB_LogErr(__FUNCTION__, "Prepare failed");
620: return 1;
629: if(DatabaseRead(rq))
636: if(nextpos < 1) nextpos = 1;
639: return nextpos;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_IsReady, DB_LogErr, TF_ToString	Trade Assistent V2.008.mq5:1080 int pos_no = DB_GetNextPosNo(_Symbol, (ENUM_TIMEFRAMES)_Period, dir, trade_no); discord_send.mqh:71 int pos_no = DB_GetNextPosNo(_Symbol, _Period, direction, trade_no);

GetObjectPriceAny [Persistenz (DB/Datei)]

```
double GetObjectPriceAny(const string name)
```

Ort im Code: db_state.mqh Zeile 643 bis 659

Rueckgabe	Parameter (Typ name [default])
double	const string name

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

2 2. Anchor 1 (für Objekte mit mehreren Punkten): Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, name) < 0)
- if(ObjectGetDouble(0, name, OBJPROP_PRICE, 0, price) && price > 0.0)
- if(ObjectGetDouble(0, name, OBJPROP_PRICE, 1, price) && price > 0.0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
645: if(ObjectFind(0, name) < 0)
646: return 0.0;
651: if(ObjectGetDouble(0, name, OBJPROP_PRICE, 0, price) && price > 0.0)
652: return price;
655: if(ObjectGetDouble(0, name, OBJPROP_PRICE, 1, price) && price > 0.0)
656: return price;
658: return 0.0;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	db_state.mqh:665 double p = GetObjectPriceAny(obj_name);

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_PRICE

SaveOne [Persistenz (DB/Datei)]

```
void SaveOne(const string key, const string obj_name)
```

Ort im Code: db_state.mqh Zeile 663 bis 668

Rueckgabe	Parameter (Typ name [default])
void	const string key const string obj_name

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_Key, DB_SetMetaText, GetObjectPriceAny.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(p > 0.0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
664: {
```

```
665: double p = GetObjectPriceAny(obj_name);
666: if(p > 0.0)
666: if(p > 0.0)
667: DB_SetMetaText(DB_Key(key), DoubleToString(p, _Digits));
667: DB_SetMetaText(DB_Key(key), DoubleToString(p, _Digits));
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_Key, DB_SetMetaText, GetObjectPriceAny	(keine Treffer im Projekt)

DB_DeletePositionKey [Persistenz (DB/Datei)]

```
bool DB_DeletePositionKey(const string symbol,
                          const string tf_txt,
                          const string direction,
                          const int trade_no,
                          const int pos_no)
```

Ort im Code: db_state.mqh Zeile 675 bis 705

Kommentar direkt darueber: Low-Level (tf bereits als Text wie "M5","H1", ...)

Rueckgabe	Parameter (Typ name [default])
bool	const string symbol const string tf_txt const string direction const int trade_no const int pos_no

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LogErr, DB_IsReady, DB_ExecPreparedNonQuery.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(rq == INVALID_HANDLE)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
681: if(!DB_IsReady())
682: return false;
689: if(rq == INVALID_HANDLE)
691: DB_LogErr(__FUNCTION__, "Prepare failed");
692: return false;
701: bool ok = DB_ExecPreparedNonQuery(rq);
704: return ok;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_ExecPreparedNonQuery, DB_IsReady, DB_LogErr	(keine Treffer im Projekt)

DB_GetPosition [Persistenz (DB/Datei)]

```
bool DB_GetPosition(const string symbol,
                    ENUM_TIMEFRAMES tf,
                    const string direction,
                    const int trade_no,
                    const int pos_no,
                    DB_PositionRow &out_row)
```

Ort im Code: db_state.mqh Zeile 706 bis 798

Rueckgabe	Parameter (Typ name [default])
bool	const string symbol ENUM_TIMEFRAMES tf const string direction const int trade_no const int pos_no DB_PositionRow &out_row

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LogErr, TF_ToString, DB_IsReady.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(rq == INVALID_HANDLE)
- if(DatabaseRead(rq))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
730: if(!DB_IsReady())
731: return false;
746: if(rq == INVALID_HANDLE)
748: DB_LogErr(__FUNCTION__, "Prepare failed");
749: return false;
760: if(DatabaseRead(rq))
797: return found;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_IsReady, DB_LogErr, TF_ToString	event.mqh:130 if(DB_GetPosition(_Symbol, (ENUM_TIMEFRAMES)_Period, direction, trade_no, pos_no, row)) gui_elemente.mqh:148 if(DB_GetPosition(_Symbol, (ENUM_TIMEFRAMES)Period(), direction, trade_no, pos_no, row)) gui_elemente.mqh:196 if(DB_GetPosition(_Symbol, (ENUM_TIMEFRAMES)Period(), direction, trade_no, pos_no, row))

DB_UpdatePositionStatus [Persistenz (DB/Datei)]

```
bool DB_UpdatePositionStatus(const string symbol,
                             ENUM_TIMEFRAMES tf,
                             const string direction,
                             const int trade_no,
                             const int pos_no,
                             const string new_status,
                             const int new_is_pending)
```

Ort im Code: db_state.mqh Zeile 799 bis 845

Rueckgabe	Parameter (Typ name [default])

bool	const string symbol ENUM_TIMEFRAMES tf const string direction const int trade_no const int pos_no const string new_status const int new_is_pending

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LogErr, TF_ToString, DB_IsReady, DB_ExecPreparedNonQuery.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(pos_no <= 0)
- if(rq == INVALID_HANDLE)
- if(pos_no > 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
807: if(!DB_IsReady())
808: return false;
813: // Wenn pos_no <= 0, dann ALLE Positionen dieses Trades updaten (z.B. Cancel).
814: // Wenn pos_no > 0, dann nur genau diese Position updaten.
816: if(pos_no <= 0)
824: if(rq == INVALID_HANDLE)
826: DB_LogErr(__FUNCTION__, "Prepare failed");
827: return false;
838: if(pos_no > 0)
841: bool ok = DB_ExecPreparedNonQuery(rq);
844: return ok;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

DB_ExecPreparedNonQuery, DB_IsReady, DB_LogErr, TF_ToString	event.mqh:486 DB_UpdatePositionStatus(_Symbol, (ENUM_TIMEFRAMES)_Period,
	trades_panel.mqh:755 DB_UpdatePositionStatus(_Symbol, (ENUM_TIMEFRAMES)_Period, "LONG", active_long_trade_no, 0, "CLOSED_CANCEL", 0);
	trades_panel.mqh:803 DB_UpdatePositionStatus(_Symbol, (ENUM_TIMEFRAMES)_Period, "SHORT", active_short_trade_no, 0, "CLOSED_CANCEL", 0);

DB_DeletePosition [Persistenz (DB/Datei)]

```
bool DB_DeletePosition(const string symbol, ENUM_TIMEFRAMES tf, const string direction, const int
```

Ort im Code: db_state.mqh Zeile 857 bis 883

Kommentar direkt darueber: 2) db_state.mqh – DB_DeletePosition hinzufügen (wichtig für “Draft rollback”) Warum: DB_GetNextPosNo() nimmt MAX(pos_no) aus der DB. Wenn ein Discord-Send fehlschlägt und der Draft drin bleibt, ist Pos2 “verbrannt”. Daher: Draft bei Fehler löschen. 2.1 Funktion einfügen In db_state.mqh direkt nach DB_UpsertPosition(...) einfügen:

Rueckgabe	Parameter (Typ name [default])

bool	const string symbol ENUM_TIMEFRAMES tf const string direction const int trade_no const int pos_no

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LogErr, TF_ToString, DB_IsReady, DB_ExecPreparedNonQuery.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(rq == INVALID_HANDLE)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
859: if(!DB_IsReady())
860: return false;
867: if(rq == INVALID_HANDLE)
869: DB_LogErr(__FUNCTION__, "Prepare failed");
870: return false;
879: bool ok = DB_ExecPreparedNonQuery(rq);
882: return ok;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_ExecPreparedNonQuery, DB_IsReady, DB_LogErr, TF_ToString	Trade Assistent V2.008.mq5:1041 DB_DeletePosition(row.symbol, (ENUM_TIMEFRAMES)_Period, row.direction, row.trade_no, row.pos_no);

DB_GetPosNoCount [Persistenz (DB/Datei)]

```
int DB_GetPosNoCount(const string symbol, ENUM_TIMEFRAMES tf, const string direction, const int tr
```

Ort im Code: db_state.mqh Zeile 886 bis 917

Kommentar direkt darueber: Ermittle den Count der aktiven Positionen des Trades

Rueckgabe	Parameter (Typ name [default])
int	const string symbol ENUM_TIMEFRAMES tf const string direction const int trade_no

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LogErr, TF_ToString, DB_IsReady.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(rq == INVALID_HANDLE)
- if(DatabaseRead(rq))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
888: if(!DB_IsReady())
889: return 1;
898: if(rq == INVALID_HANDLE)
900: DB_LogErr(__FUNCTION__, "Prepare failed");
901: return 1;
910: if(DatabaseRead(rq))
916: return posnocount;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_IsReady, DB_LogErr, TF_ToString	discord_send.mqh:74 int pos_no_count = DB_GetPosNoCount(_Symbol, _Period, direction, trade_no);

discord.mqh

Discord-Integration: Message-Formatierung, Webhook-URL Handling

Funktionen in dieser Datei (15): checkDiscord, EscapeJSON, FormatTradeMessage, SendDiscordMessageTest, getPeriodText, SendDiscordMessage, FormatSLMessage, FormatCancelTradeMessage, FormatUpdateTradeMessage, get_discord_webhook, TF_Text, TF_Short, FormatTradeMessageRow, SendScreenShot, FormatLineMoveMessage

checkDiscord [Discord-Integration]

```
bool checkDiscord()
```

Ort im Code: discord.mqh Zeile 78 bis 158

Rueckgabe	Parameter (Typ name [default])
bool	(keine)

Seiteneffekte

- Kann Discord-Nachrichten versenden oder formatieren.

Ablauf (Schritt fuer Schritt)

1 1. Start: Discord-Webhook Nachricht senden/formatieren; Logging/Benachrichtigung. Wichtige Aufrufe: get_discord_webhook.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!TerminalInfoInteger(TERMINAL_TRADE_ALLOWED))
- if(res == -1)
- if(get_discord_webhook()==discord_webhook_test)
- if(res == -1)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
 80: Print("Initialization step 1: Checking WebRequest permissions...");
 83: Print("Error: WebRequest is not allowed. Please allow in Tool -> Options -> Expert Advisors
 86: Print("TerminalInfoInteger ok!");
 88: Print("Initialization step 2: Testing Discord connection...");
 98: int res = WebRequest(
111: Print("WebRequest failed. Error code: ", error);
112: Print("Make sure these URLs are allowed:");
113: Print("https://discord.com/*");
114: Print("https://discordapp.com/*");
118: Print("Initialization step 3: Set Webhook");
121: Print("Please set Webhooks: ");
126: Print("Use Webhook: "+get_discord_webhook());
129: isWebRequestEnabled = true;
130: Print("Initialization step 4: All checks passed!");
131: Print("Successfully connected to Discord!");
151: Print("WebRequest failed. Error code: ", error);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
get_discord_webhook	Trade Assistent V2.008.mq5:218 if(!checkDiscord())

EscapeJSON [Discord-Integration]

```
string EscapeJSON(string text)
```

Ort im Code: discord.mqh Zeile 163 bis 172

Kommentar direkt darueber: | Function to escape JSON string |

Rueckgabe	Parameter (Typ name [default])
string	string text

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
164: {
165: string escaped = text;
166: StringReplace(escaped, "\\", "\\\\");
167: StringReplace(escaped, "\"", "\\\"");
168: StringReplace(escaped, "\n", "\\n");
171: return escaped;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	discord.mqh:250 string payload = "{\"content\":\"" + EscapeJSON(message) + "\"}";

FormatTradeMessage [Discord-Integration]

```
string FormatTradeMessage(const DB_PositionRow &row)
```

Ort im Code: discord.mqh Zeile 177 bis 195

Rueckgabe	Parameter (Typ name [default])
string	const DB_PositionRow &row

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(dir == "LONG")

Wichtige Codezeilen (zum Mitlesen im Editor)

```
178: {
179: string message = "@everyone\n";
180: message += ":red_circle:TRADINGSIGNAL: :red_circle:\n\n";
181: message += StringFormat("----------[Trade Nr. %d | Pos %d]----------\n\n", row.trade_no, ro
184: if(dir == "LONG")
194: return message;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	discord_send.mqh:107 string message = FormatTradeMessage(row);

SendDiscordMessageTest [Discord-Integration]

```
bool SendDiscordMessageTest(string message, bool isError = false)
```

Ort im Code: discord.mqh Zeile 201 bis 209

Rueckgabe	Parameter (Typ name [default])
bool	string message bool isError = false

Seiteneffekte

- Kann Discord-Nachrichten versenden oder formatieren.

Ablauf (Schritt fuer Schritt)

1 1. Start: Discord-Webhook Nachricht senden/formatieren. Wichtige Aufrufe: SendDiscordMessage.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
202: {
204: string discord_webhook_save = discord_webhook;
205: discord_webhook =discord_webhook_test;
207: return SendDiscordMessage(message,  false);
207: return SendDiscordMessage(message,  false);
209: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
SendDiscordMessage	(keine Treffer im Projekt)

getPeriodText [Discord-Integration]

```
string getPeriodText()
```

Ort im Code: discord.mqh Zeile 214 bis 230

Rueckgabe	Parameter (Typ name [default])
string	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(sec <= 0)
- if(min < 60)
- if(min < 24*60)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
219: if(sec <= 0)
220: return EnumToString(tf);
223: if(min < 60)
224: return "M" + IntegerToString(min);
225: if(min < 24*60)
226: return "H" + IntegerToString(min/60);
227: return "D" + IntegerToString(min/(24*60));
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	discord.mqh:431 msg += "**Symbol:** " + row.symbol + " " + getPeriodText() + "\n";

SendDiscordMessage [Discord-Integration]

```
bool SendDiscordMessage(string message, bool isError = false)
```

Ort im Code: discord.mqh Zeile 235 bis 304

Rueckgabe	Parameter (Typ name [default])
bool	string message bool isError = false

Seiteneffekte

- Kann Discord-Nachrichten versenden oder formatieren.

Ablauf (Schritt fuer Schritt)

1 1. Start: Discord-Webhook Nachricht senden/formatieren; Logging/Benachrichtigung. Wichtige Aufrufe: get_discord_webhook, EscapeJSON.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!isWebRequestEnabled)
- if(res == 200 || res == 204)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
237: if(!isWebRequestEnabled)
239: Print("Not isWebRequestEnabled");
240: return false;
257: int res = WebRequest(
267: // Both 200 and 204 are success codes for Discord webhooks
268: if(res == 200 || res == 204)
273: return true;
299: Print("Discord Error: ", error, " (", res, ")");
300: Print("Message: ", message);
301: Print("Last MT5 Error: ", GetLastError());
303: return false;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

EscapeJSON, get_discord_webhook	Trade Assistent V2.008.mq5:668 SendDiscordMessage(FormatCloseSL_DB(p));
	Trade Assistent V2.008.mq5:685 SendDiscordMessage(FormatCloseSL_DB(p));
	discord.mqh:207 return SendDiscordMessage(message, false);
	discord_send.mqh:109 bool discord_ok = SendDiscordMessage(message);
	event.mqh:67 SendDiscordMessage(msg);
	event.mqh:483 SendDiscordMessage(message);
	trades_panel.mqh:752 bool ret = SendDiscordMessage(message);
	trades_panel.mqh:801 bool ret = SendDiscordMessage(message);

FormatSLMessage [Discord-Integration]

```
string FormatSLMessage(const DB_PositionRow &row)
```

Ort im Code: discord.mqh Zeile 310 bis 316

Rueckgabe	Parameter (Typ name [default])
string	const DB_PositionRow &row

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
311: {
312: string message = "@everyone\n";
313: message += StringFormat("**Note:** %s %s Trade %d Pos %d - SL erreicht\n",
314: row.symbol, row.tf, row.trade_no, row.pos_no);
315: return message;
315: return message;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	event.mqh:479 message = FormatSLMessage(r);

FormatCancelTradeMessage [Discord-Integration]

```
string FormatCancelTradeMessage(const DB_PositionRow &row)
```

Ort im Code: discord.mqh Zeile 323 bis 329

Rueckgabe	Parameter (Typ name [default])
string	const DB_PositionRow &row

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
324: {
325: string message = "@everyone\n";
326: message += StringFormat("**Attention:** %s %s Trade %d all Pos - Order canceln\n",
326: message += StringFormat("**Attention:** %s %s Trade %d all Pos - Order canceln\n",
327: row.symbol, row.tf, row.trade_no);
328: return message;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	event.mqh:474 message = FormatCancelTradeMessage(r);
	trades_panel.mqh:751 string message = FormatCancelTradeMessage(r);
	trades_panel.mqh:800 string message = FormatCancelTradeMessage(r);

FormatUpdateTradeMessage [Discord-Integration]

```
string FormatUpdateTradeMessage(const DB_PositionRow &row)
```

Ort im Code: discord.mqh Zeile 333 bis 341

Rueckgabe	Parameter (Typ name [default])
string	const DB_PositionRow &row

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
334: {
335: string message = "@everyone\n";
336: message += StringFormat("**Attention:** %s %s Trade %d Pos %d - SL -> %s (Sabio: %s) | TP -
337: row.symbol, row.tf, row.trade_no, row.pos_no,
338: DoubleToString(row.sl, _Digits), row.sabio_sl);
340: return message;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	(keine Treffer im Projekt)

get_discord_webhook [Discord-Integration]

```
string get_discord_webhook()
```

Ort im Code: discord.mqh Zeile 349 bis 379

Rueckgabe	Parameter (Typ name [default])
string	(keine)

Seiteneffekte

- Kann Discord-Nachrichten versenden oder formatieren.

Ablauf (Schritt fuer Schritt)

1 1. Start: Discord-Webhook Nachricht senden/formatieren

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(_Symbol == SYMBOL_NAME_EURUSD)
- if(_Symbol == SYMBOL_NAME_AUDUSD)
- if(_Symbol == SYMBOL_NAME_USDCAD)
- if(_Symbol == SYMBOL_NAME_USDCHF)
- if(_Symbol == SYMBOL_NAME_USDJPY)
- if(_Symbol == SYMBOL_NAME_EURJPY)
- if(_Symbol == SYMBOL_NAME_EURNZD)
- if(_Symbol == SYMBOL_NAME_XAUUSD)
- if(_Symbol == SYMBOL_NAME_WTI)
- if(_Symbol == SYMBOL_NAME_NASDAQ)
- if(_Symbol == SYMBOL_NAME_GPBUSD)
- if(_Symbol == SYMBOL_NAME_NZDUSD)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
352: if(_Symbol == SYMBOL_NAME_EURUSD)
353: return Webhook_EURUSD;
355: if(_Symbol == SYMBOL_NAME_AUDUSD)
356: return Webhook_AUDUSD;
357: if(_Symbol == SYMBOL_NAME_USDCAD)
358: return Webhook_USDCAD;
360: return Webhook_USDCHF;
```

```
362: return Webhook_USDJPY;
364: return Webhook_EURJPY;
366: return Webhook_EURNZD;
368: return Webhook_Gold;
370: return Webhook_WTI;
372: return Webhook_Nasdaq;
374: return Webhook_GPBUSD;
376: return Webhook_NZDUSD;
377: return Webhook_System;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	discord.mqh:119 if(get_discord_webhook()==discord_webhook_test) discord.mqh:126 Print("Use Webhook: "+get_discord_webhook()); discord.mqh:133 test_message = "{\"content\":\"DowHow Signal Dienst AS3 System für "+_Symbol+" mit Webhook "+get_discord_webhook() +"\"}"; discord.mqh:259 get_discord_webhook(), discord.mqh:514 res=WebRequest("POST", get_discord_webhook(),str,5000,data,data,str);

TF_Text [Discord-Integration]

```
string TF_Text()
```

Ort im Code: discord.mqh Zeile 391 bis 395

Rueckgabe	Parameter (Typ name [default])
string	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
392: {
393: // liefert z.B. "PERIOD_H1"
394: return EnumToString((ENUM_TIMEFRAMES)_Period);
394: return EnumToString((ENUM_TIMEFRAMES)_Period);
395: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	discord.mqh:421 return TF_Text();

TF_Short [Discord-Integration]

```
string TF_Short()
```

Ort im Code: discord.mqh Zeile 398 bis 423

Kommentar direkt darueber: optional hübscher: "H1" statt "PERIOD_H1"

Rueckgabe	Parameter (Typ name [default])
string	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: TF_Text.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
403: return "M1";
405: return "M5";
407: return "M15";
409: return "M30";
411: return "H1";
413: return "H4";
415: return "D1";
417: return "W1";
419: return "MN1";
421: return TF_Text();
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
TF_Text	(keine Treffer im Projekt)

FormatTradeMessageRow [Discord-Integration]

```
string FormatTradeMessageRow(const DB_PositionRow &row, const string order_type /*BUY/SELL*/)
```

Ort im Code: discord.mqh Zeile 426 bis 436

Rueckgabe	Parameter (Typ name [default])
string	const DB_PositionRow &row const string order_type /*BUY/SELL*/

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: getPeriodText.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
427: {
428: string msg="@everyone\n:red_circle:TRADINGSIGNAL: :red_circle:\n\n";
429: msg += "----------[Trade Nr. " + IntegerToString(row.trade_no) + " | Pos " + IntegerToStrin
430: msg += (order_type=="BUY" ? ":chart_with_upwards_trend: **BUY:** " : ":chart_with_downwards_
431: msg += "**Symbol:** " + row.symbol + " " + getPeriodText() + "\n";
435: return msg;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
getPeriodText	(keine Treffer im Projekt)

SendScreenShot [Discord-Integration]

```
void SendScreenShot(string symbol,int _period, int ScreenWidth = 1912, int ScreenHeight = 1080)
```

Ort im Code: discord.mqh Zeile 443 bis 529

Rueckgabe	Parameter (Typ name [default])
void	string symbol int _period int ScreenWidth = 1912 int ScreenHeight = 1080

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).
- Liest/schreibt Persistenz (DB/Dateien).
- Kann Discord-Nachrichten versenden oder formatieren.

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben

2 2. Abschnitt: Persistenz (DB/Datei) lesen/schreiben; Discord-Webhook Nachricht senden/formatieren; Logging/Benachrichtigung. Wichtige Aufrufe: get_discord_webhook.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int x=1; x<=20; x++)
- while(i<CHARTS_MAX)
- if(ChartSymbol(currChart)==symbol)
- if(currChart==-1)
- if(ChartScreenShot(0,filename,ScreenWidth,ScreenHeight))
- if(res<0)
- if(FileSize(res)==0)
- if(FileReadArray(res,file)!=FileSize(res))
- if(ArraySize(file)>0)
- if(res==NULL)
- if(res<0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
450: for(int x=1; x<=20; x++)
454: while(i<CHARTS_MAX)
456: if(ChartSymbol(currChart)==symbol)
458: ChartRedraw(currChart);
462: if(currChart==-1)
474: Print("ChartScreenShot Error: ",(string)GetLastError());
478: res=FileOpen(filename,FILE_READ|FILE_WRITE|FILE_BIN);
481: Print("File Open Error: "+filename+", Attempt: ",x);
488: Print("FileSize Error, Attempt: ",x);
494: if(FileReadArray(res,file)!=FileSize(res))
497: Print("File Read Error: "+filename);
498: return;
514: res=WebRequest("POST", get_discord_webhook(),str,5000,data,data,str);
516: Print("Server response: ",CharArrayToString(data));
519: Print("Error: ",GetLastError());
528: return;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
get_discord_webhook	discord_send.mqh:119 SendScreenShot(_Symbol, _Period, getChartWidthInPixels(), getChartHeightInPixels());

FormatLineMoveMessage [Discord-Integration]

```
string FormatLineMoveMessage(const DB_PositionRow &row,
                             const string kind,
                             const double old_price,
                             const double new_price)
```

Ort im Code: discord.mqh Zeile 537 bis 562

Rueckgabe	Parameter (Typ name [default])
string	const DB_PositionRow &row const string kind const double old_price const double new_price

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(kind == "entry")
- if(old_price > 0.0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
541: {
542: string what = kind;
543: if(kind == "entry") what = "Entry";
543: if(kind == "entry") what = "Entry";
552: if(old_price > 0.0)
561: return message;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	(keine Treffer im Projekt)

discord_send.mqh

Low-Level Versand (DiscordSend) via WebRequest

Funktionen in dieser Datei (1): DiscordSend

DiscordSend [Discord-Integration]

```
void DiscordSend()
```

Ort im Code: discord_send.mqh Zeile 14 bis 192

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).
- Kann Discord-Nachrichten versenden oder formatieren.

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung
2. 2. TradeNr aus Eingabefeld (TRNB): Interne Logik / Berechnung
3. 3. Basis-Validierung (Preis vs. Markt): Logging/Benachrichtigung
4. 4. Aktive Tradenummer der Richtung: Interne Logik / Berechnung
5. 5. TradeNo bestimmen (AUTO-KORREKTUR statt blockieren): Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: update_Text.
6. 6. PosNo bestimmen (nur bestätigte Positionen zählen; Draft zählt NICHT): Persistenz (DB/Datei) lesen/schreiben; Logging/Benachrichtigung. Wichtige Aufrufe: DB_GetPosNoCount, DB_GetNextPosNo.
7. 7. Draft in DB schreiben: Persistenz (DB/Datei) lesen/schreiben; Logging/Benachrichtigung. Wichtige Aufrufe: DB_UpsertPosition, TF_ToString.
8. 8. Discord senden: Discord-Webhook Nachricht senden/formatieren; Logging/Benachrichtigung. Wichtige Aufrufe: SendDiscordMessage, getChartWidthInPixels, RollbackDraftRow, FormatTradeMessage, getChartHeightInPixels, SendScreenShot.
9. 9. Commit: Position bestätigt: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: showActive_long, showCancel_long, DB_UpsertPosition, showCancel_short, showActive_short, update_Text, UI_TradesPanel_RebuildRows.
10. 10. Meta aktualisieren: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_Key, DB_SetMetaInt, UI_TradesPanel_RebuildRows.
11. ... (weitere Abschnitte: 2)

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ui_direction_is_long)
- if(Entry_Price <= CurrentAskPrice)
- if(Entry_Price >= CurrentBidPrice)
- if(active_trade_no > 0)
- if(trade_no != active_trade_no)
- if(trade_no <= last_trade_nummer)
- if(pos_no < 1)

- if(pos_no_count < 1)
- if(pos_no_count > DB_MAX_POS_PER_SIDE)
- if(!DB_UpsertPosition(row))
- if(!discord_ok)
- if(isLong)
- if(starting_new_trade && pos_no == 1 && trade_no > last_trade_nummer)
- if(starting_new_trade && pos_no == 1)
- if(isLong)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
 30: MessageBox("Entry ist tiefer als aktueller Ask.", NULL, MB_OK);
 38: MessageBox("Entry ist höher als aktueller Bid.", NULL, MB_OK);
 70: // --- PosNo bestimmen (nur bestätigte Positionen zählen; Draft zählt NICHT)
 71: int pos_no = DB_GetNextPosNo(_Symbol, _Period, direction, trade_no);
 74: int pos_no_count = DB_GetPosNoCount(_Symbol, _Period, direction, trade_no);
 77: if(pos_no_count > DB_MAX_POS_PER_SIDE)
 79: MessageBox("Maximale Pyramide erreicht (Pos4).", NULL, MB_OK);
 84: DB_PositionRow row;
100: if(!DB_UpsertPosition(row))
102: MessageBox("DB Fehler: Position konnte nicht gespeichert werden.", NULL, MB_OK);
106: // --- Discord senden
109: bool discord_ok = SendDiscordMessage(message);
113: Print("Discord Fehler (Draft bleibt in DB, PosNo bleibt wiederverwendbar).");
121: // --- Commit: Position bestätigt
127: DB_UpsertPosition(row);
134: ObjectSetInteger(0,TP_BTN_ACTIVE_LONG,OBJPROP_BGCOLOR,clrBlack);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_GetNextPosNo, DB_GetPosNoCount, DB_Key, DB_SetMetaInt, DB_UpsertPosition, FormatTradeMessage, RollbackDraftRow, SendDiscordMessage, SendScreenShot, TF_ToString, UI_TradesPanel_RebuildRows, UI_UpdateNextTradePosUI, getChartHeightInPixels, getChartWidthInPixels, showActive_long, showActive_short, showCancel_long, showCancel_short, update_Text	event.mqh:392 DiscordSend(); event.mqh:397 DiscordSend();

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_BGCOLOR, OBJPROP_BORDER_COLOR, OBJPROP_TEXT

event.mqh

Eventhandling (OnChartEvent), Reaktion auf Button-Klicks und Linienbewegung

Funktionen in dieser Datei (4): TP_FinalizeLineMove, OnChartEvent, UI_TradeHasAnyPendingPosition, UI_CloseOnePositionAndNotify

TP_FinalizeLineMove [Trades-Panel (UI links)]

```
void TP_FinalizeLineMove()
```

Ort im Code: event.mqh Zeile 25 bis 80

Kommentar direkt darueber: Finalisiert eine TradePos-Linienverschiebung: Tag updaten, speichern, Discord senden

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).
- Liest/schreibt Persistenz (DB/Dateien).
- Kann Discord-Nachrichten versenden oder formatieren.

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung
2. 2. Objekt muss existieren: Interne Logik / Berechnung
3. 3. Tag sauber nachziehen: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: UI_CreateOrUpdateLineTag.
4. 4. persistieren (erst nach dem old/new Vergleich): Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_SaveLinePrices.
5. 5. nur melden, wenn sich der Preis wirklich geändert hat: Discord-Webhook Nachricht senden/formatieren. Wichtige Aufrufe: SendDiscordMessage, TF_ToString.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!g_tp_drag_active || g_tp_drag_name == "")
- if(ObjectFind(0, g_tp_drag_name) < 0)
- if(changed && (g_tp_drag_kind == "entry" || g_tp_drag_kind == "sl"))
- if(old_price > 0.0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
27: if(!g_tp_drag_active || g_tp_drag_name == "")
28: return;
31: if(ObjectFind(0, g_tp_drag_name) < 0)
35: return;
43: ChartRedraw(0);
46: DB_SaveLinePrices();
48: // Discord: nur melden, wenn sich der Preis wirklich geändert hat
52: if(changed && (g_tp_drag_kind == "entry" || g_tp_drag_kind == "sl"))
59: if(old_price > 0.0)
67: SendDiscordMessage(msg);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

DB_SaveLinePrices, SendDiscordMessage, TF_ToString, UI_CreateOrUpdateLineTag	event.mqh:170 TP_FinalizeLineMove(); event.mqh:205 TP_FinalizeLineMove();

OnChartEvent [EA-Lifecycle / Eventhandler]

```
      void OnChartEvent(const int id,         // Identifikator des Ereignisses
                        const long &lparam,   // Parameter des Ereignisses des Typs long, X cordinates
                        const double &dparam, // Parameter des Ereignisses des Typs double, Y cordinates
                        const string &sparam) // Parameter des Ereignisses des Typs string, name of the
```

Ort im Code: event.mqh Zeile 86 bis 425

Rueckgabe	Parameter (Typ name [default])
	(keine)

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).
- Liest/schreibt Persistenz (DB/Dateien).
- Kann Discord-Nachrichten versenden oder formatieren.

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_TradesPanel_OnChartEvent.
2. 2. Trade-Pos-Linien: Tag live nachziehen (DRAG), Discord/DB genau 1x pro Drag (Finalize): UI/Chart-Objekte anlegen/aktualisieren/verschieben; Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: UI_ParseTradePosFromName, UI_CreateOrUpdateLineTag, DB_GetPosition, UI_IsTradePosLine.
3. 3. finalize sofort (Discord + DB): Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: TP_FinalizeLineMove.
4. 4. nur speichern/Tag: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: UI_CreateOrUpdateLineTag, DB_SaveLinePrices, Get_Price_d, UI_IsTradePosLine.
5. 5. Wenn MT5 kein CHARTEVENT_OBJECT_CHANGE feuert,: UI/Chart-Objekte anlegen/aktualisieren/verschieben; Persistenz (DB/Datei) lesen/schreiben; Discord-Webhook Nachricht senden/formatieren; Logging/Benachrichtigung. Wichtige Aufrufe: TP_FinalizeLineMove, Get_Price_s, calcLots, DB_SaveLinePrices, Get_Price_d, update_Text.
6. 6. \Program Files\IC Markets (SC) Demo 51680033\Sounds\Alert2.wav);: Discord-Webhook Nachricht senden/formatieren. Wichtige Aufrufe: UI_UpdateAllLineTags, DiscordSend, UI_UpdateNextTradePosUI, UpdateSabioTP.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(UI_TradesPanel_OnChartEvent(id, lparam, dparam, sparam))
- if(id == CHARTEVENT_OBJECT_DRAG)
- if(UI_ParseTradePosFromName(sparam, direction, trade_no, pos_no, kind) && (kind == "entry" || kind == "sl"))
- if(!g_tp_drag_active || g_tp_drag_name != sparam)
- if(DB_GetPosition(_Symbol, (ENUM_TIMEFRAMES)_Period, direction, trade_no, pos_no, row))
- if(g_tp_drag_old <= 0.0)
- if(now - last_redraw > 50)
- if(UI_IsTradePosLine(sparam))
- if(id == CHARTEVENT_OBJECT_CHANGE)
- if(g_tp_drag_active && sparam == g_tp_drag_name)

- if(sparam == PR_HL || sparam == SL_HL || UI_IsTradePosLine(sparam))
- if(UI_IsTradePosLine(sparam))
- if(id == CHARTEVENT_MOUSE_MOVE)
- if(MouseState == 0 && g_tp_drag_active)
- if(prevMouseState == 0 && MouseState == 1)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
104: // --- Trade-Pos-Linien: Tag live nachziehen (DRAG), Discord/DB genau 1x pro Drag (Finalize
129: DB_PositionRow row;
130: if(DB_GetPosition(_Symbol, (ENUM_TIMEFRAMES)_Period, direction, trade_no, pos_no, row))
149: ChartRedraw(0);
166: // Wenn MT5 CHANGE liefert: finalize sofort (Discord + DB)
180: DB_SaveLinePrices();
203: // senden wir Discord + speichern beim MouseUp (state==0).
247: ObjectSetInteger(0, SLButton, OBJPROP_YDISTANCE, mlbDownYD_R5 + MouseD_Y - mlbDownY5);
248: ObjectSetInteger(0, SabioSL, OBJPROP_YDISTANCE, mlbDownYD_R5 + MouseD_Y + 30 - mlbDownY5);
256: ObjectSetInteger(0, SL_HL, OBJPROP_TIME, dt_SL);
257: ObjectSetDouble(0, SL_HL, OBJPROP_PRICE, price_SL);
295: ChartRedraw(0);
301: ObjectSetInteger(0, EntryButton, OBJPROP_YDISTANCE, mlbDownYD_R3 + MouseD_Y - mlbDownY3);
303: ObjectSetInteger(0, SLButton, OBJPROP_YDISTANCE, mlbDownYD_R5 + MouseD_Y - mlbDownY5);
304: ObjectSetInteger(0, SENDTRADEBTN, OBJPROP_YDISTANCE, mlbDownYD_R3 + MouseD_Y - mlbDownY3);
389: //            MessageBoxSound = PlaySound(C:\Program Files\IC Markets (SC) Demo 51680033\So
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_GetPosition, DB_SaveLinePrices, DiscordSend, Get_Price_d, Get_Price_s, TP_FinalizeLineMove, UI_CreateOrUpdateLineTag, UI_IsTradePosLine, UI_ParseTradePosFromName, UI_TradesPanel_OnChartEvent, UI_UpdateAllLineTags, UI_UpdateNextTradePosUI, UpdateSabioTP, calcLots, update_Text	(keine Treffer im Projekt)

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_PRICE, OBJPROP_STATE, OBJPROP_TIME, OBJPROP_XDISTANCE, OBJPROP_XSIZE, OBJPROP_YDISTANCE, OBJPROP_YSIZE
- Chart-Properties: CHART_MOUSE_SCROLL

UI_TradeHasAnyPendingPosition [UI / Chart-Objekte]

```
bool UI_TradeHasAnyPendingPosition(const string direction, const int trade_no)
```

Ort im Code: event.mqh Zeile 432 bis 454

Kommentar direkt darueber: Prüft, ob innerhalb einer Trade-Nummer (und Richtung) noch irgendeine pending Position existiert. (falls nein -> Trade ist "zu" und darf nicht mehr als aktiv gelten)

Rueckgabe	Parameter (Typ name [default])
bool	const string direction const int trade_no

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LoadPositions.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int i = 0; i < n; i++)
- if(rows[i].direction != direction)
- if(rows[i].trade_no != trade_no)
- if(rows[i].was_sent != 1)
- if(rows[i].is_pending != 1)
- if(StringFind(rows[i].status, "CLOSED", 0) == 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
434: DB_PositionRow rows[];
435: int n = DB_LoadPositions(_Symbol, _Period, rows);
437: for(int i = 0; i < n; i++)
439: if(rows[i].direction != direction)   continue;
440: if(rows[i].trade_no   != trade_no)   continue;
442: // Nur echte/gesendete Positionen berücksichtigen
443: if(rows[i].was_sent   != 1)          continue;
446: if(rows[i].is_pending != 1)          continue;
449: if(StringFind(rows[i].status, "CLOSED", 0) == 0) continue;
451: return true;
453: return false;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_LoadPositions	event.mqh:524 if(!UI_TradeHasAnyPendingPosition(direction, trade_no))

UI_CloseOnePositionAndNotify [UI / Chart-Objekte]

```
void UI_CloseOnePositionAndNotify(const string action,
                                  const string direction,
                                  const int trade_no,
                                  const int pos_no)
```

Ort im Code: event.mqh Zeile 456 bis 567

Rueckgabe	Parameter (Typ name [default])
void	const string action const string direction const int trade_no const int pos_no

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).
- Liest/schreibt Persistenz (DB/Dateien).
- Kann Discord-Nachrichten versenden oder formatieren.

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung
2. 2. Discord: Persistenz (DB/Datei) lesen/schreiben; Discord-Webhook Nachricht senden/formatieren. Wichtige Aufrufe: SendDiscordMessage, FormatCancelTradeMessage, TF_ToString, FormatSLMessage.
3. 3. DB: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_UpdatePositionStatus.
4. 4. Linien/Labels dieser Position entfernen (falls vorhanden): UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: UI_UpdateAllLineTags.

5 5. Falls das die letzte pending Position des Trades war -> Runtime + Meta zurücksetzen: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_Key, DB_SetMetaInt, UI_TradeHasAnyPendingPosition.

6 6. UI Refresh: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: UI_UpdateNextTradePosUI.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(action == "CANCEL")
- if(direction == "LONG")
- if(!UI_TradeHasAnyPendingPosition(direction, trade_no))
- if(direction == "LONG")
- if(active_long_trade_no == trade_no)
- if(ObjectFind(0, "ActiveLongTrade") >= 0)
- if(active_short_trade_no == trade_no)
- if(ObjectFind(0, "ActiveShortTrade") >= 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
461: // 1) Discord
462: DB_PositionRow r;
483: SendDiscordMessage(message);
486: DB_UpdatePositionStatus(_Symbol, (ENUM_TIMEFRAMES)_Period,
490: // 3) Linien/Labels dieser Position entfernen (falls vorhanden)
491: // 3) Linien/Labels dieser Position entfernen (falls vorhanden)
498: ObjectDelete(0, Entry_Long + suf_tp);
499: ObjectDelete(0, SL_Long    + suf_tp);
500: ObjectDelete(0, LabelEntryLong + suf_tp);
501: ObjectDelete(0, LabelSLLong    + suf_tp);
503: ObjectDelete(0, Entry_Long + suf_p);
504: ObjectDelete(0, SL_Long    + suf_p);
505: ObjectDelete(0, LabelEntryLong + suf_p);
506: ObjectDelete(0, LabelSLLong    + suf_p);
510: ObjectDelete(0, Entry_Short + suf_tp);
524: if(!UI_TradeHasAnyPendingPosition(direction, trade_no))
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

DB_Key, DB_SetMetaInt, DB_UpdatePositionStatus, FormatCancelTradeMessage, FormatSLMessage, SendDiscordMessage, TF_ToString, UI_TradeHasAnyPendingPosition, UI_UpdateAllLineTags, UI_UpdateNextTradePosUI	trades_panel.mqh:844 UI_CloseOnePositionAndNotify("CANCEL", "LONG", trade_no, pos_no);
	trades_panel.mqh:859 UI_CloseOnePositionAndNotify("SL", "LONG", trade_no, pos_no);
	trades_panel.mqh:874 UI_CloseOnePositionAndNotify("CANCEL", "SHORT", trade_no, pos_no);
	trades_panel.mqh:889 UI_CloseOnePositionAndNotify("SL", "SHORT", trade_no, pos_no);

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_BGCOLOR, OBJPROP_COLOR

gui_elemente.mqh

UI-Helfer: Linien/Buttons/Labels erzeugen, Layout-Funktionen, Update der Texte

Funktionen in dieser Datei (25): getChartHeightInPixels, getChartWidthInPixels, update_Text, UI_ParseTradePosFromName, DB_SaveLinePrices, DB_SaveOneLinePrice, CreateEntryAndSLLines, CreateLabelsTPcolor_SLLines, CreateLabelsLong, CreateLabelsShort, DeleteLinesandLabelsShort, DeleteLinesandLabelsLong, UI_DeleteAllTradePosLinesByScan, createButton, createHL, SendButton, del_SENDBTN_TRNB_POSNB, SabioEdit, del_sabiosl_sabioEntry, UI_GetOverviewTopY, UI_PositionOverviewPanel, UI_TradeLists_TopY, UI_TradeLists_Height, UI_TradeLists_Deinit, UI_TradeLists_AutoRefresh

getChartHeightInPixels [UI / Chart-Objekte]

```
int getChartHeightInPixels(const long chartID = 0, const int subwindow = 0)
```

Ort im Code: gui_elemente.mqh Zeile 28 bis 42

Kommentar direkt darueber: | Die Funktion erhält den Wert der Höhe des Charts in Pixeln |

Rueckgabe	Parameter (Typ name [default])
int	const long chartID = 0 const int subwindow = 0

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung
2. 2. Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten: Interne Logik / Berechnung
3. 3. Setzen den Wert des Fehlers zurück: Interne Logik / Berechnung
4. 4. Erhalten wir den Wert der Eigenschaft: Interne Logik / Berechnung
5. 5. Schreiben die Fehlermeldung in den Log "Experten": Logging/Benachrichtigung
6. 6. Geben den Wert der Eigenschaft zurück: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!ChartGetInteger(chartID, CHART_HEIGHT_IN_PIXELS, 0, result))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
29: {
30: //--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
31: long result = -1;
35: if (!ChartGetInteger(chartID, CHART_HEIGHT_IN_PIXELS, 0, result))
38: Print(__FUNCTION__ + ", Error Code = ", GetLastError());
41: return ((int)result);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	Trade Assistent V2.008.mq5:1144 yd3 = getChartHeightInPixels()/2; discord_send.mqh:119 SendScreenShot(_Symbol, _Period, getChartWidthInPixels(), getChartHeightInPixels());

Verwendete MQL-Konstanten (UI)

- Chart-Properties: CHART_HEIGHT_IN_PIXELS

getChartWidthInPixels [UI / Chart-Objekte]

```
int getChartWidthInPixels(const long chart_ID = 0)
```

Ort im Code: gui_elemente.mqh Zeile 48 bis 62

Kommentar direkt darueber: | Die Funktion erhält den Wert der Breite des Charts in Pixeln |

Rueckgabe	Parameter (Typ name [default])
int	const long chart_ID = 0

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung
2. 2. Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten: Interne Logik / Berechnung
3. 3. Setzen den Wert des Fehlers zurück: Interne Logik / Berechnung
4. 4. Erhalten wir den Wert der Eigenschaft: Interne Logik / Berechnung
5. 5. Schreiben die Fehlermeldung in den Log "Experten": Logging/Benachrichtigung
6. 6. Geben den Wert der Eigenschaft zurück: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!ChartGetInteger(chart_ID, CHART_WIDTH_IN_PIXELS, 0, result))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
49: {
50: //--- Bereiten wir eine Variable, um den Wert der Eigenschaft zu erhalten
51: long result = -1;
55: if (!ChartGetInteger(chart_ID, CHART_WIDTH_IN_PIXELS, 0, result))
58: Print(__FUNCTION__ + ", Error Code = ", GetLastError());
61: return ((int)result);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	Trade Assistent V2.008.mq5:1143 xd3 = getChartWidthInPixels() -DistancefromRight-10; discord_send.mqh:119 SendScreenShot(_Symbol, _Period, getChartWidthInPixels(), getChartHeightInPixels());

Verwendete MQL-Konstanten (UI)

- Chart-Properties: CHART_WIDTH_IN_PIXELS

update_Text [UI / Chart-Objekte]

```
string update_Text(string name, string val)
```

Ort im Code: gui_elemente.mqh Zeile 67 bis 70

Rueckgabe	Parameter (Typ name [default])
string	string name string val

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Wichtige Codezeilen (zum Mitlesen im Editor)

```
68: {
69: return (string)ObjectSetString(0, name, OBJPROP_TEXT, val);
69: return (string)ObjectSetString(0, name, OBJPROP_TEXT, val);
70: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	Trade Assistent V2.008.mq5:251 update_Text(TRNB, last_trade_nummer);
	Trade Assistent V2.008.mq5:1063 update_Text(obj, IntegerToString(v));
	Trade Assistent V2.008.mq5:1205 update_Text(EntryButton, "Buy Stop @ " + Get_Price_s(PR_HL) + " | Lot: " + DoubleToString(NormalizeDouble(calcLots(SL_Price - Entry_Price), 2), 2));
	Trade Assistent V2.008.mq5:1206 update_Text(SLButton, "SL: " + DoubleToString(((Get_Price_d(PR_HL) - Get_Price_d(SL_HL)) / _Point), 0) + " Points | " + Get_Price_s(SL_HL));
	discord_send.mqh:55 update_Text(TRNB, IntegerToString(trade_no)); // UI korrigieren
	discord_send.mqh:64 update_Text(TRNB, IntegerToString(trade_no)); // UI korrigieren
	discord_send.mqh:136 update_Text(TP_BTN_ACTIVE_LONG, "ACTIVE POSITION");
	discord_send.mqh:145 update_Text(TP_BTN_ACTIVE_SHORT, "ACTIVE POSITION");
	... (23 weitere)

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_TEXT

UI_ParseTradePosFromName [UI / Chart-Objekte]

```
bool UI_ParseTradePosFromName(const string name, string &direction, int &trade_no, int &pos_no, st
```

Ort im Code: gui_elemente.mqh Zeile 76 bis 98

Kommentar direkt darueber: direction: "LONG"/"SHORT" kind: "entry"/"sl"/"tp"

Rueckgabe	Parameter (Typ name [default])
bool	const string name string &direction int &trade_no int &pos_no string &kind

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(StringFind(name, "Entry_Long_") == 0)
- if(n < 2)
- if(trade_no <= 0 || pos_no <= 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
77: {
78: direction = ""; kind = "";
81: if(StringFind(name, "Entry_Long_") == 0) { direction="LONG";  kind="entry"; }
91: if(n < 2) return false;
96: if(trade_no <= 0 || pos_no <= 0) return false;
97: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	event.mqh:111 if(UI_ParseTradePosFromName(sparam, direction, trade_no, pos_no, kind) && (kind == "entry" || kind == "sl")) gui_elemente.mqh:142 if(!UI_ParseTradePosFromName(name, direction, trade_no, pos_no, kind)) gui_elemente.mqh:190 if(!UI_ParseTradePosFromName(name, direction, trade_no, pos_no, kind))

DB_SaveLinePrices [Persistenz (DB/Datei)]

```
void DB_SaveLinePrices()
```

Ort im Code: gui_elemente.mqh Zeile 101 bis 158

Kommentar direkt darueber: ================== PERSIST / RESTORE LINE PRICES (SQLite Meta) ==================

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_IsReady.
2. 2. Basislinien wie bisher: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_Key, DB_SetMetaText.
3. 3. Alle Trade-HLines mitspeichern + (für Entry/SL) positions updaten: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_Key, DB_SetMetaText, UI_ParseTradePosFromName, DB_UpsertPosition, DB_GetPosition, UI_IsTradePosLine.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(ObjectFind(0, PR_HL) >= 0)
- if(ObjectFind(0, SL_HL) >= 0)
- for(int i = 0; i < total; i++)
- if(!UI_IsTradePosLine(name))
- if((ENUM_OBJECT)ObjectGetInteger(0, name, OBJPROP_TYPE) != OBJ_HLINE)

- if(!UI_ParseTradePosFromName(name, direction, trade_no, pos_no, kind))
- if(kind == "entry" || kind == "sl")
- if(DB_GetPosition(_Symbol, (ENUM_TIMEFRAMES)Period(), direction, trade_no, pos_no, row))
- if(kind == "entry")

Wichtige Codezeilen (zum Mitlesen im Editor)

```
103: if(!DB_IsReady())
104: return;
109: if(ObjectFind(0, PR_HL) >= 0)
112: DB_SetMetaText(DB_Key("price_entry"), DoubleToString(p, _Digits));
115: if(ObjectFind(0, SL_HL) >= 0)
118: DB_SetMetaText(DB_Key("price_sl"), DoubleToString(p, _Digits));
123: for(int i = 0; i < total; i++)
126: if(!UI_IsTradePosLine(name))
130: if((ENUM_OBJECT)ObjectGetInteger(0, name, OBJPROP_TYPE) != OBJ_HLINE)
136: DB_SetMetaText(DB_Key("hline|" + name), DoubleToString(price, _Digits));
138: // 2b) Positions-Tabelle: Entry/SL sauber persistieren (damit RestoreTradeLines_All stimmt)
142: if(!UI_ParseTradePosFromName(name, direction, trade_no, pos_no, kind))
145: if(kind == "entry" || kind == "sl")
147: DB_PositionRow row;
148: if(DB_GetPosition(_Symbol, (ENUM_TIMEFRAMES)Period(), direction, trade_no, pos_no, row))
154: DB_UpsertPosition(row);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_GetPosition, DB_IsReady, DB_Key, DB_SetMetaText, DB_UpsertPosition, UI_IsTradePosLine, UI_ParseTradePosFromName	Trade Assistent V2.008.mq5:754 DB_SaveLinePrices(); event.mqh:46 DB_SaveLinePrices(); event.mqh:180 DB_SaveLinePrices(); event.mqh:369 DB_SaveLinePrices();

Verwendete MQL-Konstanten (UI)

- Objekt-Typen: OBJ_HLINE
- Objekt-Properties: OBJPROP_PRICE, OBJPROP_TYPE

DB_SaveOneLinePrice [Persistenz (DB/Datei)]

```
void DB_SaveOneLinePrice(const string name)
```

Ort im Code: gui_elemente.mqh Zeile 160 bis 205

Rueckgabe	Parameter (Typ name [default])
void	const string name

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_Key, DB_SetMetaText, UI_ParseTradePosFromName, DB_UpsertPosition, DB_IsReady, DB_GetPosition, UI_IsTradePosLine.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!DB_IsReady())
- if(ObjectFind(0, name) < 0)
- if((ENUM_OBJECT)ObjectGetInteger(0, name, OBJPROP_TYPE) != OBJ_HLINE)
- if(name == PR_HL)

- if(name == SL_HL)
- if(!UI_IsTradePosLine(name))
- if(!UI_ParseTradePosFromName(name, direction, trade_no, pos_no, kind))
- if(kind == "entry" || kind == "sl")
- if(DB_GetPosition(_Symbol, (ENUM_TIMEFRAMES)Period(), direction, trade_no, pos_no, row))
- if(kind == "entry")

Wichtige Codezeilen (zum Mitlesen im Editor)

```
162: if(!DB_IsReady()) return;
163: if(ObjectFind(0, name) < 0) return;
165: if((ENUM_OBJECT)ObjectGetInteger(0, name, OBJPROP_TYPE) != OBJ_HLINE)
166: return;
171: if(name == PR_HL)
173: DB_SetMetaText(DB_Key("price_entry"), DoubleToString(price, _Digits));
174: return;
176: if(name == SL_HL)
178: DB_SetMetaText(DB_Key("price_sl"), DoubleToString(price, _Digits));
179: return;
184: return;
186: DB_SetMetaText(DB_Key("hline|" + name), DoubleToString(price, _Digits));
191: return;
195: DB_PositionRow row;
196: if(DB_GetPosition(_Symbol, (ENUM_TIMEFRAMES)Period(), direction, trade_no, pos_no, row))
202: DB_UpsertPosition(row);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_GetPosition, DB_IsReady, DB_Key, DB_SetMetaText, DB_UpsertPosition, UI_IsTradePosLine, UI_ParseTradePosFromName	(keine Treffer im Projekt)

Verwendete MQL-Konstanten (UI)

- Objekt-Typen: OBJ_HLINE
- Objekt-Properties: OBJPROP_PRICE, OBJPROP_TYPE

CreateEntryAndSLLines [UI / Chart-Objekte]

```
bool CreateEntryAndSLLines(string objName, datetime time1, double price1, color clr)
```

Ort im Code: gui_elemente.mqh Zeile 211 bis 245

Rueckgabe	Parameter (Typ name [default])
bool	string objName datetime time1 double price1 color clr

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben; Logging/Benachrichtigung. Wichtige Aufrufe: UI_Reg_Add.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, objName) >= 0)
- if(!ObjectCreate(0, objName, OBJ_HLINE, 0, time1, price1))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
218: ObjectSetDouble(0, objName, OBJPROP_PRICE, price1);
219: ObjectSetInteger(0, objName, OBJPROP_COLOR, clr);
220: ObjectSetInteger(0, objName, OBJPROP_STYLE, STYLE_DASH);
221: ObjectSetInteger(0, objName, OBJPROP_BACK, false);
222: ObjectSetInteger(0, objName, OBJPROP_SELECTABLE, true);
223: ObjectSetInteger(0, objName, OBJPROP_SELECTED, false);
224: ChartRedraw(0);
230: if (!ObjectCreate(0, objName, OBJ_HLINE, 0, time1, price1))
232: Print(__FUNCTION__, ": Failed to create ", objName, " err=", GetLastError());
236: ObjectSetDouble(0, objName, OBJPROP_PRICE, price1);
237: ObjectSetInteger(0, objName, OBJPROP_COLOR, clr);
238: ObjectSetInteger(0, objName, OBJPROP_STYLE, STYLE_DASH);
239: ObjectSetInteger(0, objName, OBJPROP_BACK, false);
240: ObjectSetInteger(0, objName, OBJPROP_SELECTABLE, true);
241: ObjectSetInteger(0, objName, OBJPROP_SELECTED, false);
243: ChartRedraw(0);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

UI_Reg_Add	Trade Assistent V2.008.mq5:1010 CreateEntryAndSLLines(Entry_Long + suf, TimeCurrent(), entry_draw, TradeEntryLineLong);
	Trade Assistent V2.008.mq5:1011 CreateEntryAndSLLines(SL_Long + suf, TimeCurrent(), sl_draw, Tradecolor_SLLineLong);
	Trade Assistent V2.008.mq5:1023 CreateEntryAndSLLines(Entry_Short + suf, TimeCurrent(), entry_draw, TradeEntryLineShort);
	Trade Assistent V2.008.mq5:1024 CreateEntryAndSLLines(SL_Short + suf, TimeCurrent(), sl_draw, Tradecolor_SLLineShort);
	trades_panel.mqh:417 CreateEntryAndSLLines(Entry_Long + suf, TimeCurrent(), entry_draw, TradeEntryLineLong);
	trades_panel.mqh:418 CreateEntryAndSLLines(SL_Long + suf, TimeCurrent(), sl_draw, Tradecolor_SLLineLong);
	trades_panel.mqh:422 CreateEntryAndSLLines(Entry_Short + suf, TimeCurrent(), entry_draw, TradeEntryLineShort);
	trades_panel.mqh:423 CreateEntryAndSLLines(SL_Short + suf, TimeCurrent(), sl_draw, Tradecolor_SLLineShort);
	... (4 weitere)

Verwendete MQL-Konstanten (UI)

- Objekt-Typen: OBJ_HLINE
- Objekt-Properties: OBJPROP_BACK, OBJPROP_COLOR, OBJPROP_PRICE, OBJPROP_SELECTABLE, OBJPROP_SELECTED, OBJPROP_STYLE

CreateLabelsTPcolor_SLLines [UI / Chart-Objekte]

```
void CreateLabelsTPcolor_SLLines(string LABEL_NAME, string text, double price2, color clr1)
```

Ort im Code: gui_elemente.mqh Zeile 254 bis 281

Kommentar direkt darueber: | Create Line Labels

Rueckgabe	Parameter (Typ name [default])
void	string LABEL_NAME string text double price2 color clr1

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1. 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben; Logging/Benachrichtigung. Wichtige Aufrufe: UI_Reg_Add.
2. 2. Label-Position anpassen: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: update_Text.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, LABEL_NAME) < 0)
- if(!ObjectCreate(0, LABEL_NAME, OBJ_TEXT, 0, TimeCurrent(), price2))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
259: if (ObjectFind(0, LABEL_NAME) < 0)
261: if (!ObjectCreate(0, LABEL_NAME, OBJ_TEXT, 0, TimeCurrent(), price2))
263: Print(__FUNCTION__, ": Failed to create ", LABEL_NAME, " Error Code: ", GetLastError());
264: return; // raus bei Fehler
268: ObjectSetInteger(0, LABEL_NAME, OBJPROP_COLOR, clr1);
269: ObjectSetInteger(0, LABEL_NAME, OBJPROP_FONTSIZE, InpFontSize);
270: ObjectSetString(0, LABEL_NAME, OBJPROP_FONT, InpFont);
271: ObjectSetString(0, LABEL_NAME, OBJPROP_TEXT, " ");
275: // Falls sich der Preis geändert hat: Label-Position anpassen
276: ObjectMove(0, LABEL_NAME, 0, TimeCurrent(), price2);
279: // Kein ChartRedraw() hier – das ist auf Dauer zu teuer.
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_Add, update_Text	gui_elemente.mqh:289 CreateLabelsTPcolor_SLLines(LabelSLLong, "StoppLoss Long Trade", SL_Price, Tradecolor_SLLineLong); gui_elemente.mqh:290 CreateLabelsTPcolor_SLLines(LabelEntryLong, "Entry Price Long Trade", Entry_Price, TradeEntryLineLong); gui_elemente.mqh:302 CreateLabelsTPcolor_SLLines(LabelSLShort, "StoppLoss Short Trade", SL_Price, Tradecolor_SLLineShort); gui_elemente.mqh:303 CreateLabelsTPcolor_SLLines(LabelEntryShort, "Entry Price Short Trade", Entry_Price, TradeEntryLineShort);

Verwendete MQL-Konstanten (UI)

- Objekt-Typen: OBJ_TEXT
- Objekt-Properties: OBJPROP_COLOR, OBJPROP_FONT, OBJPROP_FONTSIZE, OBJPROP_TEXT

CreateLabelsLong [UI / Chart-Objekte]

```
void CreateLabelsLong()
```

Ort im Code: gui_elemente.mqh Zeile 286 bis 294

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: CreateLabelsTPcolor_SLLines, update_Text.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
287: {
289: CreateLabelsTPcolor_SLLines(LabelSLLong, "StoppLoss Long Trade", SL_Price, Tradecolor_SLLin
290: CreateLabelsTPcolor_SLLines(LabelEntryLong, "Entry Price Long Trade", Entry_Price, TradeEnt
292: update_Text(LabelSLLong, "SL Long Trade");
293: update_Text(LabelEntryLong, "Entry Long Trade");
294: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
CreateLabelsTPcolor_SLLines, update_Text	Trade Assistent V2.008.mq5:348 CreateLabelsLong();

CreateLabelsShort [UI / Chart-Objekte]

```
void CreateLabelsShort()
```

Ort im Code: gui_elemente.mqh Zeile 299 bis 307

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: CreateLabelsTPcolor_SLLines, update_Text.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
300: {
302: CreateLabelsTPcolor_SLLines(LabelSLShort, "StoppLoss Short Trade", SL_Price, Tradecolor_SLL
303: CreateLabelsTPcolor_SLLines(LabelEntryShort, "Entry Price Short Trade", Entry_Price, TradeE
305: update_Text(LabelSLShort, "SL Short Trade");
306: update_Text(LabelEntryShort, "Entry Short Trade");
307: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
CreateLabelsTPcolor_SLLines, update_Text	Trade Assistent V2.008.mq5:353 CreateLabelsShort();

DeleteLinesandLabelsShort [UI / Chart-Objekte]

```
void DeleteLinesandLabelsShort()
```

Ort im Code: gui_elemente.mqh Zeile 312 bis 329

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_Reg_DeleteOne.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int i = total - 1; i >= 0; i--)
- for(int p = 0; p < ArraySize(prefixes); p++)
- if(StringFind(name, prefixes[p]) == 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
313: {
314: // löscht alle Objekte, die zu SHORT-Trade-Linien/Labels gehören (inkl. _1.._4)
315: string prefixes[] = {LabelSLShort, Entry_Short, LabelSLShort, LabelEntryShort};
317: for (int i = total - 1; i >= 0; i--)
320: for (int p = 0; p < ArraySize(prefixes); p++)
322: if (StringFind(name, prefixes[p]) == 0)
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_DeleteOne	trades_panel.mqh:820 DeleteLinesandLabelsShort();

DeleteLinesandLabelsLong [UI / Chart-Objekte]

```
void DeleteLinesandLabelsLong()
```

Ort im Code: gui_elemente.mqh Zeile 333 bis 351

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_Reg_DeleteOne.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int i = total - 1; i >= 0; i--)
- for(int p = 0; p < ArraySize(prefixes); p++)
- if(StringFind(name, prefixes[p]) == 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
334: {
335: // löscht alle Objekte, die zu LONG-Trade-Linien/Labels gehören (inkl. _1.._4)
336: string prefixes[] = {SL_Long, Entry_Long, LabelSLLong, LabelEntryLong};
338: for (int i = total - 1; i >= 0; i--)
341: for (int p = 0; p < ArraySize(prefixes); p++)
343: if (StringFind(name, prefixes[p]) == 0)
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_DeleteOne	trades_panel.mqh:775 DeleteLinesandLabelsLong();

UI_DeleteAllTradePosLinesByScan [UI / Chart-Objekte]

```
int UI_DeleteAllTradePosLinesByScan()
```

Ort im Code: gui_elemente.mqh Zeile 367 bis 400

Kommentar direkt darueber: Wichtig: löscht NICHT PR_HL / SL_HL (Basislinien) – nur die suf-Linien.
Rückgabe: Anzahl gelöschter Objekte.

Rueckgabe	Parameter (Typ name [default])

int	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_Reg_DeleteOne.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int i = total - 1; i >= 0; i--)
- if(name == "")
- if(match)
- if(UI_Reg_DeleteOne(name))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
368: {
380: for (int i = total - 1; i >= 0; i--)
383: if (name == "")
392: if (match)
394: if (UI_Reg_DeleteOne(name))
399: return deleted;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_DeleteOne	(keine Treffer im Projekt)

createButton [UI / Chart-Objekte]

```
bool createButton(string objName, string text, int xD, int yD, int xS, int yS, color clrTxt, color
```

Ort im Code: gui_elemente.mqh Zeile 405 bis 433

Kommentar direkt darueber: | Create Trading Button |

Rueckgabe	Parameter (Typ name [default])
bool	string objName string text int xD int yD int xS int yS color clrTxt color clrBG int fontsize = 8 color clrBorder = clrNONE string font = "Arial"

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben; Logging/Benachrichtigung.
Wichtige Aufrufe: UI_Reg_Add.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!ObjectCreate(0, objName, OBJ_BUTTON, 0, 0, TimeCurrent()))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
408: if (!ObjectCreate(0, objName, OBJ_BUTTON, 0, 0, TimeCurrent()))
410: Print(__FUNCTION__, ": Failed to create Btn: Error Code: ", GetLastError());
414: ObjectSetInteger(0, objName, OBJPROP_XDISTANCE, xD);
415: ObjectSetInteger(0, objName, OBJPROP_YDISTANCE, yD);
416: ObjectSetInteger(0, objName, OBJPROP_XSIZE, xS);
417: ObjectSetInteger(0, objName, OBJPROP_YSIZE, yS);
418: ObjectSetInteger(0, objName, OBJPROP_CORNER, CORNER_LEFT_UPPER);
419: ObjectSetString(0, objName, OBJPROP_TEXT, text);
420: ObjectSetInteger(0, objName, OBJPROP_FONTSIZE, InpFontSize);
421: ObjectSetString(0, objName, OBJPROP_FONT, InpFont);
422: ObjectSetInteger(0, objName, OBJPROP_COLOR, clrTxt);
423: ObjectSetInteger(0, objName, OBJPROP_BGCOLOR, clrBG);
424: ObjectSetInteger(0, objName, OBJPROP_BORDER_COLOR, clrBorder);
425: ObjectSetInteger(0, objName, OBJPROP_BACK, false);
426: ObjectSetInteger(0, objName, OBJPROP_STATE, false);
427: ObjectSetInteger(0, objName, OBJPROP_SELECTABLE, false);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_Add	Trade Assistent V2.008.mq5:1161 createButton(EntryButton, "", xd3, yd3, xs3, ys3, PriceButton_font_color, PriceButton_bgcolor, InpFontSize, clrNONE, InpFont); Trade Assistent V2.008.mq5:1199 createButton(SLButton, "", xd5, yd5, xs5, ys5, SLButton_font_color, SLButton_bgcolor, InpFontSize, clrNONE, InpFont);

Verwendete MQL-Konstanten (UI)

- Objekt-Typen: OBJ_BUTTON
- Objekt-Properties: OBJPROP_ANCHOR, OBJPROP_BACK, OBJPROP_BGCOLOR, OBJPROP_BORDER_COLOR, OBJPROP_COLOR, OBJPROP_CORNER, OBJPROP_FONT, OBJPROP_FONTSIZE, OBJPROP_SELECTABLE, OBJPROP_SELECTED, OBJPROP_STATE, OBJPROP_TEXT

createHL [UI / Chart-Objekte]

```
bool createHL(string objName, datetime time1, double price1, color clr)
```

Ort im Code: gui_elemente.mqh Zeile 438 bis 455

Kommentar direkt darueber: | Create Preislinien Trading Buttton |

Rueckgabe	Parameter (Typ name [default])
bool	string objName datetime time1 double price1 color clr

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben; Logging/Benachrichtigung. Wichtige Aufrufe: UI_Reg_Add.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!ObjectCreate(0, objName, OBJ_HLINE, 0, time1, price1))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
441: if (!ObjectCreate(0, objName, OBJ_HLINE, 0, time1, price1))
443: Print(__FUNCTION__, ": Failed to create HL: Error Code: ", GetLastError());
```

```
444: return (false);
447: ObjectSetInteger(0, objName, OBJPROP_TIME, time1);
448: ObjectSetDouble(0, objName, OBJPROP_PRICE, price1);
449: ObjectSetInteger(0, objName, OBJPROP_COLOR, clr);
450: ObjectSetInteger(0, objName, OBJPROP_BACK, false);
451: ObjectSetInteger(0, objName, OBJPROP_STYLE, STYLE_SOLID);
453: ChartRedraw(0);
454: return (true);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_Add	Trade Assistent V2.008.mq5:1171 createHL(SL_HL, dt_sl, price_sl, color_SLLine);

Verwendete MQL-Konstanten (UI)

- Objekt-Typen: OBJ_HLINE
- Objekt-Properties: OBJPROP_BACK, OBJPROP_COLOR, OBJPROP_PRICE, OBJPROP_STYLE, OBJPROP_TIME

SendButton [UI / Chart-Objekte]

```
void SendButton()
```

Ort im Code: gui_elemente.mqh Zeile 459 bis 517

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: UI_Reg_Add, del_SENDBTN_TRNB_POSNB.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!SendOnlyButton)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
464: ObjectCreate(0, SENDTRADEBTN, OBJ_BUTTON, 0, 0, 0);
466: ObjectSetInteger(0, SENDTRADEBTN, OBJPROP_XDISTANCE, xd3 - 100);
467: ObjectSetInteger(0, SENDTRADEBTN, OBJPROP_XSIZE, 100);
468: ObjectSetInteger(0, SENDTRADEBTN, OBJPROP_YDISTANCE, yd3);
469: ObjectSetInteger(0, SENDTRADEBTN, OBJPROP_YSIZE, 30);
470: ObjectSetInteger(0, SENDTRADEBTN, OBJPROP_CORNER, 0);
473: ObjectSetString(0, SENDTRADEBTN, OBJPROP_TEXT, "T & S"); // label
474: ObjectSetInteger(0, SENDTRADEBTN, OBJPROP_BGCOLOR, TSButton_bgcolor);
475: ObjectSetInteger(0, SENDTRADEBTN, OBJPROP_COLOR, TSButton_font_color);
479: ObjectSetString(0, SENDTRADEBTN, OBJPROP_TEXT, "Send only"); // label
480: ObjectSetInteger(0, SENDTRADEBTN, OBJPROP_BGCOLOR, SendOnlyButton_bgcolor);
481: ObjectSetInteger(0, SENDTRADEBTN, OBJPROP_COLOR, SendOnlyButton_font_color);
484: ObjectSetString(0, SENDTRADEBTN, OBJPROP_FONT, InpFont);
485: ObjectSetInteger(0, SENDTRADEBTN, OBJPROP_FONTSIZE, InpFontSize);
488: ObjectCreate(0, TRNB, OBJ_EDIT, 0, 0, 0);
490: ObjectSetInteger(0, TRNB, OBJPROP_XDISTANCE, xd3 - 100);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_Add, del_SENDBTN_TRNB_POSNB	Trade Assistent V2.008.mq5:1187 SendButton();

Verwendete MQL-Konstanten (UI)

- Objekt-Typen: OBJ_BUTTON, OBJ_EDIT
- Objekt-Properties: OBJPROP_ALIGN, OBJPROP_BGCOLOR, OBJPROP_COLOR, OBJPROP_CORNER, OBJPROP_FONT, OBJPROP_FONTSIZE, OBJPROP_READONLY, OBJPROP_TEXT, OBJPROP_XDISTANCE, OBJPROP_XSIZE, OBJPROP_YDISTANCE, OBJPROP_YSIZE

del_SENDBTN_TRNB_POSNB [UI / Chart-Objekte]

```
void del_SENDBTN_TRNB_POSNB()
```

Ort im Code: gui_elemente.mqh Zeile 518 bis 523

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_Reg_DeleteOne.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
519: {
520: UI_Reg_DeleteOne(SENDTRADEBTN);
521: UI_Reg_DeleteOne(TRNB);
522: UI_Reg_DeleteOne(POSNB);
523: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_DeleteOne	gui_elemente.mqh:462 del_SENDBTN_TRNB_POSNB();

SabioEdit [UI / Chart-Objekte]

```
void SabioEdit()
```

Ort im Code: gui_elemente.mqh Zeile 527 bis 565

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: UI_Reg_Add, del_sabiosl_sabioEntry.

2 2. Objektkoordinaten angeben: Interne Logik / Berechnung

3 3. Objektgröße setzen: Interne Logik / Berechnung

4 4. den Text setzen: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: Get_Price_s.

5 5. Schriftgröße setzen: Interne Logik / Berechnung

6 6. aktivieren (true) oder deaktivieren (false) den schreibgeschützten Modus: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: UI_Reg_Add.

7 7. Objektkoordinaten angeben: Interne Logik / Berechnung

8 8. Objektgröße setzen: Interne Logik / Berechnung

9 9. den Text setzen: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: Get_Price_s.

10 10. Schriftgröße setzen: Interne Logik / Berechnung

11 ... (weitere Abschnitte: 1)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
531: ObjectCreate(0, SabioSL, OBJ_EDIT, 0, 0, 0);
534: ObjectSetInteger(0, SabioSL, OBJPROP_XDISTANCE, xd5);
535: ObjectSetInteger(0, SabioSL, OBJPROP_YDISTANCE, yd5 + 30);
537: ObjectSetInteger(0, SabioSL, OBJPROP_XSIZE, 280);
538: ObjectSetInteger(0, SabioSL, OBJPROP_YSIZE, 30);
540: ObjectSetString(0, SabioSL, OBJPROP_TEXT, "SABIO SL: " + Get_Price_s(SL_HL));
542: ObjectSetInteger(0, SabioSL, OBJPROP_BGCOLOR, clrWhite);
543: ObjectSetInteger(0, SabioSL, OBJPROP_COLOR, clrBlack);
546: ObjectSetInteger(0, SabioSL, OBJPROP_READONLY, false);
549: ObjectCreate(0, SabioEntry, OBJ_EDIT, 0, 0, 0);
552: ObjectSetInteger(0, SabioEntry, OBJPROP_XDISTANCE, xd3);
553: ObjectSetInteger(0, SabioEntry, OBJPROP_YDISTANCE, yd3 + 30);
555: ObjectSetInteger(0, SabioEntry, OBJPROP_XSIZE, 280);
556: ObjectSetInteger(0, SabioEntry, OBJPROP_YSIZE, 30);
558: ObjectSetString(0, SabioEntry, OBJPROP_TEXT, "SABIO ENTRY: " + Get_Price_s(PR_HL));
560: ObjectSetInteger(0, SabioEntry, OBJPROP_BGCOLOR, clrWhite);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
Get_Price_s, UI_Reg_Add, del_sabiosl_sabioEntry	Trade Assistent V2.008.mq5:1181 SabioEdit();

Verwendete MQL-Konstanten (UI)

- Objekt-Typen: OBJ_EDIT
- Objekt-Properties: OBJPROP_BGCOLOR, OBJPROP_COLOR, OBJPROP_READONLY, OBJPROP_TEXT, OBJPROP_XDISTANCE, OBJPROP_XSIZE, OBJPROP_YDISTANCE, OBJPROP_YSIZE

del_sabiosl_sabioEntry [UI / Chart-Objekte]

```
void del_sabiosl_sabioEntry()
```

Ort im Code: gui_elemente.mqh Zeile 566 bis 570

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_Reg_DeleteOne.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
567: {
568: UI_Reg_DeleteOne(SabioEntry);
569: UI_Reg_DeleteOne(SabioSL);
570: }
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_DeleteOne	gui_elemente.mqh:529 del_sabiosl_sabioEntry();

UI_GetOverviewTopY [UI / Chart-Objekte]

```
int UI_GetOverviewTopY()
```

Ort im Code: gui_elemente.mqh Zeile 573 bis 609

Kommentar direkt darueber: ================== OVERVIEW PANEL (LONG | SHORT) ================== Bestimmt die Y-Position unterhalb der Cancel-Buttons.

Rueckgabe	Parameter (Typ name [default])
int	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung
2. 2. Cancel Buttons: Interne Logik / Berechnung
3. 3. ActiveTrade Labels: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int i = 0; i < ArraySize(btns); i++)
- if(ObjectFind(0, btns[i]) >= 0)
- if(y_max < 0)
- for(int i = 0; i < ArraySize(lbls); i++)
- if(ObjectFind(0, lbls[i]) >= 0)
- if(y_max < 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
579: string btns[] = {"ButtonCancelOrder", "ButtonCancelOrderSell"};
580: for (int i = 0; i < ArraySize(btns); i++)
582: if (ObjectFind(0, btns[i]) >= 0)
591: if (y_max < 0)
594: for (int i = 0; i < ArraySize(lbls); i++)
596: if (ObjectFind(0, lbls[i]) >= 0)
605: if (y_max < 0)
606: return fallback_y;
608: return y_max + 10; // Abstand nach unten
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	gui_elemente.mqh:619 int panel_y = UI_GetOverviewTopY(); gui_elemente.mqh:677 return UI_GetOverviewTopY() + 10;

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_YDISTANCE, OBJPROP_YSIZE

UI_PositionOverviewPanel [UI / Chart-Objekte]

```
void UI_PositionOverviewPanel()
```

Ort im Code: gui_elemente.mqh Zeile 612 bis 658

Kommentar direkt darueber: Positioniert BG + Labels konsistent unter den Cancel-Buttons.

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_GetOverviewTopY.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, TA_OVERVIEW_BG) < 0)
- if(ObjectFind(0, TA_OVERVIEW_TXT) >= 0)
- if(ObjectFind(0, TA_OVERVIEW_TXT_LONG) >= 0)
- if(ObjectFind(0, TA_OVERVIEW_TXT_SHORT) >= 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
614: if (ObjectFind(0, TA_OVERVIEW_BG) < 0)
615: return;
623: ObjectSetInteger(0, TA_OVERVIEW_BG, OBJPROP_CORNER, CORNER_LEFT_UPPER);
624: ObjectSetInteger(0, TA_OVERVIEW_BG, OBJPROP_XDISTANCE, panel_x);
625: ObjectSetInteger(0, TA_OVERVIEW_BG, OBJPROP_YDISTANCE, panel_y);
626: ObjectSetInteger(0, TA_OVERVIEW_BG, OBJPROP_XSIZE, panel_w);
627: ObjectSetInteger(0, TA_OVERVIEW_BG, OBJPROP_YSIZE, panel_h);
632: ObjectSetInteger(0, TA_OVERVIEW_TXT, OBJPROP_CORNER, CORNER_LEFT_UPPER);
633: ObjectSetInteger(0, TA_OVERVIEW_TXT, OBJPROP_XDISTANCE, panel_x + 8);
634: ObjectSetInteger(0, TA_OVERVIEW_TXT, OBJPROP_YDISTANCE, panel_y + 6);
647: ObjectSetInteger(0, TA_OVERVIEW_TXT_LONG, OBJPROP_CORNER, CORNER_LEFT_UPPER);
648: ObjectSetInteger(0, TA_OVERVIEW_TXT_LONG, OBJPROP_XDISTANCE, x_long);
649: ObjectSetInteger(0, TA_OVERVIEW_TXT_LONG, OBJPROP_YDISTANCE, y_text);
654: ObjectSetInteger(0, TA_OVERVIEW_TXT_SHORT, OBJPROP_CORNER, CORNER_LEFT_UPPER);
655: ObjectSetInteger(0, TA_OVERVIEW_TXT_SHORT, OBJPROP_XDISTANCE, x_short);
656: ObjectSetInteger(0, TA_OVERVIEW_TXT_SHORT, OBJPROP_YDISTANCE, y_text);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_GetOverviewTopY	(keine Treffer im Projekt)

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_CORNER, OBJPROP_XDISTANCE, OBJPROP_XSIZE, OBJPROP_YDISTANCE, OBJPROP_YSIZE

UI_TradeLists_TopY [UI / Chart-Objekte]

```
int UI_TradeLists_TopY()
```

Ort im Code: gui_elemente.mqh Zeile 667 bis 678

Rueckgabe	Parameter (Typ name [default])
int	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_GetOverviewTopY.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, TA_OVERVIEW_BG) >= 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
668: {
669: // unter das bestehende Overview-Panel (falls vorhanden)
670: if (ObjectFind(0, TA_OVERVIEW_BG) >= 0)
670: if (ObjectFind(0, TA_OVERVIEW_BG) >= 0)
674: return y + h + 10;
677: return UI_GetOverviewTopY() + 10;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_GetOverviewTopY	gui_elemente.mqh:686 int y = UI_TradeLists_TopY();

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_YDISTANCE, OBJPROP_YSIZE

UI_TradeLists_Height [UI / Chart-Objekte]

```
int UI_TradeLists_Height()
```

Ort im Code: gui_elemente.mqh Zeile 683 bis 694

Rueckgabe	Parameter (Typ name [default])
int	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_TradeLists_TopY.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(h < 160)
- if(h > 360)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
684: {
685: int ch = (int)ChartGetInteger(0, CHART_HEIGHT_IN_PIXELS, 0);
686: int y = UI_TradeLists_TopY();
689: if (h < 160)
691: if (h > 360)
693: return h;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_TradeLists_TopY	(keine Treffer im Projekt)

Verwendete MQL-Konstanten (UI)

- Chart-Properties: CHART_HEIGHT_IN_PIXELS

UI_TradeLists_Deinit [UI / Chart-Objekte]

```
void UI_TradeLists_Deinit(const int reason)
```

Ort im Code: gui_elemente.mqh Zeile 699 bis 705

Rueckgabe	Parameter (Typ name [default])
void	const int reason

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!g_TA_TradeListsCreated)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
700: {
701: if (!g_TA_TradeListsCreated)
701: if (!g_TA_TradeListsCreated)
702: return;
702: return;
704: g_TA_TradeListsCreated = false;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	(keine Treffer im Projekt)

UI_TradeLists_AutoRefresh [UI / Chart-Objekte]

```
void UI_TradeLists_AutoRefresh()
```

Ort im Code: gui_elemente.mqh Zeile 708 bis 716

Kommentar direkt darueber: optional: in OnTick aufrufen (throttled)

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(now_ms - last_ms < 1500)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
709: {
710: static uint last_ms = 0;
711: uint now_ms = GetTickCount();
712: if (now_ms - last_ms < 1500)
712: if (now_ms - last_ms < 1500)
713: return;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	(keine Treffer im Projekt)

trades_panel.mqh

Linkes Trades-Panel (Long/Short Bereiche, Row-Buttons, Rebuild/Show/Hide)

Funktionen in dieser Datei (23): TP_DeleteByPrefix, TP_CreateRectBG, Del_TP_CreateRectBG, TP_CreateLabel, TP_CreateButton, UI_TradesPanel_Create, showCancel_long, showActive_long, showCancel_short, showActive_short, UI_TradesPanel_Destroy, UI_DeleteObjectIfExists, UI_DeleteLineAndTag, UI_DeleteTradePosLines, UI_TradesPanel_RebuildRows, UI_TradesPanel_RebuildRows, SortRowsByTradePos, UI_TradesPanel_OnChartEvent, UI_ParseTradePosFromButtonName, UI_IsPriceVisible, UI_ChartMidPrice, UI_DrawPriceOrMid, UI_DeleteTradeLinesByTradeNo

TP_DeleteByPrefix [Trades-Panel (UI links)]

```
void TP_DeleteByPrefix(const string prefix)
```

Ort im Code: trades_panel.mqh Zeile 46 bis 55

Kommentar direkt darueber: ================== Helpers ==================

Rueckgabe	Parameter (Typ name [default])
void	const string prefix

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_Reg_DeleteOne.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int i = total - 1; i >= 0; --i)
- if(StringFind(n, prefix, 0) == 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
47: {
48: int total = ObjectsTotal(0, -1, -1);
49: for(int i = total - 1; i >= 0; --i)
49: for(int i = total - 1; i >= 0; --i)
50: {
52: if(StringFind(n, prefix, 0) == 0)
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_DeleteOne	trades_panel.mqh:287 TP_DeleteByPrefix(TP_ROW_LONG_TR_PREFIX); trades_panel.mqh:288 TP_DeleteByPrefix(TP_ROW_SHORT_TR_PREFIX); trades_panel.mqh:290 TP_DeleteByPrefix(TP_ROW_LONG_TR_Cancel_PREFIX); trades_panel.mqh:291 TP_DeleteByPrefix(TP_ROW_SHORT_TR_Cancel_PREFIX); trades_panel.mqh:293 TP_DeleteByPrefix(TP_ROW_LONG_PREFIX); trades_panel.mqh:294 TP_DeleteByPrefix(TP_ROW_SHORT_PREFIX); trades_panel.mqh:296 TP_DeleteByPrefix(TP_ROW_LONG_Cancel_PREFIX); trades_panel.mqh:297 TP_DeleteByPrefix(TP_ROW_SHORT_Cancel_PREFIX); ... (12 weitere)

TP_CreateRectBG [Trades-Panel (UI links)]

```
bool TP_CreateRectBG(const string name, const int x, const int y, const int w, const int h)
```

Ort im Code: trades_panel.mqh Zeile 60 bis 82

Rueckgabe	Parameter (Typ name [default])
bool	const string name const int x const int y const int w const int h

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: UI_Reg_Add.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, name) < 0)
- if(!ObjectCreate(0, name, OBJ_RECTANGLE_LABEL, 0, 0, 0))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
62: if(ObjectFind(0, name) < 0)
63: if(!ObjectCreate(0, name, OBJ_RECTANGLE_LABEL, 0, 0, 0))
64: return false;
67: ObjectSetInteger(0, name, OBJPROP_CORNER, CORNER_LEFT_UPPER);
68: ObjectSetInteger(0, name, OBJPROP_XDISTANCE, x);
69: ObjectSetInteger(0, name, OBJPROP_YDISTANCE, y);
70: ObjectSetInteger(0, name, OBJPROP_XSIZE, w);
71: ObjectSetInteger(0, name, OBJPROP_YSIZE, h);
73: ObjectSetInteger(0, name, OBJPROP_COLOR, clrDimGray);
75: ObjectSetInteger(0, name, OBJPROP_SELECTABLE, false);
76: ObjectSetInteger(0, name, OBJPROP_BGCOLOR, clrBlack); // NICHT clrNONE
77: ObjectSetInteger(0, name, OBJPROP_BACK, false);       // Vordergrund
78: ObjectSetInteger(0, name, OBJPROP_ZORDER, 10000);     // ganz nach vorne
81: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

UI_Reg_Add	trades_panel.mqh:160 if(!TP_CreateRectBG(TP_BG, TP_X, TP_Y, TP_W, TP_H))
	trades_panel.mqh:367 TP_CreateRectBG(TP_BG, TP_X, TP_Y, TP_W, TP_H);
	trades_panel.mqh:494 TP_CreateRectBG(TP_BG, TP_X, TP_Y, TP_W, TP_H);

Verwendete MQL-Konstanten (UI)

- Objekt-Typen: OBJ_RECTANGLE_LABEL
- Objekt-Properties: OBJPROP_BACK, OBJPROP_BGCOLOR, OBJPROP_COLOR, OBJPROP_CORNER, OBJPROP_SELECTABLE, OBJPROP_XDISTANCE, OBJPROP_XSIZE, OBJPROP_YDISTANCE, OBJPROP_YSIZE, OBJPROP_ZORDER

Del_TP_CreateRectBG [Trades-Panel (UI links)]

```
bool Del_TP_CreateRectBG(const string name)
```

Ort im Code: trades_panel.mqh Zeile 86 bis 92

Rueckgabe	Parameter (Typ name [default])
bool	const string name

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_Reg_DeleteOne.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!UI_Reg_DeleteOne(name))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
87: {
88: if(!UI_Reg_DeleteOne(name))
88: if(!UI_Reg_DeleteOne(name))
89: return false;
89: return false;
91: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_DeleteOne	trades_panel.mqh:276 Del_TP_CreateRectBG(TP_BG);

TP_CreateLabel [Trades-Panel (UI links)]

```
bool TP_CreateLabel(const string name, const int x, const int y, const int w, const int h, const s
```

Ort im Code: trades_panel.mqh Zeile 96 bis 114

Rueckgabe	Parameter (Typ name [default])
bool	const string name const int x const int y const int w const int h const string txt

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: UI_Reg_Add.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, name) < 0)
- if(!ObjectCreate(0, name, OBJ_LABEL, 0, 0, 0))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
 98: if(ObjectFind(0, name) < 0)
 99: if(!ObjectCreate(0, name, OBJ_LABEL, 0, 0, 0))
100: return false;
102: ObjectSetInteger(0, name, OBJPROP_CORNER, CORNER_LEFT_UPPER);
103: ObjectSetInteger(0, name, OBJPROP_XDISTANCE, x);
104: ObjectSetInteger(0, name, OBJPROP_YDISTANCE, y);
105: ObjectSetInteger(0, name, OBJPROP_XSIZE, w);
106: ObjectSetInteger(0, name, OBJPROP_YSIZE, h);
108: ObjectSetString(0, name, OBJPROP_TEXT, txt);
109: ObjectSetInteger(0, name, OBJPROP_COLOR, clrWhite);
```

```
110: ObjectSetInteger(0, name, OBJPROP_SELECTABLE, false);
111: ObjectSetInteger(0, name, OBJPROP_ZORDER, 10010);
113: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

UI_Reg_Add	trades_panel.mqh:165 TP_CreateLabel(TP_LBL_LONG, xL, y1, col_w, TP_HDR_H, "LONG:");
	trades_panel.mqh:166 TP_CreateLabel(TP_LBL_SHORT, xR, y1, col_w, TP_HDR_H, "SHORT:");
	trades_panel.mqh:370 TP_CreateLabel(TP_LBL_LONG, xL, y1, col_w, TP_HDR_H, "LONG");
	trades_panel.mqh:371 TP_CreateLabel(TP_LBL_SHORT, xR, y1, col_w, TP_HDR_H, "SHORT");
	trades_panel.mqh:497 TP_CreateLabel(TP_LBL_LONG, xL, y1, col_w, TP_HDR_H, "LONG");
	trades_panel.mqh:498 TP_CreateLabel(TP_LBL_SHORT, xR, y1, col_w, TP_HDR_H, "SHORT");

Verwendete MQL-Konstanten (UI)

- Objekt-Typen: OBJ_LABEL
- Objekt-Properties: OBJPROP_COLOR, OBJPROP_CORNER, OBJPROP_SELECTABLE, OBJPROP_TEXT, OBJPROP_XDISTANCE, OBJPROP_XSIZE, OBJPROP_YDISTANCE, OBJPROP_YSIZE, OBJPROP_ZORDER

TP_CreateButton [Trades-Panel (UI links)]

```
bool TP_CreateButton(const string name, const int x, const int y, const int w, const int h, const
                     const int fontsize = 8)
```

Ort im Code: trades_panel.mqh Zeile 119 bis 142

Rueckgabe	Parameter (Typ name [default])
bool	const string name const int x const int y const int w const int h const string txt const int fontsize = 8

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: UI_Reg_Add.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, name) < 0)
- if(!ObjectCreate(0, name, OBJ_BUTTON, 0, 0, 0))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
122: if(ObjectFind(0, name) < 0)
123: if(!ObjectCreate(0, name, OBJ_BUTTON, 0, 0, 0))
124: return false;
127: ObjectSetInteger(0, name, OBJPROP_CORNER, CORNER_LEFT_UPPER);
128: ObjectSetInteger(0, name, OBJPROP_XDISTANCE, x);
129: ObjectSetInteger(0, name, OBJPROP_YDISTANCE, y);
130: ObjectSetInteger(0, name, OBJPROP_XSIZE, w);
131: ObjectSetInteger(0, name, OBJPROP_YSIZE, h);
133: ObjectSetString(0, name, OBJPROP_TEXT, txt);
```

```
134: ObjectSetInteger(0, name, OBJPROP_FONTSIZE, InpFontSize);
135: ObjectSetString(0, name, OBJPROP_FONT, InpFont);
136: ObjectSetInteger(0, name, OBJPROP_SELECTABLE, false);
137: ObjectSetInteger(0, name, OBJPROP_STATE, 0);
138: ObjectSetInteger(0, name, OBJPROP_ZORDER, 10010);
141: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_Add	trades_panel.mqh:170 TP_CreateButton(TP_BTN_ACTIVE_LONG, xL, y2, col_w, TP_BTN_H, "Active Trade", InpFontSize); trades_panel.mqh:171 TP_CreateButton(TP_BTN_ACTIVE_SHORT, xR, y2, col_w, TP_BTN_H, "Active Trade", InpFontSize); trades_panel.mqh:178 TP_CreateButton(TP_BTN_CANCEL_LONG, xL, y3, col_w, TP_BTN_H, "Cancel Trade", InpFontSize); trades_panel.mqh:179 TP_CreateButton(TP_BTN_CANCEL_SHORT, xR, y3, col_w, TP_BTN_H, "Cancel Trade", InpFontSize); trades_panel.mqh:374 TP_CreateButton(TP_BTN_ACTIVE_LONG, xL, y2, col_w, TP_BTN_H, "Active Trade", 9); trades_panel.mqh:375 TP_CreateButton(TP_BTN_ACTIVE_SHORT, xR, y2, col_w, TP_BTN_H, "Active Trade", 9); trades_panel.mqh:378 TP_CreateButton(TP_BTN_CANCEL_LONG, xL, y3, col_w, TP_BTN_H, "Cancel Trade", 9); trades_panel.mqh:379 TP_CreateButton(TP_BTN_CANCEL_SHORT, xR, y3, col_w, TP_BTN_H, "Cancel Trade", 9); ... (20 weitere)

Verwendete MQL-Konstanten (UI)

- Objekt-Typen: OBJ_BUTTON
- Objekt-Properties: OBJPROP_CORNER, OBJPROP_FONT, OBJPROP_FONTSIZE, OBJPROP_SELECTABLE, OBJPROP_STATE, OBJPROP_TEXT, OBJPROP_XDISTANCE, OBJPROP_XSIZE, OBJPROP_YDISTANCE, OBJPROP_YSIZE, OBJPROP_ZORDER

UI_TradesPanel_Create [Trades-Panel (UI links)]

```
bool UI_TradesPanel_Create(const int x, const int y, const int w, const int h)
```

Ort im Code: trades_panel.mqh Zeile 145 bis 186

Kommentar direkt darueber: ================== API: Create / Destroy / Rebuild ==================

Rueckgabe	Parameter (Typ name [default])
bool	const int x const int y const int w const int h

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: TP_CreateRectBG.

2 2. Labels LONG / SHORT: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: TP_CreateLabel.

3 3. ActiveTrade Buttons: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: TP_CreateButton, showActive_long, showActive_short.

4 4. CancelTrade Buttons: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: TP_CreateButton, showCancel_long, showCancel_short.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(!TP_CreateRectBG(TP_BG, TP_X, TP_Y, TP_W, TP_H))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
146: {
147: TP_X = x;
160: if(!TP_CreateRectBG(TP_BG, TP_X, TP_Y, TP_W, TP_H))
161: return false;
184: ChartRedraw(0);
185: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
TP_CreateButton, TP_CreateLabel, TP_CreateRectBG, showActive_long, showActive_short, showCancel_long, showCancel_short	Trade Assistent V2.008.mq5:223 UI_TradesPanel_Create(10, 40, 530, 400);

showCancel_long [Trades-Panel (UI links)]

```
void showCancel_long(bool show)
```

Ort im Code: trades_panel.mqh Zeile 191 bis 207

Rueckgabe	Parameter (Typ name [default])
void	bool show

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(show)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
194: if(show)
196: ObjectSetInteger(0, TP_BTN_CANCEL_LONG, OBJPROP_COLOR, clrBlack);
197: ObjectSetInteger(0, TP_BTN_CANCEL_LONG, OBJPROP_BGCOLOR, clrWhite);
198: ObjectSetInteger(0, TP_BTN_CANCEL_LONG, OBJPROP_BORDER_COLOR, clrBlack);
203: ObjectSetInteger(0, TP_BTN_CANCEL_LONG, OBJPROP_COLOR, clrNONE);
204: ObjectSetInteger(0, TP_BTN_CANCEL_LONG, OBJPROP_BGCOLOR, clrNONE);
205: ObjectSetInteger(0, TP_BTN_CANCEL_LONG, OBJPROP_BORDER_COLOR, clrNONE);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	discord_send.mqh:133 showCancel_long(true);
	trades_panel.mqh:181 showCancel_long(false);
	trades_panel.mqh:454 showCancel_long(true);
	trades_panel.mqh:682 showCancel_long(true);

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_BGCOLOR, OBJPROP_BORDER_COLOR, OBJPROP_COLOR

showActive_long [Trades-Panel (UI links)]

```
void showActive_long(bool show)
```

Ort im Code: trades_panel.mqh Zeile 211 bis 227

Rueckgabe	Parameter (Typ name [default])
void	bool show

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(show)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
214: if(show)
216: ObjectSetInteger(0, TP_BTN_ACTIVE_LONG, OBJPROP_COLOR, clrWhite);
217: ObjectSetInteger(0, TP_BTN_ACTIVE_LONG, OBJPROP_BGCOLOR, clrRed);
218: ObjectSetInteger(0, TP_BTN_ACTIVE_LONG, OBJPROP_BORDER_COLOR, clrWhite);
223: ObjectSetInteger(0, TP_BTN_ACTIVE_LONG, OBJPROP_COLOR, clrNONE);
224: ObjectSetInteger(0, TP_BTN_ACTIVE_LONG, OBJPROP_BGCOLOR, clrNONE);
225: ObjectSetInteger(0, TP_BTN_ACTIVE_LONG, OBJPROP_BORDER_COLOR, clrNONE);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	discord_send.mqh:132 showActive_long(true);
	trades_panel.mqh:174 showActive_long(false);
	trades_panel.mqh:453 showActive_long(true);
	trades_panel.mqh:681 showActive_long(true);

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_BGCOLOR, OBJPROP_BORDER_COLOR, OBJPROP_COLOR

showCancel_short [Trades-Panel (UI links)]

```
void showCancel_short(bool show)
```

Ort im Code: trades_panel.mqh Zeile 232 bis 248

Rueckgabe	Parameter (Typ name [default])
void	bool show

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(show)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
235: if(show)
237: ObjectSetInteger(0, TP_BTN_CANCEL_SHORT, OBJPROP_COLOR, clrBlack);
238: ObjectSetInteger(0, TP_BTN_CANCEL_SHORT, OBJPROP_BGCOLOR, clrWhite);
239: ObjectSetInteger(0, TP_BTN_CANCEL_SHORT, OBJPROP_BORDER_COLOR, clrBlack);
244: ObjectSetInteger(0, TP_BTN_CANCEL_SHORT, OBJPROP_COLOR, clrNONE);
245: ObjectSetInteger(0, TP_BTN_CANCEL_SHORT, OBJPROP_BGCOLOR, clrNONE);
246: ObjectSetInteger(0, TP_BTN_CANCEL_SHORT, OBJPROP_BORDER_COLOR, clrNONE);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	discord_send.mqh:142 showCancel_short(true);
	trades_panel.mqh:182 showCancel_short(false);
	trades_panel.mqh:473 showCancel_short(true);
	trades_panel.mqh:687 showCancel_short(true);

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_BGCOLOR, OBJPROP_BORDER_COLOR, OBJPROP_COLOR

showActive_short [Trades-Panel (UI links)]

```
void showActive_short(bool show)
```

Ort im Code: trades_panel.mqh Zeile 252 bis 268

Rueckgabe	Parameter (Typ name [default])
void	bool show

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(show)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
255: if(show)
257: ObjectSetInteger(0, TP_BTN_ACTIVE_SHORT, OBJPROP_COLOR, clrWhite);
258: ObjectSetInteger(0, TP_BTN_ACTIVE_SHORT, OBJPROP_BGCOLOR, clrRed);
259: ObjectSetInteger(0, TP_BTN_ACTIVE_SHORT, OBJPROP_BORDER_COLOR, clrWhite);
264: ObjectSetInteger(0, TP_BTN_ACTIVE_SHORT, OBJPROP_COLOR, clrNONE);
265: ObjectSetInteger(0, TP_BTN_ACTIVE_SHORT, OBJPROP_BGCOLOR, clrNONE);
266: ObjectSetInteger(0, TP_BTN_ACTIVE_SHORT, OBJPROP_BORDER_COLOR, clrNONE);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	discord_send.mqh:141 showActive_short(true);
	trades_panel.mqh:173 showActive_short(false);
	trades_panel.mqh:472 showActive_short(true);
	trades_panel.mqh:686 showActive_short(true);

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_BGCOLOR, OBJPROP_BORDER_COLOR, OBJPROP_COLOR

UI_TradesPanel_Destroy [Trades-Panel (UI links)]

```
void UI_TradesPanel_Destroy()
```

Ort im Code: trades_panel.mqh Zeile 273 bis 303

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: UI_Reg_DeleteOne, Del_TP_CreateRectBG, TP_DeleteByPrefix.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
274: {
276: Del_TP_CreateRectBG(TP_BG);
278: UI_Reg_DeleteOne(TP_BG);
280: UI_Reg_DeleteOne(TP_LBL_LONG);
281: UI_Reg_DeleteOne(TP_LBL_SHORT);
302: ChartRedraw(0);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
Del_TP_CreateRectBG, TP_DeleteByPrefix, UI_Reg_DeleteOne	(keine Treffer im Projekt)

UI_DeleteObjectIfExists [Trades-Panel (UI links)]

```
bool UI_DeleteObjectIfExists(const string name)
```

Ort im Code: trades_panel.mqh Zeile 314 bis 319

Kommentar direkt darueber: Löscht Entry/SL-Linien (und optional deren _TAG) für eine konkrete Trade/Pos. Passt exakt zu deinem Suffix-Schema: "_" + trade_no + "_" + pos_no Voraussetzung: Entry_Long, SL_Long, Entry_Short, SL_Short sind string-Constants.

Rueckgabe	Parameter (Typ name [default])
bool	const string name

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, name) >= 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
315: {
316: if(ObjectFind(0, name) >= 0)
316: if(ObjectFind(0, name) >= 0)
317: return ObjectDelete(0, name);
317: return ObjectDelete(0, name);
318: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	trades_panel.mqh:328 ok = UI_DeleteObjectIfExists(line_name) && ok; trades_panel.mqh:332 ok = UI_DeleteObjectIfExists(tag_name) && ok;

UI_DeleteLineAndTag [Trades-Panel (UI links)]

```
bool UI_DeleteLineAndTag(const string line_name)
```

Ort im Code: trades_panel.mqh Zeile 324 bis 335

Rueckgabe	Parameter (Typ name [default])
bool	const string line_name

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_DeleteObjectIfExists.

Wichtige Codezeilen (zum Mitlesen im Editor)

```
325: {
326: bool ok = true;
328: ok = UI_DeleteObjectIfExists(line_name) && ok;
330: // Tag-Objekt mit gleichem Namen + "_TAG" (falls du es nutzt)
331: string tag_name = line_name + "_TAG";
334: return ok;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

UI_DeleteObjectIfExists	trades_panel.mqh:348 ok = UI_DeleteLineAndTag(Entry_Long + suf) && ok;
	trades_panel.mqh:349 ok = UI_DeleteLineAndTag(SL_Long + suf) && ok;
	trades_panel.mqh:350 ok = UI_DeleteLineAndTag(Entry_Short + suf) && ok;
	trades_panel.mqh:351 ok = UI_DeleteLineAndTag(SL_Short + suf) && ok;

UI_DeleteTradePosLines [Trades-Panel (UI links)]

```
bool UI_DeleteTradePosLines(const int trade_no, const int pos_no)
```

Ort im Code: trades_panel.mqh Zeile 340 bis 354

Kommentar direkt darueber: Löscht die 4 möglichen Linien-Namen für genau diese Trade/Pos (LONG: Entry_Long_#_# und SL_Long_#_#) + (SHORT: Entry_Short_#_# und SL_Short_#_#) -> sicher, auch wenn nur LONG oder nur SHORT existiert.

Rueckgabe	Parameter (Typ name [default])
bool	const int trade_no const int pos_no

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_DeleteLineAndTag.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(trade_no <= 0 || pos_no <= 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
341: {
342: if(trade_no <= 0 || pos_no <= 0)
342: if(trade_no <= 0 || pos_no <= 0)
343: return false;
343: return false;
353: return ok;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_DeleteLineAndTag	(keine Treffer im Projekt)

UI_TradesPanel_RebuildRows [Trades-Panel (UI links)]

```
/*void UI_TradesPanel_RebuildRows()
```

Ort im Code: trades_panel.mqh Zeile 358 bis 478

Kommentar direkt darueber: 4) je ein Button pro Trade/Pos pro Seite (LONG links / SHORT rechts) + Header-Buttons (Active/Cancel) werden hier ebenfalls neu gesetzt/positioniert

Rueckgabe	Parameter (Typ name [default])
/*void	(keine)

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).
- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung
2. 2. Safety: Header-Controls sicherstellen (falls aus irgendeinem Grund weg): Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: TP_CreateButton, TP_CreateRectBG, TP_CreateLabel.
3. 3. alte Rows löschen: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: TP_DeleteByPrefix.
4. 4. Listbereich berechnen: Interne Logik / Berechnung
5. 5. DB laden & Rows bauen: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LoadPositions.
6. 6. Restore Entry/SL Lines für diese Position: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: TP_CreateButton, UI_DrawPriceOrMid, showActive_long, showCancel_long, showCancel_short, showActive_short, CreateEntryAndSLLines.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(maxRows < 1)
- for(int i=0; i<n; i++)
- if(StringFind(rows[i].status,"CLOSED",0)==0)
- if(rows[i].was_sent == 1 && rows[i].pos_no >= 1 && rows[i].pos_no <= 4)
- if(rows[i].direction == "LONG")
- if(rows[i].direction == "LONG")
- if(idxL >= maxRows)
- if(idxR >= maxRows)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
390: if(maxRows < 1) maxRows = 1;
393: DB_PositionRow rows[];
394: int n = DB_LoadPositions(_Symbol, (ENUM_TIMEFRAMES)_Period, rows);
399: for(int i=0; i<n; i++)
401: if(StringFind(rows[i].status,"CLOSED",0)==0)
404: // ---- Restore Entry/SL Lines für diese Position
405: if(rows[i].was_sent == 1 && rows[i].pos_no >= 1 && rows[i].pos_no <= 4)
415: if(rows[i].direction == "LONG")
438: if(rows[i].direction == "LONG")
440: if(idxL >= maxRows) continue;
459: if(idxR >= maxRows) continue;
477: ChartRedraw(0);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
CreateEntryAndSLLines, DB_LoadPositions, TP_CreateButton, TP_CreateLabel, TP_CreateRectBG, TP_DeleteByPrefix, UI_DrawPriceOrMid, showActive_long, showActive_short, showCancel_long, showCancel_short	Trade Assistent V2.008.mq5:238 UI_TradesPanel_RebuildRows(); Trade Assistent V2.008.mq5:330 UI_TradesPanel_RebuildRows(); discord_send.mqh:137 UI_TradesPanel_RebuildRows(); discord_send.mqh:146 UI_TradesPanel_RebuildRows(); discord_send.mqh:165 UI_TradesPanel_RebuildRows(); discord_send.mqh:172 UI_TradesPanel_RebuildRows(); trades_panel.mqh:778 UI_TradesPanel_RebuildRows(); trades_panel.mqh:783 UI_TradesPanel_RebuildRows(); ... (6 weitere)

UI_TradesPanel_RebuildRows [Trades-Panel (UI links)]

```
void UI_TradesPanel_RebuildRows()
```

Ort im Code: trades_panel.mqh Zeile 485 bis 691

Rueckgabe	Parameter (Typ name [default])
void	(keine)

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).
- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung
2. 2. Header / Panel: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: TP_CreateButton, TP_CreateRectBG, TP_CreateLabel.
3. 3. Rows löschen: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: TP_DeleteByPrefix.
4. 4. Listbereich: Interne Logik / Berechnung
5. 5. DB laden: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_LoadPositions.
6. 6. 1) In LONG/SHORT Arrays filtern (und Lines restoren): Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: UI_DrawPriceOrMid, UI_CreateOrUpdateLineTag, DB_SaveTradeLines, CreateEntryAndSLLines.
7. 7. Sortieren (Trade zuerst, dann Position): Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: SortRowsByTradePos.
8. 8. LONG Seite zeichnen: Trade-Header immer über den Positionen: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: TP_CreateButton.

9 9. SHORT Seite zeichnen: Trade-Header immer über den Positionen: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: TP_CreateButton.

10 10. Active/Cancel visibility nur einmal setzen: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: UI_UpdateAllLineTags, showActive_long, showCancel_long, showCancel_short, showActive_short.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(maxRows < 1)
- for(int i = 0; i < n; i++)
- if(StringFind(rows[i].status, "CLOSED", 0) == 0)
- if(rows[i].was_sent == 1 && rows[i].pos_no >= 1 && rows[i].pos_no <= 4)
- if(rows[i].direction == "LONG")
- if(rows[i].direction == "SHORT")
- if(rows[i].direction == "LONG")
- if(rows[i].direction == "SHORT")
- if(nL > 1)
- if(nS > 1)
- for(int i = 0; i < nL; i++)
- if(idxL >= maxRows)
- if(trade_no != lastTradeL)
- if(idxL >= maxRows)
- for(int i = 0; i < nS; i++)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
523: if(maxRows < 1)
527: DB_PositionRow rows[];
528: int n = DB_LoadPositions(_Symbol, (ENUM_TIMEFRAMES)_Period, rows);
531: DB_PositionRow longRows[];
532: DB_PositionRow shortRows[];
537: for(int i = 0; i < n; i++)
539: if(StringFind(rows[i].status, "CLOSED", 0) == 0)
542: // Restore Entry/SL Lines (pro Position)
543: if(rows[i].was_sent == 1 && rows[i].pos_no >= 1 && rows[i].pos_no <= 4)
552: if(rows[i].direction == "LONG")
558: DB_SaveTradeLines(suf);
567: DB_SaveTradeLines(suf);
586: // 4) Sortieren (Trade zuerst, dann Position)
592: // 5) LONG Seite zeichnen: Trade-Header immer über den Positionen
635: // 6) SHORT Seite zeichnen: Trade-Header immer über den Positionen
690: ChartRedraw(0);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
CreateEntryAndSLLines, DB_LoadPositions, DB_SaveTradeLines, SortRowsByTradePos, TP_CreateButton, TP_CreateLabel, TP_CreateRectBG, TP_DeleteByPrefix, UI_CreateOrUpdateLineTag, UI_DrawPriceOrMid, UI_UpdateAllLineTags, showActive_long, showActive_short, showCancel_long, showCancel_short	Trade Assistent V2.008.mq5:238 UI_TradesPanel_RebuildRows(); Trade Assistent V2.008.mq5:330 UI_TradesPanel_RebuildRows(); discord_send.mqh:137 UI_TradesPanel_RebuildRows(); discord_send.mqh:146 UI_TradesPanel_RebuildRows(); discord_send.mqh:165 UI_TradesPanel_RebuildRows(); discord_send.mqh:172 UI_TradesPanel_RebuildRows(); trades_panel.mqh:778 UI_TradesPanel_RebuildRows(); trades_panel.mqh:783 UI_TradesPanel_RebuildRows(); ... (6 weitere)

SortRowsByTradePos [Trades-Panel (UI links)]

```
void SortRowsByTradePos(DB_PositionRow &arr[], const int cnt)
```

Ort im Code: trades_panel.mqh Zeile 696 bis 718

Rueckgabe	Parameter (Typ name [default])
void	DB_PositionRow &arr[] const int cnt

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).

Ablauf (Schritt fuer Schritt)

1 1. Start: Persistenz (DB/Datei) lesen/schreiben

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int i = 0; i < cnt - 1; i++)
- for(int j = i + 1; j < cnt; j++)
- if(arr[j].trade_no < arr[i].trade_no)
- if(arr[j].trade_no == arr[i].trade_no && arr[j].pos_no < arr[i].pos_no)
- if(swap)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
698: for(int i = 0; i < cnt - 1; i++)
700: for(int j = i + 1; j < cnt; j++)
704: if(arr[j].trade_no < arr[i].trade_no)
707: if(arr[j].trade_no == arr[i].trade_no && arr[j].pos_no < arr[i].pos_no)
710: if(swap)
712: DB_PositionRow tmp = arr[i];
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	trades_panel.mqh:588 SortRowsByTradePos(longRows, nL); trades_panel.mqh:590 SortRowsByTradePos(shortRows, nS);

UI_TradesPanel_OnChartEvent [Trades-Panel (UI links)]

```
bool UI_TradesPanel_OnChartEvent(const int id, const long &lparam, const double &dparam, const str
```

Ort im Code: trades_panel.mqh Zeile 721 bis 908

Kommentar direkt darueber: ================== Optional: Click handling (Rows + Header Buttons) ==================

Rueckgabe	Parameter (Typ name [default])
bool	const int id const long &lparam const double &dparam const string &sparam

Seiteneffekte

- Liest/schreibt Persistenz (DB/Dateien).
- Kann Discord-Nachrichten versenden oder formatieren.

Ablauf (Schritt fuer Schritt)

1. 1. Start: Logging/Benachrichtigung
2. 2. Discord nur EINMAL senden: Persistenz (DB/Datei) lesen/schreiben; Discord-Webhook Nachricht senden/formatieren. Wichtige Aufrufe: SendDiscordMessage, FormatCancelTradeMessage, TF_ToString.
3. 3. DB: Trade sauber "geschlossen" markieren, damit OnInit ihn NICHT wieder aktiviert: Persistenz (DB/Datei) lesen/schreiben. Wichtige Aufrufe: DB_UpdatePositionStatus.
4. 4. Runtime-State komplett zurücksetzen: Interne Logik / Berechnung
5. 5. aktive Tradenummer löschen, sonst "reanimiert" OnInit das wieder: Persistenz (DB/Datei) lesen/schreiben; Discord-Webhook Nachricht senden/formatieren; Logging/Benachrichtigung. Wichtige Aufrufe: DB_Key, UI_DeleteTradeLinesByTradeNo, DeleteLinesandLabelsShort, SendDiscordMessage, UI_ParseTradePosFromButtonName, FormatCancelTradeMessage, DB_UpdatePositionStatus, DB_SetMetaInt.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(id != CHARTEVENT_OBJECT_CLICK)
- if(sparam == TP_BTN_ACTIVE_LONG)
- if(sparam == TP_BTN_ACTIVE_SHORT)
- if(sparam == TP_BTN_CANCEL_LONG)
- if(is_long_trade)
- if(sparam == TP_BTN_CANCEL_SHORT)
- if(is_sell_trade)
- if(StringFind(sparam, TP_ROW_LONG_Cancel_PREFIX, 0) == 0)
- if(UI_ParseTradePosFromButtonName(sparam, TP_ROW_LONG_Cancel_PREFIX, trade_no, pos_no))
- if(ObjectFind(0, sparam) >= 0)
- if(StringFind(sparam, TP_ROW_LONG_hitSL_PREFIX, 0) == 0)
- if(UI_ParseTradePosFromButtonName(sparam, TP_ROW_LONG_hitSL_PREFIX, trade_no, pos_no))
- if(ObjectFind(0, sparam) >= 0)
- if(StringFind(sparam, TP_ROW_SHORT_Cancel_PREFIX, 0) == 0)
- if(UI_ParseTradePosFromButtonName(sparam, TP_ROW_SHORT_Cancel_PREFIX, trade_no, pos_no))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
729: Print("Active LONG clicked");
734: Print("Active SHORT clicked");
739: Print("Cancel LONG clicked");
743: // 1) Discord nur EINMAL senden
744: DB_PositionRow r;
752: bool ret = SendDiscordMessage(message);
755: DB_UpdatePositionStatus(_Symbol, (ENUM_TIMEFRAMES)_Period, "LONG", active_long_trade_no, 0,
766: DB_SetMetaInt(DB_Key("active_long_trade_no"), active_long_trade_no);
769: ObjectSetInteger(0, TP_BTN_ACTIVE_LONG, OBJPROP_COLOR, clrNONE);
770: ObjectSetInteger(0, TP_BTN_ACTIVE_LONG, OBJPROP_BGCOLOR, clrNONE);
771: ObjectSetInteger(0, TP_BTN_ACTIVE_LONG, OBJPROP_BORDER_COLOR, clrNONE);
772: ObjectSetInteger(0, TP_BTN_CANCEL_LONG, OBJPROP_COLOR, clrNONE);
773: ObjectSetInteger(0, TP_BTN_CANCEL_LONG, OBJPROP_BGCOLOR, clrNONE);
774: ObjectSetInteger(0, TP_BTN_CANCEL_LONG, OBJPROP_BORDER_COLOR, clrNONE);
793: DB_PositionRow r;
803: DB_UpdatePositionStatus(_Symbol, (ENUM_TIMEFRAMES)_Period, "SHORT", active_short_trade_no,
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
DB_Key, DB_SetMetaInt, DB_UpdatePositionStatus, DeleteLinesandLabelsLong, DeleteLinesandLabelsShort, FormatCancelTradeMessage, SendDiscordMessage, TF_ToString, UI_CloseOnePositionAndNotify, UI_DeleteTradeLinesByTradeNo, UI_ParseTradePosFromButtonName, UI_TradesPanel_RebuildRows, UI_UpdateNextTradePosUI	event.mqh:97 if(UI_TradesPanel_OnChartEvent(id, lparam, dparam, sparam))

Verwendete MQL-Konstanten (UI)

- Objekt-Properties: OBJPROP_BGCOLOR, OBJPROP_BORDER_COLOR, OBJPROP_COLOR, OBJPROP_STATE

UI_ParseTradePosFromButtonName [Trades-Panel (UI links)]

```
bool UI_ParseTradePosFromButtonName(const string name,
                                    const string prefix,
                                    int &trade_no,
                                    int &pos_no)
```

Ort im Code: trades_panel.mqh Zeile 913 bis 933

Rueckgabe	Parameter (Typ name [default])
bool	const string name const string prefix int &trade_no int &pos_no

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(StringFind(name, prefix, 0) != 0)
- if(sep < 1)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
917: {
921: if(StringFind(name, prefix, 0) != 0)
922: return false;
926: if(sep < 1)
927: return false;
932: return (trade_no > 0 && pos_no > 0);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

(keine)	trades_panel.mqh:838 if(UI_ParseTradePosFromButtonName(sparam, TP_ROW_LONG_Cancel_PREFIX, trade_no, pos_no)) trades_panel.mqh:854 if(UI_ParseTradePosFromButtonName(sparam, TP_ROW_LONG_hitSL_PREFIX, trade_no, pos_no)) trades_panel.mqh:869 if(UI_ParseTradePosFromButtonName(sparam, TP_ROW_SHORT_Cancel_PREFIX, trade_no, pos_no)) trades_panel.mqh:884 if(UI_ParseTradePosFromButtonName(sparam, TP_ROW_SHORT_hitSL_PREFIX, trade_no, pos_no))

UI_IsPriceVisible [Trades-Panel (UI links)]

```
bool UI_IsPriceVisible(const double price, const int subwin = 0)
```

Ort im Code: trades_panel.mqh Zeile 938 bis 950

Rueckgabe	Parameter (Typ name [default])
bool	const double price const int subwin = 0

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(hi <= lo || hi == 0.0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
939: {
940: double pmin = ChartGetDouble(0, CHART_PRICE_MIN, subwin);
941: double pmax = ChartGetDouble(0, CHART_PRICE_MAX, subwin);
946: if(hi <= lo || hi == 0.0)  // Chart noch nicht “ready”
947: return true;
949: return (price >= lo && price <= hi);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	trades_panel.mqh:974 if(UI_IsPriceVisible(intended_price, subwin))

Verwendete MQL-Konstanten (UI)

- Chart-Properties: CHART_PRICE_MAX, CHART_PRICE_MIN

UI_ChartMidPrice [Trades-Panel (UI links)]

```
double UI_ChartMidPrice(const int subwin = 0)
```

Ort im Code: trades_panel.mqh Zeile 955 bis 967

Rueckgabe	Parameter (Typ name [default])
double	const int subwin = 0

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(hi <= lo || hi == 0.0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
956: {
957: double pmin = ChartGetDouble(0, CHART_PRICE_MIN, subwin);
958: double pmax = ChartGetDouble(0, CHART_PRICE_MAX, subwin);
963: if(hi <= lo || hi == 0.0)
964: return SymbolInfoDouble(_Symbol, SYMBOL_BID);
966: return (lo + hi) / 2.0; // entspricht “Chart-Höhe/2”
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	trades_panel.mqh:976 return UI_ChartMidPrice(subwin);

Verwendete MQL-Konstanten (UI)

- Chart-Properties: CHART_PRICE_MAX, CHART_PRICE_MIN

UI_DrawPriceOrMid [Trades-Panel (UI links)]

```
double UI_DrawPriceOrMid(const double intended_price, const int subwin = 0)
```

Ort im Code: trades_panel.mqh Zeile 972 bis 977

Rueckgabe	Parameter (Typ name [default])
double	const double intended_price const int subwin = 0

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_IsPriceVisible, UI_ChartMidPrice.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(UI_IsPriceVisible(intended_price, subwin))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
973: {
974: if(UI_IsPriceVisible(intended_price, subwin))
974: if(UI_IsPriceVisible(intended_price, subwin))
975: return intended_price;
975: return intended_price;
976: return UI_ChartMidPrice(subwin);
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

UI_ChartMidPrice, UI_IsPriceVisible	Trade Assistent V2.008.mq5:998 double entry_draw = UI_DrawPriceOrMid(rows[i].entry, 0);
	Trade Assistent V2.008.mq5:999 double sl_draw = UI_DrawPriceOrMid(rows[i].sl, 0);
	trades_panel.mqh:412 double entry_draw = UI_DrawPriceOrMid(rows[i].entry, 0);
	trades_panel.mqh:413 double sl_draw = UI_DrawPriceOrMid(rows[i].sl, 0);
	trades_panel.mqh:549 double entry_draw = UI_DrawPriceOrMid(rows[i].entry, 0);
	trades_panel.mqh:550 double sl_draw = UI_DrawPriceOrMid(rows[i].sl, 0);

UI_DeleteTradeLinesByTradeNo [Trades-Panel (UI links)]

```
int UI_DeleteTradeLinesByTradeNo(const int trade_no)
```

Ort im Code: trades_panel.mqh Zeile 988 bis 1024

Kommentar direkt darueber: Löscht ALLE suf-Linien (Entry/SL Long/Short + optional _TAG) für EINEN Trade. Trifft Namen wie: Entry_Long_<trade>_<pos>, SL_Long_<trade>_<pos>, ... Rückgabe: Anzahl gelöschter Objekte.

Rueckgabe	Parameter (Typ name [default])
int	const int trade_no

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Berechnung

2 2. Prefix + "_" + trade_no + "_": UI/Chart-Objekte anlegen/aktualisieren/verschieben

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(trade_no <= 0)
- for(int i = total - 1; i >= 0; i--)
- if(name == "")
- if(match)
- if(ObjectDelete(0, name))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
 990: if(trade_no <= 0)
 991: return 0;
1004: for(int i = total - 1; i >= 0; i--)
1007: if(name == "")
1016: if(match)
1018: if(ObjectDelete(0, name))
1023: return deleted;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	trades_panel.mqh:776 int d = UI_DeleteTradeLinesByTradeNo(active_long_trade_no); trades_panel.mqh:821 int d = UI_DeleteTradeLinesByTradeNo(active_short_trade_no);

ui_registry.mqh

Zentrales Registry fuer alle erzeugten Chart-Objekte (sauberes Loeschen/Adoptieren)

Funktionen in dieser Datei (9): UI_Reg_IndexOf, UI_Reg_Add, UI_Reg_Remove, UI_Reg_DeleteOne, UI_Reg_DeleteAll, UI_Reg_DeleteByPrefix, UI_Reg_DeleteBySuffix, UI_Reg_AdoptByPrefix, UI_Reg_AdoptBySuffix

UI_Reg_IndexOf [UI / Chart-Objekte]

```
int UI_Reg_IndexOf(const string name)
```

Ort im Code: ui_registry.mqh Zeile 17 bis 24

Kommentar direkt darueber: Helpers

Rueckgabe	Parameter (Typ name [default])
int	const string name

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Berechnung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int i=0; i<n; i++)
- if(g_UI_Registry[i] == name)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
18: {
19: int n = ArraySize(g_UI_Registry);
20: for(int i=0; i<n; i++)
21: if(g_UI_Registry[i] == name)
22: return i;
23: return -1;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	ui_registry.mqh:35 if(UI_Reg_IndexOf(name) >= 0) ui_registry.mqh:48 int idx = UI_Reg_IndexOf(name);

UI_Reg_Add [UI / Chart-Objekte]

```
void UI_Reg_Add(const string name)
```

Ort im Code: ui_registry.mqh Zeile 29 bis 41

Rueckgabe	Parameter (Typ name [default])
void	const string name

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1. 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_Reg_IndexOf.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(name == "")

- if(UI_Reg_IndexOf(name) >= 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
30: {
31: if(name == "")
31: if(name == "")
32: return;
35: if(UI_Reg_IndexOf(name) >= 0)
36: return;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

UI_Reg_IndexOf	Trade Assistent V2.008.mq5:1237 UI_Reg_Add(name);//Speichere Object im Array zum späteren löschen
	Trade Assistent V2.008.mq5:1311 UI_Reg_Add(tag_name);
	gui_elemente.mqh:235 UI_Reg_Add(objName); // Speichere Object im Array zum späteren löschen
	gui_elemente.mqh:266 UI_Reg_Add(LABEL_NAME); // Speichere Object im Array zum späteren löschen
	gui_elemente.mqh:413 UI_Reg_Add(objName); // Speichere Object im Array zum späteren löschen
	gui_elemente.mqh:446 UI_Reg_Add(objName); // Speichere Object im Array zum späteren löschen
	gui_elemente.mqh:465 UI_Reg_Add(SENDTRADEBTN); // Speichere Object im Array zum späteren löschen
	gui_elemente.mqh:489 UI_Reg_Add(TRNB); // Speichere Object im Array zum späteren löschen
	... (9 weitere)

UI_Reg_Remove [UI / Chart-Objekte]

```
bool UI_Reg_Remove(const string name)
```

Ort im Code: ui_registry.mqh Zeile 46 bis 58

Rueckgabe	Parameter (Typ name [default])
bool	const string name

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_Reg_IndexOf.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(idx < 0)
- for(int i=idx; i<n-1; i++)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
47: {
48: int idx = UI_Reg_IndexOf(name);
49: if(idx < 0)
50: return false;
53: for(int i=idx; i<n-1; i++)
57: return true;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

UI_Reg_IndexOf	ui_registry.mqh:68 UI_Reg_Remove(name); // auch wenn nicht existiert -> sauber halten

UI_Reg_DeleteOne [UI / Chart-Objekte]

```
bool UI_Reg_DeleteOne(const string name)
```

Ort im Code: ui_registry.mqh Zeile 61 bis 70

Kommentar direkt darueber: Loescht Objekt (falls existiert) + entfernt es aus Registry

Rueckgabe	Parameter (Typ name [default])
bool	const string name

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben. Wichtige Aufrufe: UI_Reg_Remove.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(ObjectFind(0, name) >= 0)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
62: {
63: bool ok = true;
65: if(ObjectFind(0, name) >= 0)
65: if(ObjectFind(0, name) >= 0)
66: ok = ObjectDelete(0, name);
69: return ok;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)

UI_Reg_Remove	gui_elemente.mqh:324 UI_Reg_DeleteOne(name);
	gui_elemente.mqh:345 UI_Reg_DeleteOne(name);
	gui_elemente.mqh:394 if (UI_Reg_DeleteOne(name))
	gui_elemente.mqh:520 UI_Reg_DeleteOne(SENDTRADEBTN);
	gui_elemente.mqh:521 UI_Reg_DeleteOne(TRNB);
	gui_elemente.mqh:522 UI_Reg_DeleteOne(POSNB);
	gui_elemente.mqh:568 UI_Reg_DeleteOne(SabioEntry);
	gui_elemente.mqh:569 UI_Reg_DeleteOne(SabioSL);
	... (9 weitere)

UI_Reg_DeleteAll [UI / Chart-Objekte]

```
int UI_Reg_DeleteAll()
```

Ort im Code: ui_registry.mqh Zeile 77 bis 99

Kommentar direkt darueber: Loescht ALLE registrierten Objekte (in Reverse-Reihenfolge)

Rueckgabe	Parameter (Typ name [default])
int	(keine)

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben; Logging/Benachrichtigung

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int i = ArraySize(g_UI_Registry) - 1; i >= 0; i--)
- if(name == "")
- if(ObjectFind(0, name) >= 0)
- if(ObjectDelete(0, name))

Wichtige Codezeilen (zum Mitlesen im Editor)

```
81: for(int i = ArraySize(g_UI_Registry) - 1; i >= 0; i--)
84: if(name == "")
87: if(ObjectFind(0, name) >= 0)
89: if(ObjectDelete(0, name))
92: Print("Delete Objekt: "+name);
98: return deleted;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	Trade Assistent V2.008.mq5:734 int deleted = UI_Reg_DeleteAll();

UI_Reg_DeleteByPrefix [UI / Chart-Objekte]

```
int UI_Reg_DeleteByPrefix(const string prefix)
```

Ort im Code: ui_registry.mqh Zeile 102 bis 126

Kommentar direkt darueber: Löscht registrierte Objekte nach Prefix (und entfernt sie aus Registry)

Rueckgabe	Parameter (Typ name [default])
int	const string prefix

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int i = ArraySize(g_UI_Registry) - 1; i >= 0; i--)
- if(StringFind(name, prefix, 0) == 0)
- if(ObjectFind(0, name) >= 0)
- if(ObjectDelete(0, name))
- for(int k=i; k<n-1; k++)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
106: for(int i = ArraySize(g_UI_Registry) - 1; i >= 0; i--)
109: if(StringFind(name, prefix, 0) == 0)
111: if(ObjectFind(0, name) >= 0)
113: if(ObjectDelete(0, name))
119: for(int k=i; k<n-1; k++)
125: return deleted;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	(keine Treffer im Projekt)

UI_Reg_DeleteBySuffix [UI / Chart-Objekte]

```
int UI_Reg_DeleteBySuffix(const string suffix)
```

Ort im Code: ui_registry.mqh Zeile 129 bis 155

Kommentar direkt darueber: Löscht registrierte Objekte nach Suffix (und entfernt sie aus Registry)

Rueckgabe	Parameter (Typ name [default])
int	const string suffix

Seiteneffekte

- Aendert Chart-Objekte (Linien/Buttons/Labels/Panel).

Ablauf (Schritt fuer Schritt)

1 1. Start: UI/Chart-Objekte anlegen/aktualisieren/verschieben

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int i = ArraySize(g_UI_Registry) - 1; i >= 0; i--)
- if(nLen >= sufLen && StringSubstr(name, nLen - sufLen, sufLen) == suffix)
- if(ObjectFind(0, name) >= 0)
- if(ObjectDelete(0, name))
- for(int k=i; k<n-1; k++)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
134: for(int i = ArraySize(g_UI_Registry) - 1; i >= 0; i--)
139: if(nLen >= sufLen && StringSubstr(name, nLen - sufLen, sufLen) == suffix)
141: if(ObjectFind(0, name) >= 0)
143: if(ObjectDelete(0, name))
148: for(int k=i; k<n-1; k++)
154: return deleted;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
(keine)	(keine Treffer im Projekt)

UI_Reg_AdoptByPrefix [UI / Chart-Objekte]

```
int UI_Reg_AdoptByPrefix(const string prefix)
```

Ort im Code: ui_registry.mqh Zeile 161 bis 182

Kommentar direkt darueber: Adopt: nimmt bereits existierende Chart-Objekte per Prefix in die Registry auf. Damit kannst du beim EA-Start "alte Leichen" einsammeln und später sauber löschen. Rückgabe: Anzahl neu adoptierter Objekte.

Rueckgabe	Parameter (Typ name [default])
int	const string prefix

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_Reg_Add.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- for(int i = 0; i < total; i++)
- if(name == "")

- if(StringFind(name, prefix, 0) == 0)
- if(ArraySize(g_UI_Registry) > before)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
162: {
166: for(int i = 0; i < total; i++)
169: if(name == "")
172: if(StringFind(name, prefix, 0) == 0)
176: if(ArraySize(g_UI_Registry) > before)
181: return adopted;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_Add	(keine Treffer im Projekt)

UI_Reg_AdoptBySuffix [UI / Chart-Objekte]

```
int UI_Reg_AdoptBySuffix(const string suffix)
```

Ort im Code: ui_registry.mqh Zeile 188 bis 216

Kommentar direkt darueber: Adopt: per Suffix (z.B. "_TAG" oder "_12_1"). Rückgabe: Anzahl neu adoptierter Objekte.

Rueckgabe	Parameter (Typ name [default])
int	const string suffix

Seiteneffekte

- Keine externen Seiteneffekte (nur Berechnung/Formatierung).

Ablauf (Schritt fuer Schritt)

1 1. Start: Interne Logik / Hilfsfunktionen. Wichtige Aufrufe: UI_Reg_Add.

Kontrollfluss (erste Bedingungen/Schleifen in Reihenfolge)

- if(sufLen == 0)
- for(int i = 0; i < total; i++)
- if(name == "")
- if(nLen < sufLen)
- if(StringSubstr(name, nLen - sufLen, sufLen) == suffix)
- if(ArraySize(g_UI_Registry) > before)

Wichtige Codezeilen (zum Mitlesen im Editor)

```
192: if(sufLen == 0)
193: return 0;
196: for(int i = 0; i < total; i++)
199: if(name == "")
203: if(nLen < sufLen)
206: if(StringSubstr(name, nLen - sufLen, sufLen) == suffix)
210: if(ArraySize(g_UI_Registry) > before)
215: return adopted;
```

Abhaengigkeiten

Ruft auf (intern)	Wird verwendet von (Callsites)
UI_Reg_Add	(keine Treffer im Projekt)