

Verfügbare Variablen und Datenstrukturen

	Variable	Typ-Referenz	Hintergründe	Beispiel-Zugriffe
1	varApplication	<u>TApplication</u>	<p>Zugriff auf das Applikationsobjekt der Anwendung wie z.B.</p> <ul style="list-style-type: none"> • Application.ProcessMessages (Abarbeitung von UI-Refreshes) 	
2	varParamCombobox	TAdvOfficeComboBox	<p>Zugriff auf eine Auswahlbox, die für den Nutzer angezeigt wird, um für das Script Auswahloptionen für den Nutzer zur Verfügung zu stellen. In der Susan-Levermann-Strategie wird über diese eine Liste der verfügbaren Watchlisten angezeigt und nutzbar gemacht. Für Nutzer der Strategie steht der entsprechende Source-Code zur Verfügung. z.B. für</p> <ul style="list-style-type: none"> • Watchlisten • Depots • Marktsegmente 	<p>Auswahlbox für den Nutzer füllen mit allen Watchlisten-Einträgen, danach mit allen MarktSegment-Einträgen und final als ersten Eintrag (Index=0) "Kein-Watchlisten-Filter" einfügen.</p> <pre> varParamCombobox for llx:=1 to varWatchListCount varParamCombobox.Items.Add(WatchList[llx].Name) for llx:=1 to varSegmentCount varParamCombobox.Items.Add(MarketSegment[llx].Name) varParamCombobox.Items.Insert(0, "Kein-Watchlisten-Filter") </pre>

	Variable	Typ-Referenz	Hintergründe	Beispiel-Zugriffe
3	varParamCheckbox	TAdvOfficeCheckBox	<p>Zugriff auf eine Checkbox, die für den Nutzer angezeigt wird, um für das Script Auswahloptionen für den Nutzer zur Verfügung zu stellen. In der Susan-Levermann-Strategie wird diese für "Detaillierte Konfiguration" als Option genutzt.</p>	<p>Aktiviere überhaupt die Auswahlbox für den Nutzer. Im Standard wird die Auswahlbox überhaupt nicht angezeigt. Hinweis: Dies sollte vorzugsweise in einer "doInit-Methode" passieren.</p> <div><p>varParamCheckbox varParamCheckbox varParamCheckbox</p></div>

	Variable	Typ-Referenz	Hintergründe	Beispiel-Zugriffe
4	varLog	TAdvListView	<p>Die Primärausgabe erfolgt über das varLog-Objekt vom Typ TAdvListView. Es ist eine Komponente zur hierarchischen, gruppierten Tabellenausgabe von Informationen. Die Objekte besitzt Columns und eine GroupView-Eigenschaft zur automatischen Gruppierung von Einträgen. Um einen neuen Log-Eintrag zu erzeugen, kann über varLog.Items.Add eine neue Zeile eingefügt werden. Das dabei zurückgegebene Objekt ist vom Typ TListItem, was eine Caption, eine GroupID, ImageIndex, Checked-Eigenschaft besitzt.</p>	<p>Kopfzeile der Logausgabe gestalten:</p> <pre>(* BeginUpdate/End varLog.BeginUpdate varLog.Items.Clear; (* Header für das Log varLog.Columns.Clear IColumn:=varLog.Columns.New IColumn:=varLog.Columns.New IColumn:=varLog.Columns.New varLog.GroupView:=varLog.GroupView varLog.EndUpdate;</pre> <p>Konkretes Logging:</p> <pre>function doLogWithCaption begin result:=varLog.Items.New with result as TListItem begin Caption:=logMessage subitems.add(logMessage, groupID:=0; end; end;</pre> <p>Automatische Sortierung:</p> <pre>varLog.SortColumn:=varLog.SortColumn varLog.SortDirection:=varLog.SortDirection varLog.SortType:=varLog.SortType varLog.Sort;</pre>
5	varPanelStatus	TAdvOfficeStatusPanel	<p>Status-Updates für den Nutzer über die Statuszeile. Dabei können primär Textnachrichten über die varPanelStatus.Text Eigenschaft ausgegeben werden.</p>	

	Variable	Typ-Referenz	Hintergründe	Beispiel-Zugriffe
6	varPanelProgress	TAdvOfficeStatusPanel	Status-Updates für den Nutzer über die Statuszeile. Dabei liegt der Fokus auf der Fortschrittsanzeige.	<div> varPanelProgress.Pr varPanelProgress.Pr varPanelProgress.Pr </div>
7	varIsCanceled	Boolean	Der Nutzer kann ein Script in der Berechnung abbrechen. Dabei wird das Script nicht hart abgebrochen, sondern zunächst nur die Variable auf "true" gesetzt. Innerhalb des Scripts sollte diese daher immer abgefragt werden, um einen geordneten Abbruch zu realisieren.	Haupt-Iteration über alle Aktien, wobei aber bei jeder Iteration explizit die Abbruchvariable "varIsCanceled" geprüft wird, um geordnet abzubrechen: <div> (* Durchlauf alle T for idxStock:=1 to begin </div>
8	varStocks	<u>TStocks</u>	Zugriff auf alle Assets/Stocks in ShareHolder. Der Zugriff kann dabei sowohl lesend, als auch schreibend erfolgen. So können z.B. automatisiert Namensanpassungen vorgenommen werden.	Iteration über alle Stocks: <div> (* Durchlauf alle Tite for idxStock:=1 to v begin itStock:=varStoc </div> Zugriff auf Stock-Variablen, um diese automatisiert zu verändern: <div> (* Zugriff entspricht IStockVar:=IStock.S if (IStockVar<>nil) t begin ... </div>

	Variable	Typ-Referenz	Hintergründe	Beispiel-Zugriffe
9	varIndicators	<u>TIndicatorGroups</u>	Mittels Indicator.Parameter.It ems[1].Wert:=10; könnten die Parameter des Indikators verändert werden. Die Bedeutung der Parameter muss pro Indikator nachgeschaut werden z.B. über Einstellungen / Indikatorengruppen / Doppelklick auf den Indikator.	<div>itIndicator:=varInc</div>
10	varSegments	<u>TSegments</u>	Zugriff auf alle Marktsegmente z.B. DAX, MDAX etc.	
11	varTransactions	<u>TTransactions</u>		

	Variable	Typ-Referenz	Hintergründe	Beispiel-Zugriffe
12	varWatchlists	<u>TWatchlists</u>	Zugriff auf die internen Watchlisten.	<p>Neue Watchliste erzeugen, wenn nicht zuvor angelegt mit dem Namen "Wa:ETF-Momentum-Sortlist"</p> <pre> if (fFirst) and (fUseS begin fWatchlist:=varWa fWatchlist.Clear(); fFirst:=false; end; </pre> <p>Eintrag hinzufügen</p> <pre> if (fWatchlist<>nil) begin if (not fUseSepar fWatchItem:=fV begin fWatchItem:=T fWatchlist.List.A end; if (fWatchItem<> begin fWatchItem.Co //itWatchItem.C fWatchItem.Po end; end; </pre>
13	varDepot	<u>TCalculatedDepotltem</u>	Zugriff auf alle berechneten Depot-Positionen mit Stückzahl, gemittelten Preis, letzter Transaktion etc.	
14	varStops	<u>TStopRates</u>	Zugriff auf alle definierten Stopps im Programm für Titel. Dabei enthält die Liste nur definierte Stopps d.h. Titel müssen keine Stopps haben. Stopps sind unabhängig von Depotpositionen.	

	Variable	Typ-Referenz	Hintergründe	Beispiel-Zugriffe
15	varProgramSettings	<u>TProgramSettings</u>	Zugriff auf alle internen Programmeinstellungen wie Schriftgrößen, Kalkulationsbasis etc.	Verwende die offizielle Programmeinstellung für die KGV-Berechnung d.h. aktuelles Basisjahr (0), kommendes Jahr etc. <div>function calcPointsK begin Result:=itStock.KG end;</div>
16	varStockExchanges	<u>TStockExchanges</u>	Zugriff auf die definierten Börsen wie F, Nasdaq, Xetra	
17	varINetVars	<u>TINetVars</u>	Zugriff auf die definierten Aktualisierungs-Internetvariablen	
18	varINetAddrs	<u>TINetAddrs</u>	Zugriff auf die definierten Aktualisierungs-Internetadressen	
19	varAssets	<u>TAssets</u>	Zugriff auf die definierten Asset-Klassen	
20	varStrategy	<u>TStrategy</u>	Zugriff auf alle definierten Strategien	
21	varNNetze	<u>TNets</u>	Zugriff auf alle definierten Vorhersage-Modelle	
22	varAccounts	<u>TAccounts</u>	Zugriff auf alle Konten	
23	varImpFormate	<u>TImpFormats</u>	Zugriff auf alle Importformate	
24	varTradeMethods	<u>TTradeMethods</u>		
25	varAssessments	<u>TAssessments</u>		
26	varSparplaene	<u>TDepotSavingPlans</u>	Zugriff auf die Sparpläne	
27	varINetUpdateGroups	<u>TINetUpdateGroups</u>	Zugriff auf die Kursaktualisierungsgruppen	

Registrierung von Variablen und Kontext im Detail für das Scripting-Studio

Version 19.3.2

```
with Scripter do
begin
  LibOptions.SearchPath.Add(IDEEngine.BasePath);
  LibOptions.SourceFileExt := '.script';
  LibOptions.CompiledFileExt := '.psc';

  Scripter.OptionExplicit := false;
  logList.Items.Clear;
  Scripter.AddObject('varApplication', Application);
  Scripter.AddObject('varParamCombobox', fAdvCombobox);
  Scripter.AddObject('varParamCheckbox', fAdvCheckbox);

  Scripter.DefineClassByRTTI(TApplication);
  Scripter.DefineClassByRTTI(TBasisobject); // ,roInclude,false,'TBasisObject',[mvPublic,mvPublished]
  Scripter.DefineClassByRTTI(TAdvListView);
  Scripter.DefineClassByRTTI(TListItems);
  Scripter.DefineClassByRTTI(TStrings);
  Scripter.DefineClassByRTTI(TListItem);
  Scripter.DefineClassByRTTI(TListGroup);
  Scripter.DefineClassByRTTI(TListGroups);
  Scripter.DefineClassByRTTI(TStringList);

  Scripter.AddObject('varLog', flogList);
  Scripter.DefineClassByRTTI(TListColumns);
  Scripter.DefineClassByRTTI(TListColumn);
  Scripter.AddObject('varPanelStatus', fStatusPanel);
```



```
Scripter.AddObject('varPanelProgress', fProgressPanel);
Scripter.AddVariable('varIsCanceled', flsCanceled);

Scripter.DefineClassByRTTI(TStocks);
Scripter.DefineClassByRTTI(TAsset);
Scripter.DefineClassByRTTI(TAssets);
Scripter.DefineClassByRTTI(TAutoImports);
Scripter.DefineClassByRTTI(TCalculatedDepotAccount);
Scripter.DefineClassByRTTI(TCalculatedDepotAccounts);
Scripter.DefineClassByRTTI(TCalculatedDepotItem);
Scripter.DefineClassByRTTI(TCalculatedDepotItems);
Scripter.DefineClassByRTTI(TCandle);
Scripter.DefineClassByRTTI(TCandleFormation);
Scripter.DefineClassByRTTI(TChartIndicator);
Scripter.DefineClassByRTTI(TDepotSavingPlan);
Scripter.DefineClassByRTTI(TDepotSavingPlans);
Scripter.DefineClassByRTTI(TDynamicFilter);
Scripter.DefineClassByRTTI(TDynamicFilterCondition);
Scripter.DefineClassByRTTI(TDynamicFilterConditions);
Scripter.DefineClassByRTTI(TDynamicFilterConditionTree);
Scripter.DefineClassByRTTI(TDynamicFilterConditionTrees);
Scripter.DefineClassByRTTI(TDynamicFilterResult);
Scripter.DefineClassByRTTI(TDynamicFilterResults);
Scripter.DefineClassByRTTI(TDynamicFilters);
Scripter.DefineClassByRTTI(TEnhancedList);
Scripter.DefineClassByRTTI(TFilterTaipan);
Scripter.DefineClassByRTTI(TFilterTaipanItem);
Scripter.DefineClassByRTTI(THelperAverage);
Scripter.DefineClassByRTTI(THTMLParser);
Scripter.DefineClassByRTTI(TIAroon);
Scripter.DefineClassByRTTI(TIBollinger);
Scripter.DefineClassByRTTI(TICandleFormationen);
Scripter.DefineClassByRTTI(TICandleFormationenCache);
Scripter.DefineClassByRTTI(TICCI);
Scripter.DefineClassByRTTI(TIChaikin);
Scripter.DefineClassByRTTI(TICoppock);
Scripter.DefineClassByRTTI(TIDMI);
Scripter.DefineClassByRTTI(TIDSSStochastik);
Scripter.DefineClassByRTTI(TIForceIndex);
Scripter.DefineClassByRTTI(TIGDEMA);
```

```
Scripter.DefineClassByRTTI(TIGDUmsatz);
Scripter.DefineClassByRTTI(TIHistVol);
Scripter.DefineClassByRTTI(TIMACD);
Scripter.DefineClassByRTTI(TIMFI);
Scripter.DefineClassByRTTI(TIMomentum);
Scripter.DefineClassByRTTI(TImpFormat);
Scripter.DefineClassByRTTI(TImpFormats);
Scripter.DefineClassByRTTI(TIndicator);
Scripter.DefineClassByRTTI(TIndicatorGroup);
Scripter.DefineClassByRTTI(TIndicatorGroups);
Scripter.DefineClassByRTTI(TIndicatorParam);
Scripter.DefineClassByRTTI(TIndicatorParams);
Scripter.DefineClassByRTTI(TIndicatorSignals);
Scripter.DefineClassByRTTI(TINegativeVolumeIndex);
Scripter.DefineClassByRTTI(TINetAddr);
Scripter.DefineClassByRTTI(TINetAddrs);
Scripter.DefineClassByRTTI(TINetUpdateGroup);
Scripter.DefineClassByRTTI(TINetUpdateGroups);
Scripter.DefineClassByRTTI(TINetVar);
Scripter.DefineClassByRTTI(TINetVars);
Scripter.DefineClassByRTTI(TINewHigh);
Scripter.DefineClassByRTTI(TINewLow);
Scripter.DefineClassByRTTI(TINNkorrelation);
Scripter.DefineClassByRTTI(TINNPrognose);
Scripter.DefineClassByRTTI(TInternetProperties);
Scripter.DefineClassByRTTI(TIntSignalIndicatorCache);
Scripter.DefineClassByRTTI(TIOnBalanceVolume);
Scripter.DefineClassByRTTI(TIPFE);
Scripter.DefineClassByRTTI(TIPositiveVolumeIndex);
Scripter.DefineClassByRTTI(TIPSAR);
Scripter.DefineClassByRTTI(TIPvt);
Scripter.DefineClassByRTTI(TIRAVI);
Scripter.DefineClassByRTTI(TIRMI);
Scripter.DefineClassByRTTI(TIRSI);
Scripter.DefineClassByRTTI(TIRSL);
Scripter.DefineClassByRTTI(TIRWI);
Scripter.DefineClassByRTTI(TISignalIndicator);
Scripter.DefineClassByRTTI(TISignalIndicatorCache);
Scripter.DefineClassByRTTI(TISignalIndicators);
Scripter.DefineClassByRTTI(TIStdDev);
```

```
Scripter.DefineClassByRTTI(TIStochastik);
Scripter.DefineClassByRTTI(TITrix);
Scripter.DefineClassByRTTI(TITRWinkel);
Scripter.DefineClassByRTTI(TITSF);
Scripter.DefineClassByRTTI(TIVHF);
Scripter.DefineClassByRTTI(TIVolumeNotis);
Scripter.DefineClassByRTTI(TIVolumePriceTrend);
Scripter.DefineClassByRTTI(TIWilderVol);
Scripter.DefineClassByRTTI(TParamEnhancedList);
Scripter.DefineClassByRTTI(TProfitStop);
Scripter.DefineClassByRTTI(TProgramSettings);
Scripter.DefineClassByRTTI(TProperties);
Scripter.DefineClassByRTTI(TSegment);
Scripter.DefineClassByRTTI(TSegments);
Scripter.DefineClassByRTTI(TSignalItem);
Scripter.DefineClassByRTTI(TSplit);
Scripter.DefineClassByRTTI(TSplits);
Scripter.DefineClassByRTTI(TStock);
Scripter.DefineClassByRTTI(TStockExchange);
Scripter.DefineClassByRTTI(TStockExchanges);
Scripter.DefineClassByRTTI(TStockNetValue);
Scripter.DefineClassByRTTI(TStockNetValues);
Scripter.DefineClassByRTTI(TStockNews);
Scripter.DefineClassByRTTI(TStockNewsList);
Scripter.DefineClassByRTTI(TStockProfile);
Scripter.DefineClassByRTTI(TStockProfiles);
Scripter.DefineClassByRTTI(TStocks);
Scripter.DefineClassByRTTI(TStockVariable);
Scripter.DefineClassByRTTI(TStockVariables);
Scripter.DefineClassByRTTI(TStopRate);
Scripter.DefineClassByRTTI(TStopRates);
Scripter.DefineClassByRTTI(TStrategy);
Scripter.DefineClassByRTTI(TStringParser);
Scripter.DefineClassByRTTI(TTaipanCatalogItem);
Scripter.DefineClassByRTTI(TTaipanCatalogItems);
Scripter.DefineClassByRTTI(TTradeMethod);
Scripter.DefineClassByRTTI(TTradeMethods);
Scripter.DefineClassByRTTI(TTradingSystem);
Scripter.DefineClassByRTTI(TTradingSystemMetricHelper);
Scripter.DefineClassByRTTI(TTradingSystemMetrics);
```

```
Scripter.DefineClassByRTTI(TTradingSystemThread);
Scripter.DefineClassByRTTI(TTradingSystemTrades);
Scripter.DefineClassByRTTI(TTrailingStop);
Scripter.DefineClassByRTTI(TWatchlist);
Scripter.DefineClassByRTTI(TWatchlistItem);
Scripter.DefineClassByRTTI(TWatchlistItems);
Scripter.DefineClassByRTTI(TWatchlists);
Scripter.DefineClassByRTTI(THistoryItem);
Scripter.DefineClassByRTTI(TKurse);
Scripter.AddObject('varStocks', Stocks);

Scripter.AddConstant('ctTrendfolger', ctTrendfolger);
Scripter.AddConstant('typMA', typMA);
Scripter.AddConstant('cRMI', cRMI);
Scripter.AddConstant('cCandlesticks', cCandlesticks);
Scripter.AddConstant('cMA', cMA);
Scripter.AddConstant('cBollinger', cBollinger);
Scripter.AddConstant('cPSAR', cPSAR);
Scripter.AddConstant('cGDUmsatz', cGDUmsatz);
Scripter.AddConstant('cMomentum', cMomentum);
Scripter.AddConstant('cRSI', cRSI);
Scripter.AddConstant('cStochastik', cStochastik);
Scripter.AddConstant('cChaikin', cChaikin);
Scripter.AddConstant('cDSStochastik', cDSStochastik);
Scripter.AddConstant('cMFI', cMFI);
Scripter.AddConstant('cCoppock', cCoppock);
Scripter.AddConstant('cRSL', cRSL);
Scripter.AddConstant('cMACD', cMACD);
Scripter.AddConstant('cTRIX', cTRIX);
Scripter.AddConstant('cCCI', cCCI);
Scripter.AddConstant('cRMI', cRMI);
Scripter.AddConstant('cPFE', cPFE);
Scripter.AddConstant('cTSF', cTSF);
Scripter.AddConstant('cPVT', cPVT);
Scripter.AddConstant('cNewHigh', cNewHigh);
Scripter.AddConstant('cNewLow', cNewLow);
Scripter.AddConstant('cStdDev', cStdDev);
Scripter.AddConstant('cHistVol', cHistVol);
Scripter.AddConstant('cVHF', cVHF);
Scripter.AddConstant('cWilderVOI', cWilderVOI);
```

```
Scripter.AddConstant('cADX', cADX);
Scripter.AddConstant('cRAVI', cRAVI);
Scripter.AddConstant('cTRWinkel', cTRWinkel);
Scripter.AddConstant('cRWI', cRWI);
Scripter.AddConstant('cAroon', cAroon);
Scripter.AddConstant('cNNKorrelation', cNNKorrelation);
Scripter.AddConstant('cNNPrognose', cNNPrognose);
Scripter.AddConstant('cForceIndex', cForceIndex);
Scripter.AddConstant('cOnBalanceVolume', cOnBalanceVolume);
Scripter.AddConstant('cVolumePriceTrend', cVolumePriceTrend);
Scripter.AddConstant('cNegativeVolumeIndex', cNegativeVolumeIndex);
Scripter.AddConstant('cPositivVolumeIndex', cPositivVolumeIndex);
Scripter.AddConstant('cVolumeNotisV', cVolumeNotisV);
```

```
Scripter.DefineClassByRTTI(TIndicator);
Scripter.DefineClassByRTTI(TChartindicator);
Scripter.DefineClassByRTTI(TSignalIndicator);
Scripter.DefineClassByRTTI(TIRMI);
Scripter.DefineClassByRTTI(TIndicatorGroups);
Scripter.DefineClassByRTTI(TIndicatorParams);
Scripter.DefineClassByRTTI(TIndicatorParam);
Scripter.DefineClassByRTTI(TIndicatorSignals);
Scripter.AddObject('varIndicators', IndicatorGroups);
```

```
Scripter.DefineClassByRTTI(TSegments);
Scripter.DefineClassByRTTI(TSegment);
Scripter.AddObject('varSegments', Segments);
```

```
Scripter.DefineClassByRTTI(TTransaction);
Scripter.DefineClassByRTTI(TTransactions);
Scripter.DefineClassByRTTI(TAssessments);
Scripter.AddObject('varTransactions', Transactions);
```

```
Scripter.DefineClassByRTTI(TWatchlist);
Scripter.DefineClassByRTTI(TWatchlists);
Scripter.DefineClassByRTTI(TWatchlistItems);
Scripter.DefineClassByRTTI(TWatchlistItem);
Scripter.AddObject('varWatchlists', Watchlists);
```

```
Scripter.DefineClassByRTTI(TCalculatedDepotItems);
Scripter.DefineClassByRTTI(TCalculatedDepotItem);
Scripter.AddObject('varDepot', CalculatedDepotItems);
Scripter.DefineClassByRTTI(TStoprate);
Scripter.AddObject('varStops', StopRates);
Scripter.AddObject('varProgramSettings', ProgramSettings);

Scripter.AddObject('varStockExchanges', StockExchanges);
Scripter.AddObject('varINetVars', INetVars);
Scripter.AddObject('varINetAddrs', INetAddrs);
Scripter.AddObject('varAssets', Assets);
Scripter.AddObject('varStrategy', Strategy);
Scripter.AddObject('varNNetze', NNetze);
Scripter.AddObject('varAccounts', Accounts);
Scripter.AddObject('varImpFormate', ImpFormate);
Scripter.AddObject('varTradeMethods', TradeMethods);
Scripter.AddObject('varAssessments', Assessments);
Scripter.AddObject('varSparplaene', Sparplaene);
Scripter.AddObject('varINetUpdateGroups', INetUpdateGroups);
end;
```

Alle TAdv* Objekte sind genauer durch den Komponentenhersteller beschrieben und können aus Lizenzgründen auch von mir nicht genauer aufgegriffen werden.

Siehe hierzu unter: <https://www.tmssoftware.com/site/tmspack.asp>

Revision #1

Created 19 June 2022 07:46:49 by Jens Werschmoeller

Updated 19 June 2022 07:47:25 by Jens Werschmoeller