GUPTA

Roadshow 2013

Team Developer 6.2

Helmut Reimann

Agenda

- Was ist neu im Team Developer 6.2
  - GUI
  - SAL
  - Compiler
    - 64Bit
    - Debugging
  - Reports

Neue GUI & API in TD6.2

- Andocken von Dialogen in Dialoge
- Neue *Progress Bar*
- Neue *Ribbon Bar*
- Rechtschreibprüfung (in .NET)
- MDI Hintergrund als Schema

Neue GUI & API in TD6.2

- Erweiterungen *Named Toolbar*
- Erweiterungen für das *Grid Control*
- Erweiterungen für das *Tree Control*
- Erweiterungen im *Date/Time Picker*

Web Services

- Erzeugen von .NET Web Services in Team Developer 6.2
  - Neue Web Service Klasse

Compiler Erweiterungen

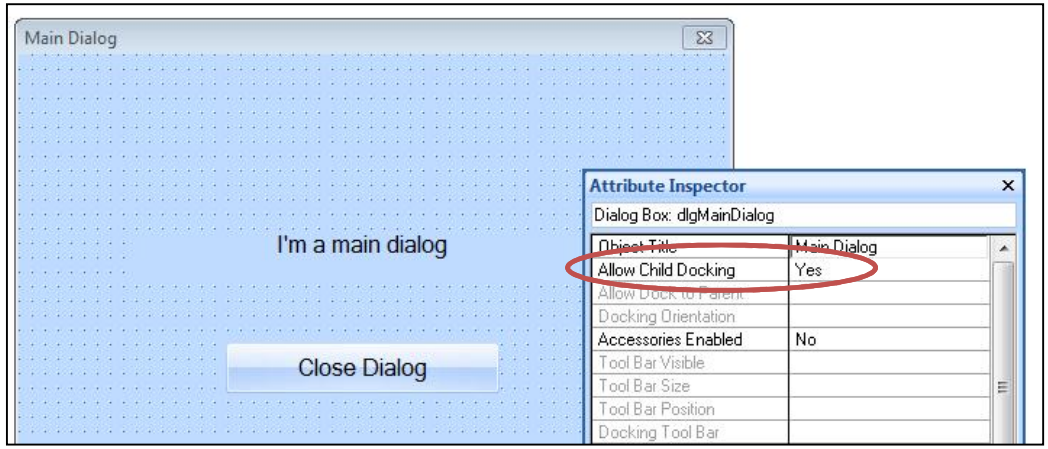
- Debuggen
  - Web Services
  - Assembly
    - TD 6.2 Assemblies in Visual Studio
    - Visual Studio Assemblies in TD 6.2
- 64-Bit Anwendungen in .NET

Report Builder Erweiterung

- **Unterstützung des neuen Chart Controls in Report Builder 6.2**

Andocken von Dialogen in Dialoge

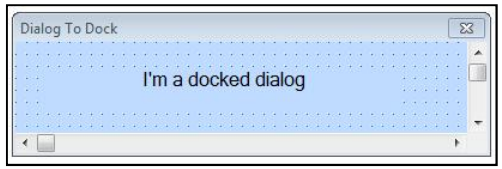
- **Attribute für den „Haupt-Dialog“**



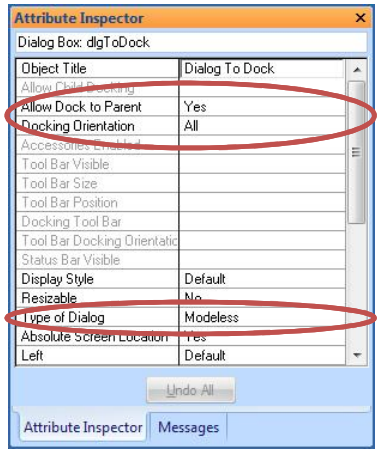
Allow Child Docking = Yes

Andocken von Dialogen in Dialoge

- **Attribute für den „Kind-Dialog“**



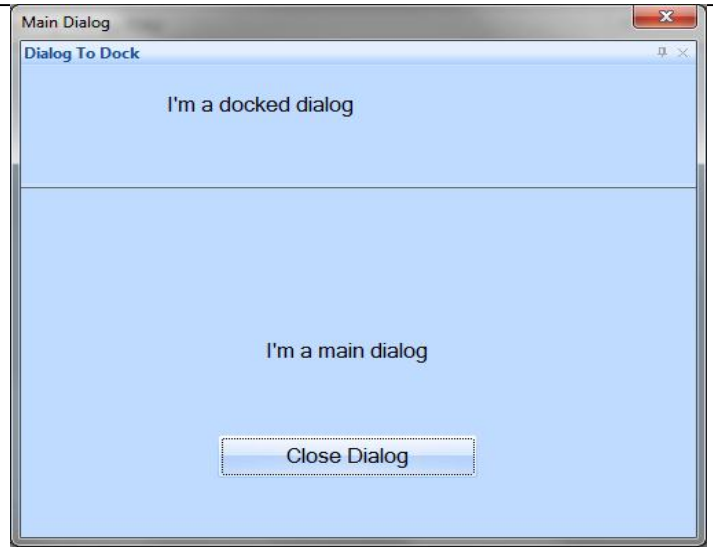
Attributes die gesetzt werden müssen:
Allow Dock to Parent= Yes
Docking Orientation = select option
Type of Dialog = Modeless



Andocken von Dialogen in Dialoge

- **Ausführung:**

```
◇ Call SalDlgSetDockStatus( SalCreateWindow( dlgToDock, dlgMainDialog ), DOCK_Top )
```



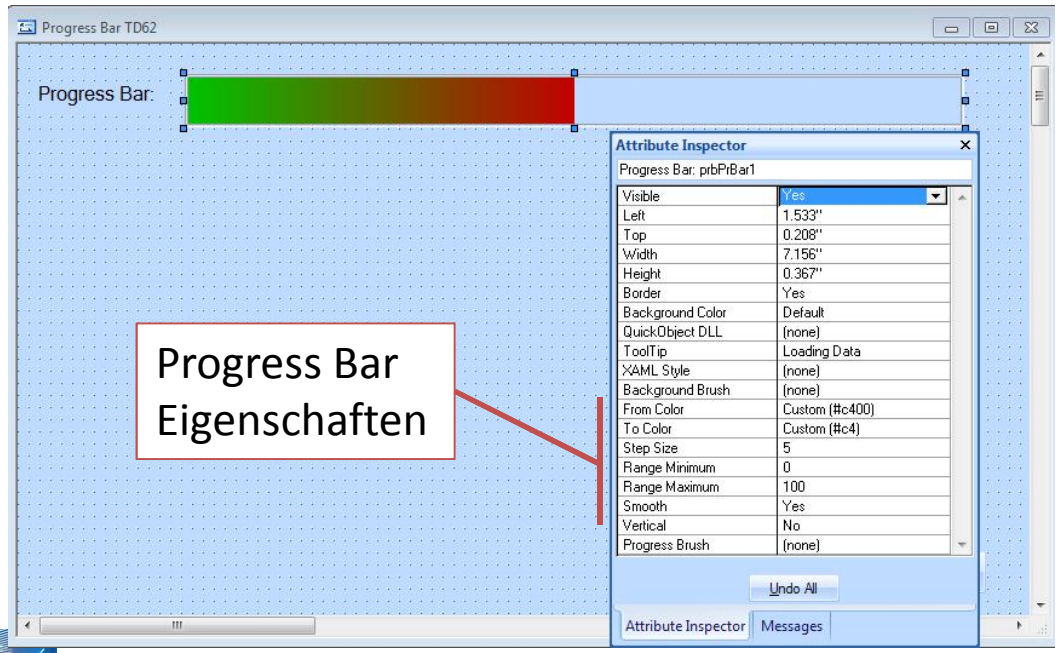
Demo

Neue *Progress Bar*

- **Ersetzt das *cMeter* Control**
  - **Attribute Inspector**
  - **Laufzeit API**
  - ***Progress Bar* Klasse**

Neue *Progress Bar*

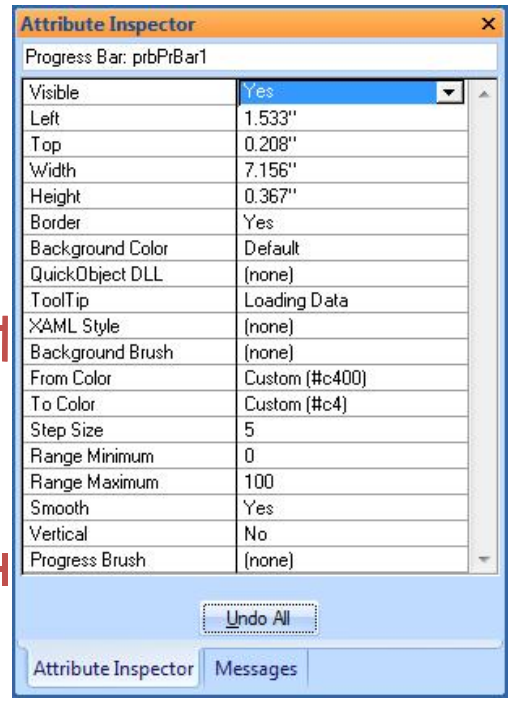
- **Attribute**



Neue *Progress Bar*

- **WPF .NET Attribute**

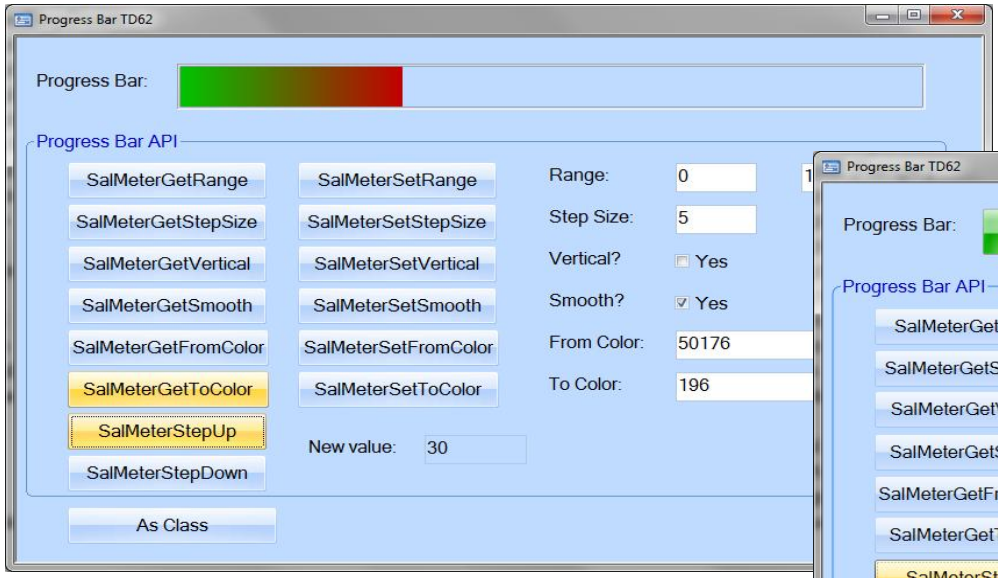
Support von *Styles* und *Brushes* für WPF .NET Anwendungen



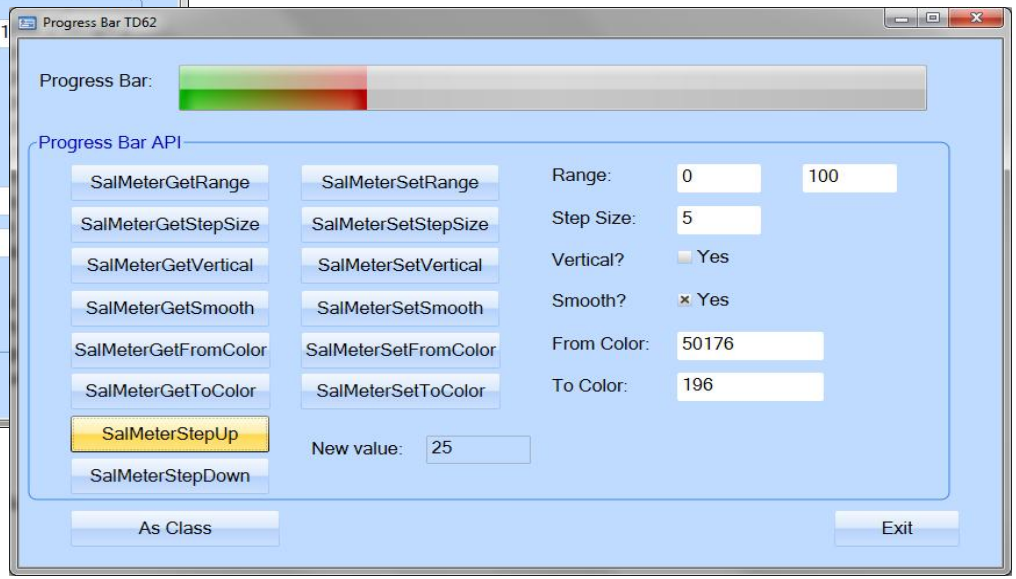
Neue *Progress Bar*

- **Progress Bar API (SalMeter...)**
  - SalMeterSetRange( ) / SalMeterGetRange( )
  - SalMeterSetSmooth( ) / SalMeterGetSmooth( )
  - SalMeterSetStepSize( ) / SalMeterGetStepSize( )
  - SalMeterSetToColor( ) / SalMeterGetToColor( )
  - SalMeterSetFromColor( ) / SalMeterGetFromColor( )
  - SalMeterSetVertical( ) / SalMeterGetVertical( )
  - SalMeterStepUp ( )
  - SalMeterStepDown( )

Neue *Progress Bar*



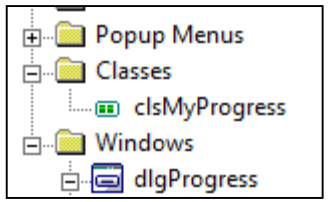
Win32

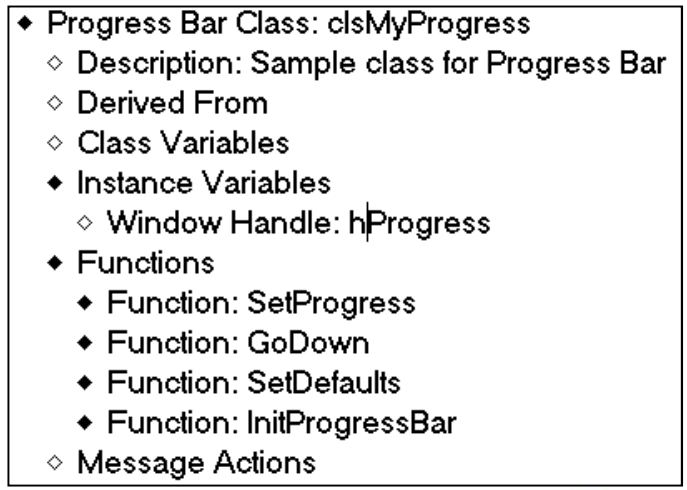


WPF . NET

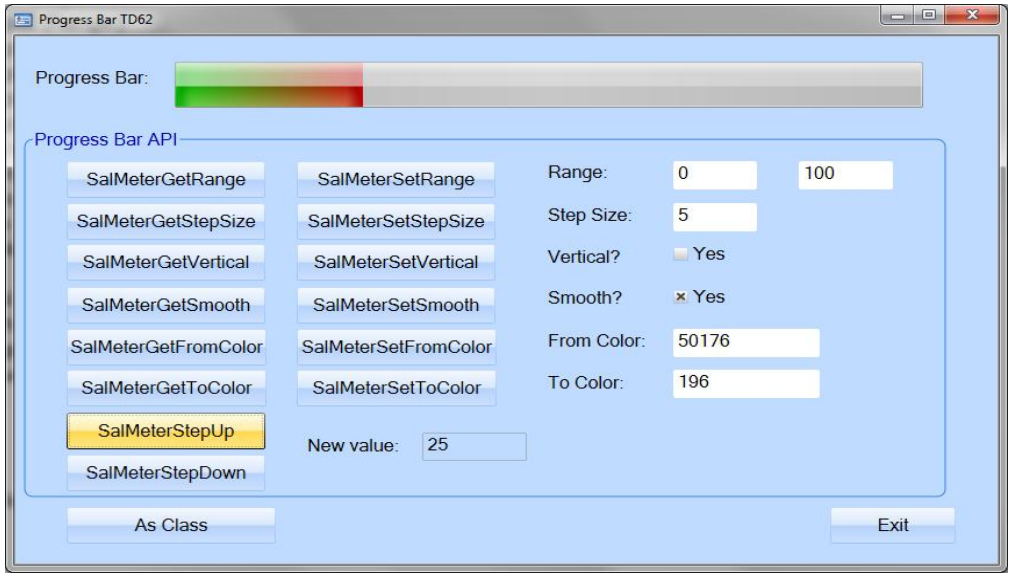
Neue *Progress Bar*

- **Eine Progress Bar kann auch als Klasse definiert werden:**





Neue *Progress Bar*



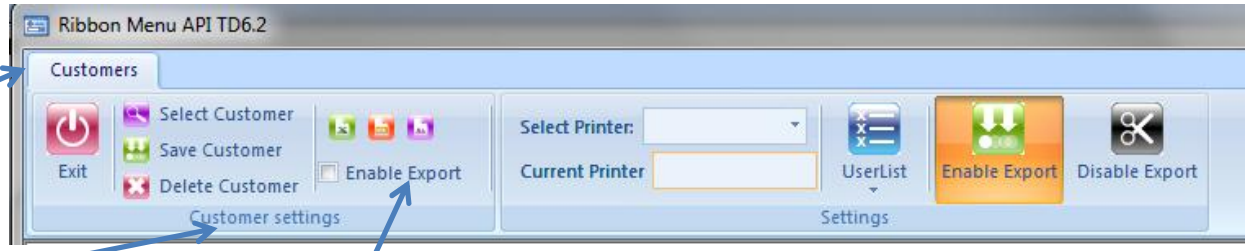
Demo

Das neue *Ribbon Bar* Control

- **Ein Überblick…**
  - **Objekte**
  - **Design**
  - **API**

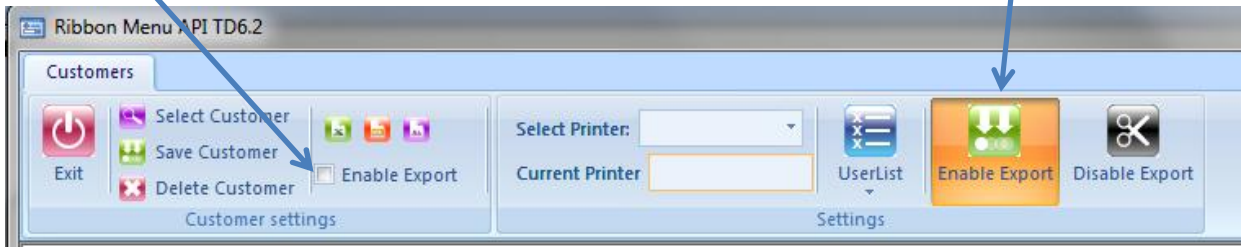
Ribbon Bar Control

- **Objekte:**
  - *Tab*
    - *Group*
      - *Ribbon Items*
        - *Button, DropDown, Tray, Combo, Check Button, Radio Button, Data Field, Seperator*



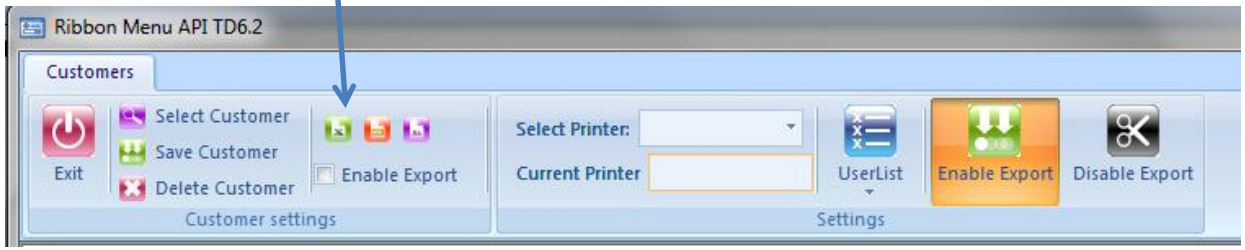
Ribbon Bar Control

- Alle *Ribbon Items* können „groß“ oder „klein“ sein



Ribbon Bar Control

- **Trays sind Gruppen von Items. Sie werden in 1/3 der Größe angezeigt.**



Ribbon Bar Control

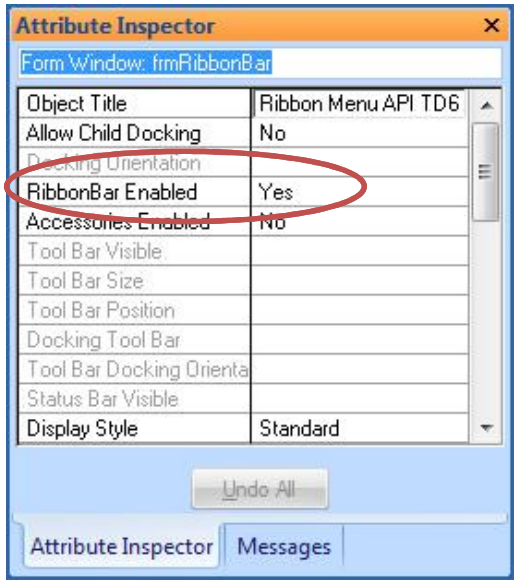
- **Automatisches Layout**
  1. **Von „oben nach unten“**
  2. **Von „links nach rechts“**

Ribbon Bar & Compiler

- ***Ribbon Bar Items*** **können nicht direkt referenziert werden!**
  - **Geht nicht:** ***Set sPrinter = rdfPrinter*** **!!!**
- ***Item*** **Namen müssen eindeutig sein!**
- **Ribbon Bars können nicht zusammengeführt werden!**

Erzeugen einer *Ribbon Bar*

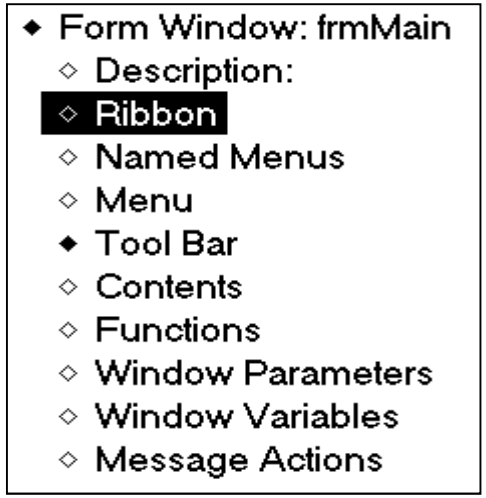
- **Jede Form bzw. MDI kann EINE *Ribbon Bar* haben:**



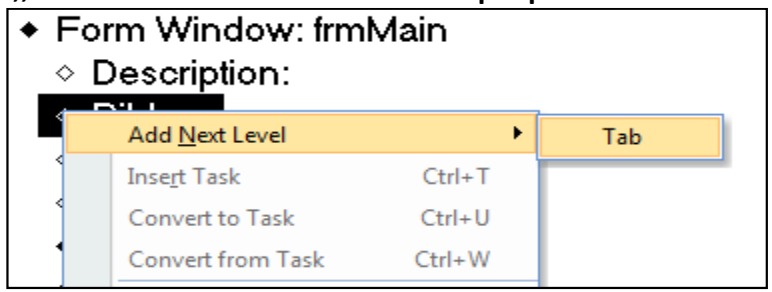
Set Attribute RibbonBar Enabled = Yes

Erzeugen einer *Ribbon Bar*

- **Definieren eines neuen *Tabs***

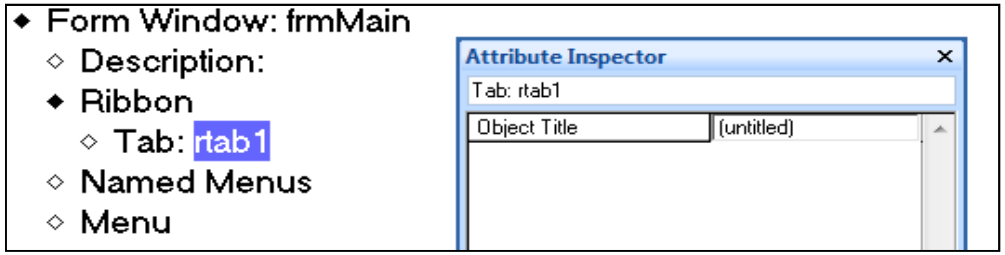


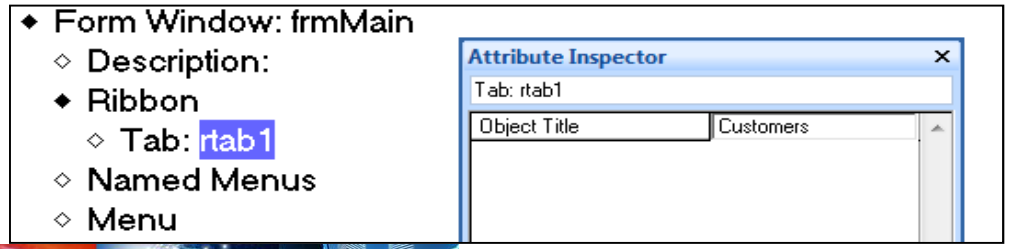
„Rechts-Click“ öffnet Popup Menu:



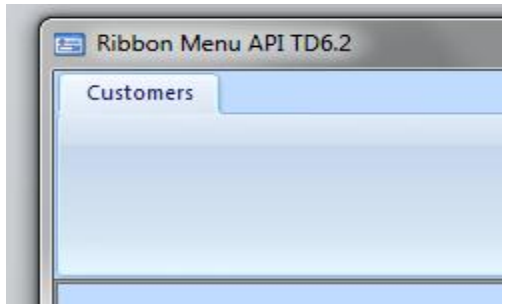
Erzeugen einer *Ribbon Bar*

- **Definieren eines neuen *Tabs***



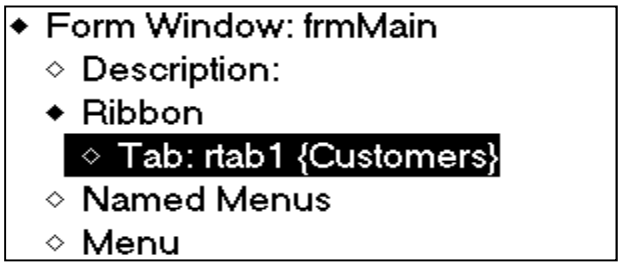


Ergebnis:

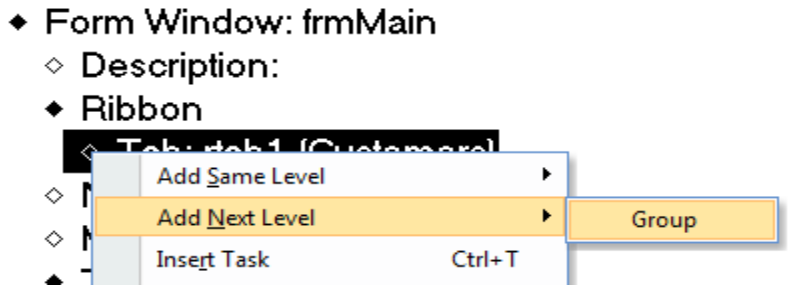


Erzeugen einer *Ribbon Bar*

- **Definieren einer neuen *Group***



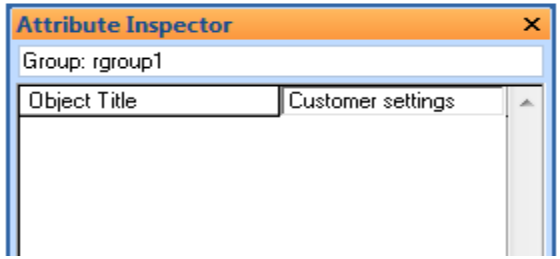
„Rechts-Click öffnet Popup Menu:



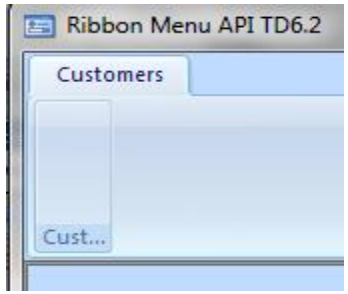
Erzeugen einer *Ribbon Bar*

- **Definieren einer neuen *Group***

- Form Window: frmMain
  - Description:
  - Ribbon
    - Tab: rtab1 {Customers}
      - Group: rgroup1
  - Named Menus
  - Menu

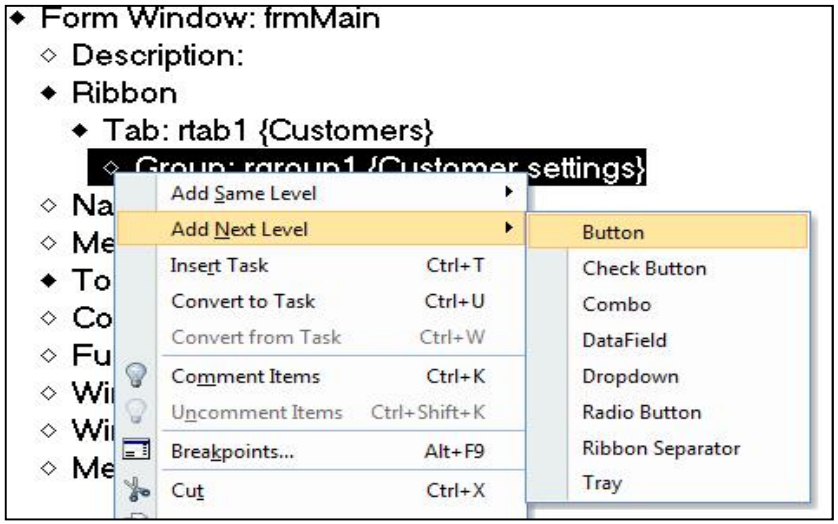


Ergebnis:



Erzeugen einer *Ribbon Bar*

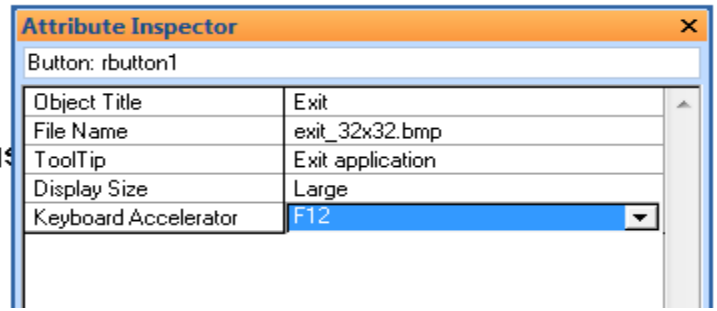
- **Definieren eines neuen *Items***



Erzeugen einer *Ribbon Bar*

- **Definieren eines grossen *Button***

- Form Window: frmMain
  - Description:
  - Ribbon
    - Tab: rtab1 {Customers}
      - Group: rgroup1 {Customer settings
        - Button: rbutton1
          - Enabled when:
          - Ribbon Actions

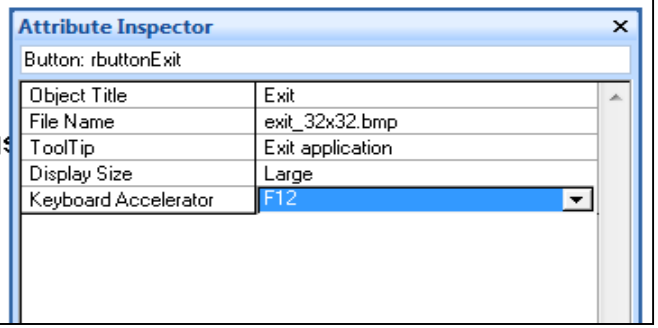


Erzeugen einer *Ribbon Bar*

- **Definieren eines großen *Button***

Define ‚Enable When' and ‚Ribbon Action' code:

```
◆ Form Window: frmMain
  ◇ Description:
  ◆ Ribbon
    ◆ Tab: rtab1 {Customers}
      ◆ Group: rgroup1 {Customer settings
        ◆ Button: rbuttonExit {Exit}
          ◇ Enabled when: bStart
          ◆ Ribbon Actions
            ◇ Call SalQuit( )
```



Ergebnis:

Erzeugen einer *Ribbon Bar*

- **Outline Struktur einer *Ribbon Bar***

```
◆ Ribbon
  ◆ Tab: rtab1 {Customers}
    ◆ Group: rgroup1 {Customer settings}
      ◆ Button: rbuttonExit {Exit}
      ◇ Ribbon Separator
      ◆ Button: rbuttonSelectCust {Select Customer}
      ◆ Button: rbuttonSaveCust {Save Customer}
      ◆ Button: rbuttonDeleteCust {Delete Customer}
      ◇ Ribbon Separator
      ◆ Tray: rtray1
        ◆ Button: rbuttonExcel {Export Excel}
        ◆ Button: rbuttonXML {Export XML}
        ◆ Button: rbuttonText {Export Text}
      ◆ Check Button: checkExport {Enable Export}
      ◇ Ribbon Separator
    ◆ Group: rgroup2 {Settings}
```

Ribbon Bar API

- **Ribbon Bar API (SalRibbon...):**

  SalRibbonGetItemText() / SalRibbonSetItemText( )

  SalRibbonGetItemChecked() / SalRibbonSetItemChecked()

  SalRibbonGetItemCaption() / SalRibbonSetItemCaption()

  SalRibbonGetItemEnabled() / SalRibbonSetItemEnabled()

  SalRibbonClearList()

  SalRibbonAddListValue()

  SalRibbonGetState()

  SalRibbonMaximize() / SalRibbonMinimize()

Ribbon Bar API (Beispiel)

- **SalRibbonGetItemText()**

  bOk = SalRibbonGetItemText(hWnd, itemName, strValue)

  Gets the current value of a *Datafield* or *Combo* ribbon item

  Parameters:

  hWnd : Window Handle. Form window or MDI window containing Ribbon Bar.

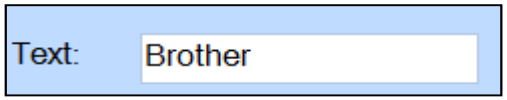
  itemName : String. Ribbon Bar Item name.

  strValue : Receive String. Value of ribbon menu item.

  Return Value:

  bOk is TRUE if the function succeeds and FALSE if it fails


```
◆ On SAM_Click
  ◇ Call SalRibbonGetItemText( hWndForm, 'rdfPrinter', dfText )
```



Ribbon Bar API (Beispiel)

- **SalRibbonAddListValue()**

  bOk = SalRibbonAddListValue(hWnd, itemName, strNewValue)

  Adds a value to the end of a Combo ribbon item (or elsewhere is Sorted=true)

  Parameters:

  hWnd : Window Handle. Form window or MDI window containing Ribbon Bar.

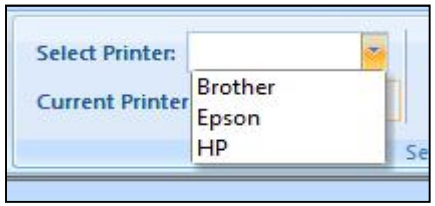
  itemName : String. Ribbon Bar Item name.

  strNewValue : String. Value to be added.

  Return Value:

  bOk is TRUE if the function succeeds and FALSE if it fails

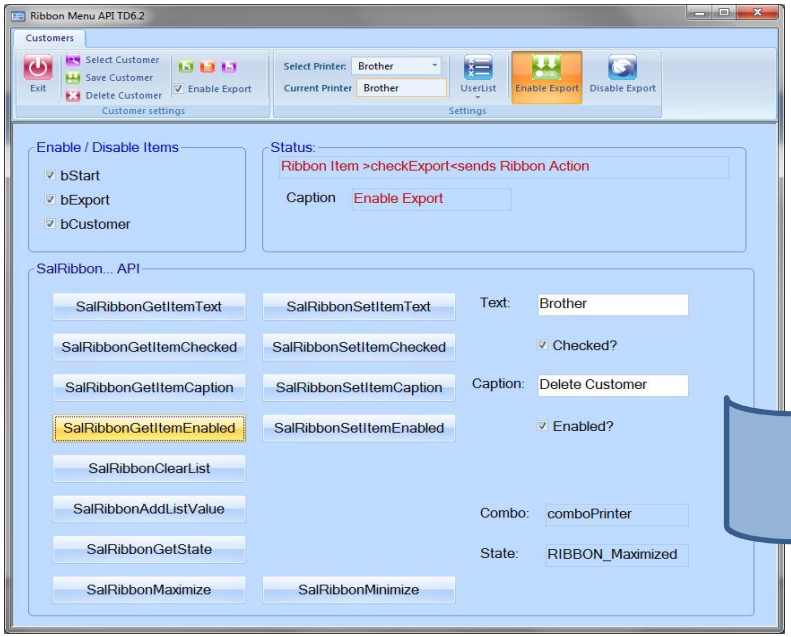
```
◇ Call SalRibbonAddListValue( hWndForm, 'comboPrinter', 'Epson' )
◇ Call SalRibbonAddListValue( hWndForm, 'comboPrinter', 'HP' )
◇ Call SalRibbonAddListValue( hWndForm, 'comboPrinter', 'Brother' )
```



Ribbon Bar in TD 6.2

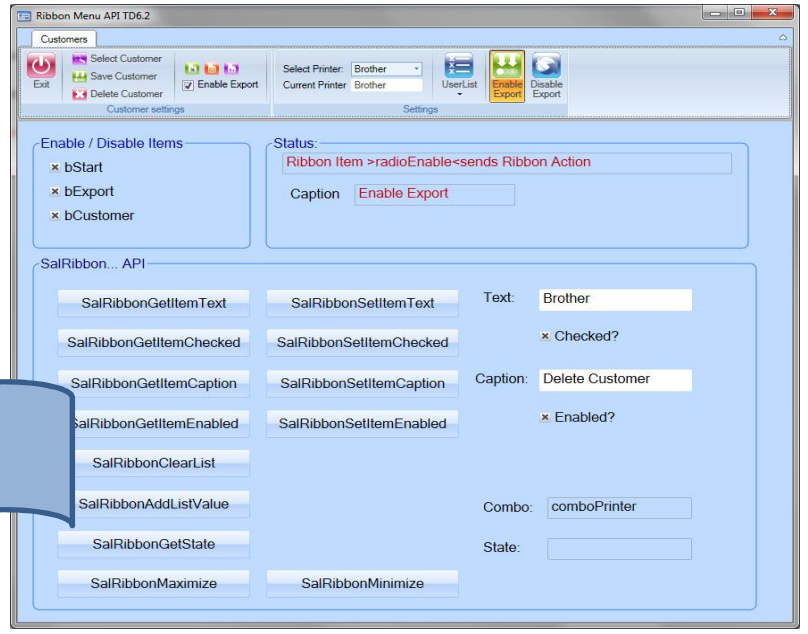
- Kein globaler Aspekt
- Keine Ribbon Bar Klasse
- Kein „mergen“ von Ribbon Bars
- Ein dynamisches erzeugen von *Ribbon Items*
- Keine „externe“ API (z.B Call SalHideWindow(rbtExit))

Ribbon Bar Beispiel



Win32

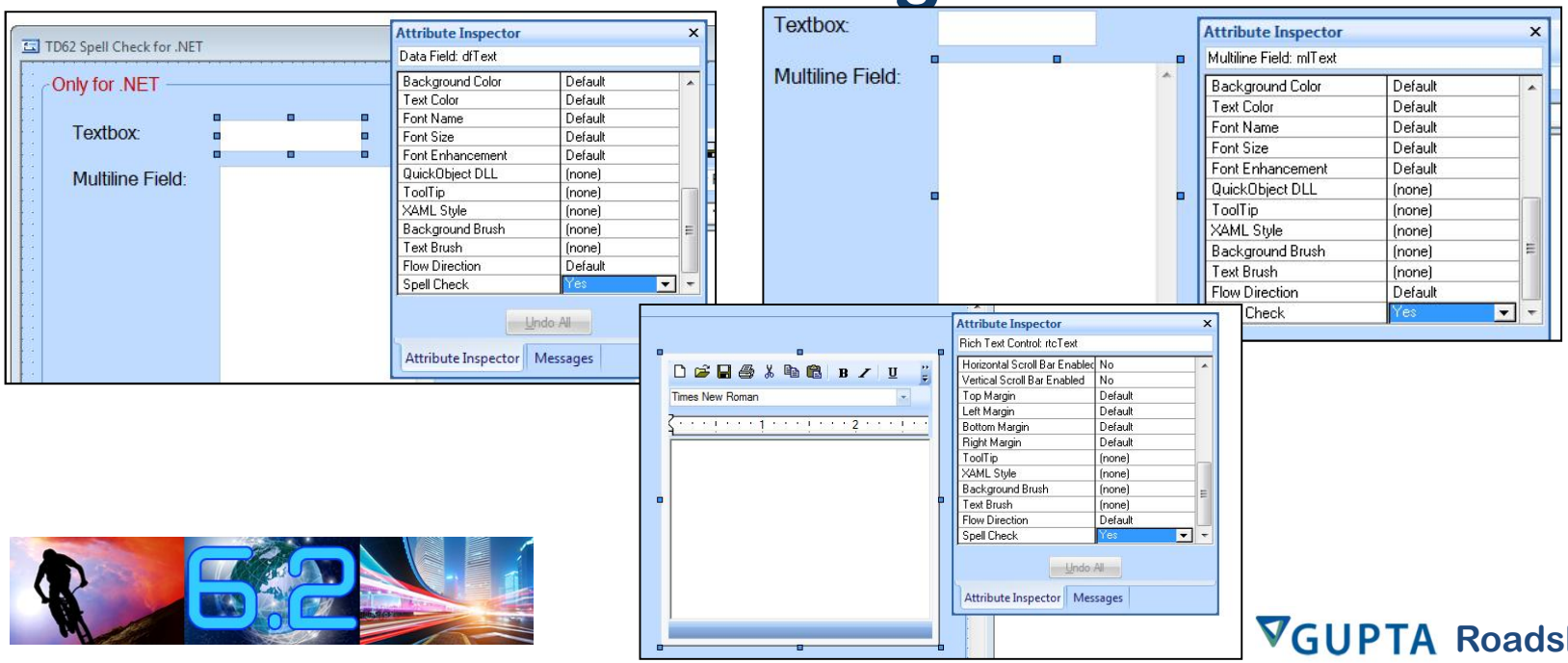
Demo



WPF

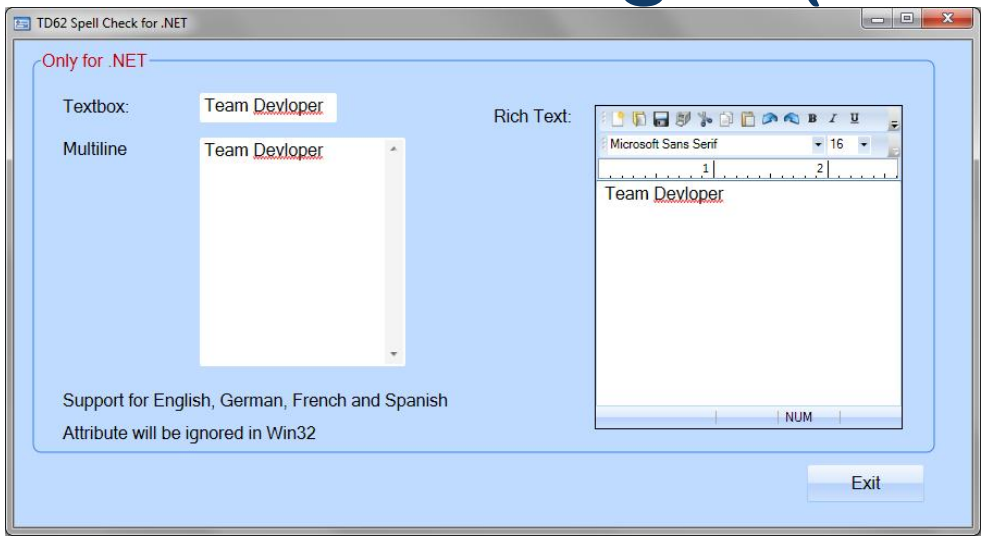
Rechtschreibprüfung

- **Für .NET Anwendungen**



Rechtschreibprüfung

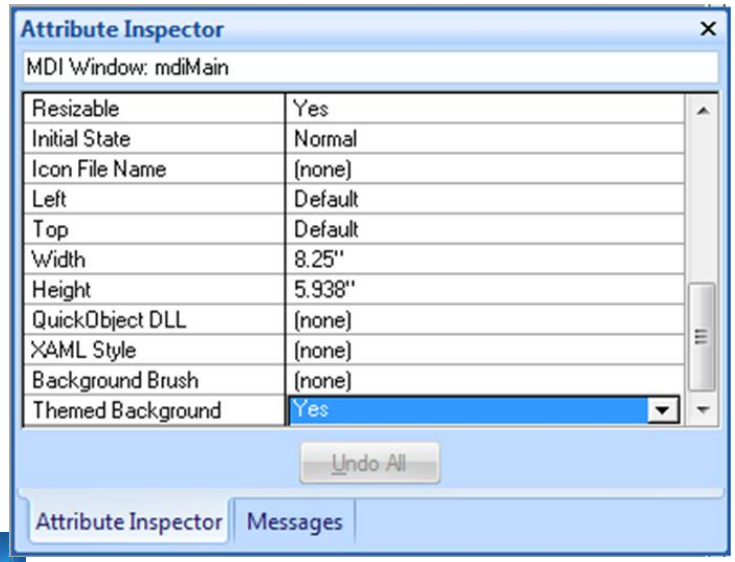
- **Für .NET Anwendungen (Beispiel)**



Demo

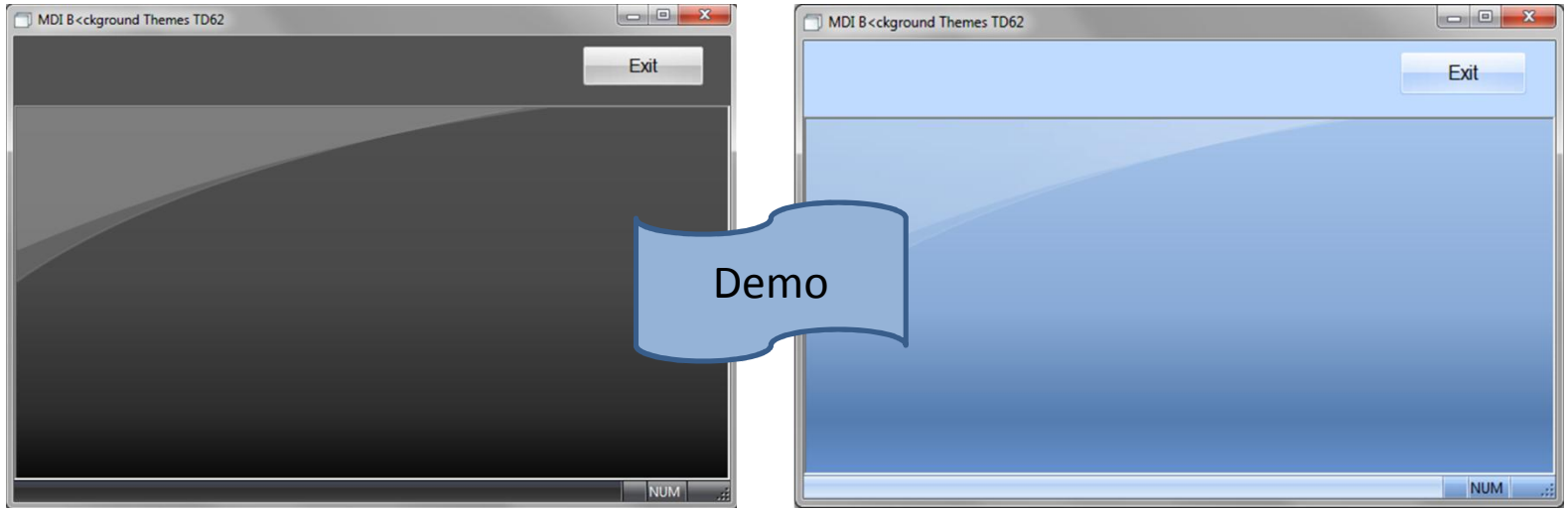
MDI Hintergrund Schema

- **Neu im Attribute Inspector**



MDI Hintergrund Schema

- **Neu im Attribute Inspector**

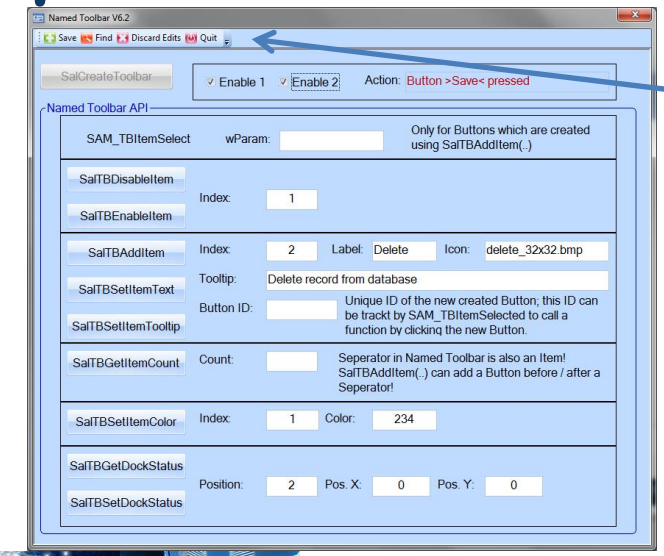


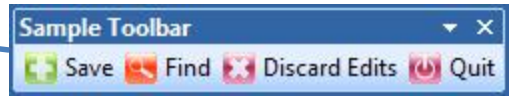
Named Toolbar Erweiterungen

- ***Named Toolbars*** **können in** ***Forms*** **oder** ***MDI Windows*** **verwendet werden**
- ***Forms*** **oder** ***MDI Windows*** **können mehrere** ***Named Toolbars*** **haben**
- ***Named Toolbars*** **können gedockt oder an beliebiger X/Y Position dargestellt werden**
- ***Named Toobars*** **können in mehreren Forms gleichzeitig genutzt werden**

Named Toolbar Erweiterungen

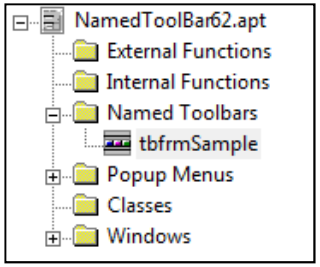
- Beispiel einer *Named Toolbar*





Named Toolbar Erweiterungen

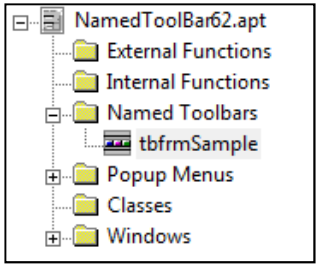
- **Beispiel einer *Named Toolbar***




```
◆ Toolbar: tbfrmSample
  ◇ Description: sample Named Toolbar for testing API
  ◇ Title: Sample Toolbar
  ◆ Toolbar Button: Save
    ◇ Status Text: Save Record
    ◇ Enabled when: bEnable1
    ◇ Picture File Name: add_32x32.bmp
    ◆ Actions
      ◇ Call frmSample.TrackAction('Button >Save< pressed' )
  ◆ Toolbar Button: Find
  ◆ Toolbar Button: Discard Edits
  ◇ Menu Separator
  ◆ Toolbar Button: Quit
```

Named Toolbar Erweiterungen

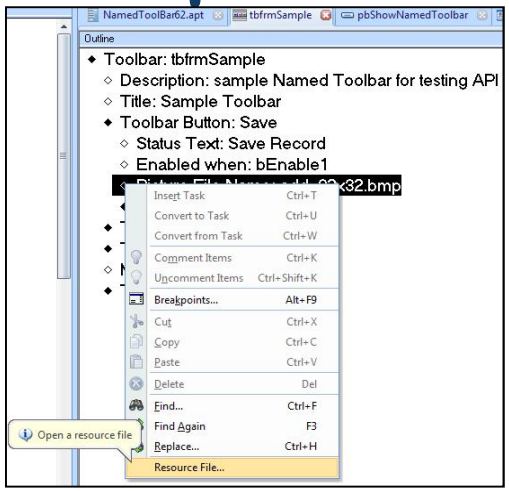
- **Beispiel einer *Named Toolbar***

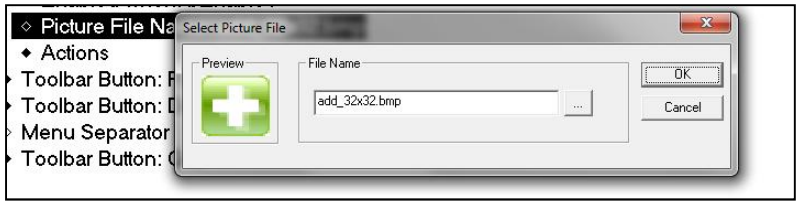



```
◆ Toolbar: tbfrmSample
  ◇ Description: sample Named Toolbar for testing API
  ◇ Title: Sample Toolbar
  ◆ Toolbar Button: Save
    ◇ Status Text: Save Record
    ◇ Enabled when: bEnable1
    ◇ Picture File Name: add_32x32.bmp
    ◆ Actions
      ◇ Call frmSample.TrackAction('Button >Save< pressed' )
  ◆ Toolbar Button: Find
  ◆ Toolbar Button: Discard Edits
  ◇ Menu Separator
  ◆ Toolbar Button: Quit
```

Named Toolbar Erweiterungen

- Beispiel einer *Named Toolbar*





Named Toolbar Erweiterungen

- **Named Toolbar API (SalTB...):**
  - **SalCreateToolbar( )**
  - **SalTBDisableItem( ) / SalTBEnableItem( )**
  - **SalTBAddItem( )**
  - **SalTBSetItemText( )**
  - **SalTBSetItemTooltip( )**
  - **SalTBGetItemCount( )**
  - **SalTBSetItemColor( )**
  - **SalTBGetDockStatus( ) / SalTBSetDockStatus( )**

Named Toolbar Erweiterungen

- Folgendes Ereignis kann in einem Form oder MDI Window ausgewertet werden:
  - ***SAM_TBItemSelected***

Dieses Ereignis wird nur dann ausgelöst, wenn ein dynamisch erzeugter *Toolbar Button* geklickt wird

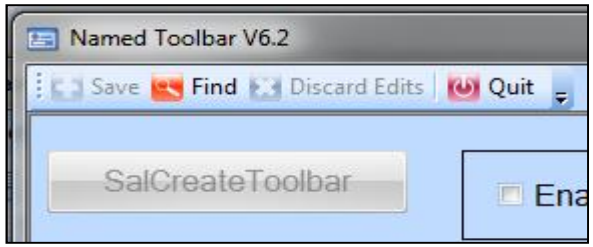
Named Toolbar Erweiterungen

- Folgendes Ereignis kann in einem Form oder MDI Window ausgewertet werden:
  - ***SAM_TBItemSelected***

Dieses Ereignis wird nur dann ausgelöst, wenn ein dynamisch erzeugter *Toolbar Button* geklickt wird

Named Toolbar Erweiterung

```
! Assign Named Toolbar to form
Set hToolbar =  SalCreateToolbar( tbfrmSample, hWndForm, DOCK_Top, 0, 0 )
!
```



Achtung

Das Ergebnis der Funktion ist der eindeutige Handle dieser Named Toolbar. Es sollte für alle SalTB Funktionen, die einen Handle als Parameter haben, eingesetzt werden.

Named Toolbar Erweiterung

- **SalTBAddItem( )**

nResult = SalTBAddItem(hWndTB, nIndex, sCaption, sFileName, sTooltip)

Dynamisches erzeugen eines *Toolbar Buttons*

- **Parameter**

hWndTB :	Window Handle. Der *Window Handle* der *Named Toolbar*
nIndex :	Number. 0-basierender Index (Position innerhalb der *Named Toolobar*)
sCaption :	String. Text des neuen *Buttons*
sFileName :	String. Name der Bild Datei
sTooltip :	String. *Tooltip* für den neuen *Button*

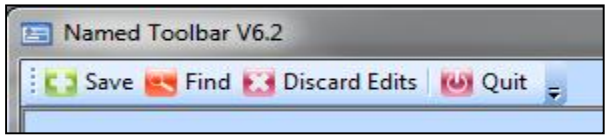
- **Return**

nResult :	Number. Eindeutige ID des neu erzeugten *Buttons* (-1 = Fehler)

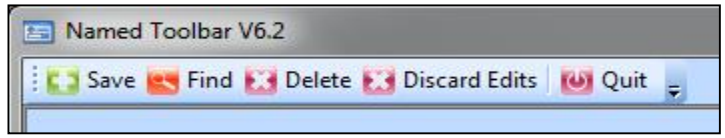
Named Toolbar Erweiterung

nResult = SalTBAddItem(hWndTB, nIndex, sCaption, sFileName, sTooltip)

- Pushbutton: pbNewButton
  - Message Actions
    - On SAM_Click
      - Set nNewID = SalTBAddItem(hToolbar, dfIndex, dfLabel, dfIcon, dfTooltip)



Vorher



Nachher...

Named Toolbar Erweiterungen

SAM_TBItemSelect **wird gesendet, wenn ein dynamisch generierter Button, der mit SalTBAddItem( ) erzeugt wurde, geklickt wurde:**

- **Message Variables**

  hWndForm: Window Handle des Parent Window.

  hWndItem : Window Handle der Named Toolbar.

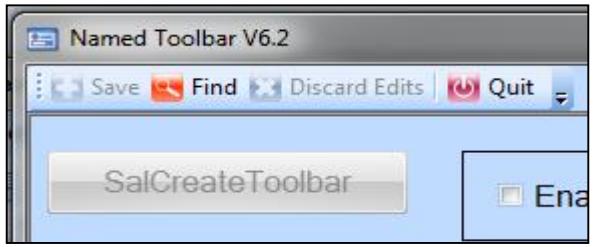
  wParam: Number. Unique ID des neuen Buttons.

  lParam: Not used.

```
On SAM_Click
    Set nNewID =  SalTBAddItem( hToolbar, dfIndex, dfLabel, dfIcon, dfTooltip )
```

```
On SAM_TBItemSelect
    Set dfwParam = wParam
    If wParam = nNewID
        Call ButtonFunktion( wParam )
```

Named Toolbar Erweiterungen



Demo

Grid Erweiterungen

- **Neue und erweiterte API für das Grid**
  - **SalGridDataExport( ) (erweitert)**
  - **SalGridWriteState( )**
  - **SalGridApplyState( )**
  - **SalTblSetCellBackColor( )**

Grid Erweiterungen

bOk = SalGridDataExport (hWndGrd, sDateiName, nFormat)

Wenn *sDateiName* leer ist, dann wird ein “Speichern als” – Dialog geöffnet.

Dateiformate in nFormat:

- Format_TEXT (.txt file)
- Format_XML (.xml file)
- Format_EXCEL (.xls file)
- Format_CSV (.csv file)

Grid Erweiterungen

bOk = SalGridWriteState(hWndGrid, sXMLFile)

Speichert das aktuelle Layout des *Grid* in einen XML File.

Parameter:

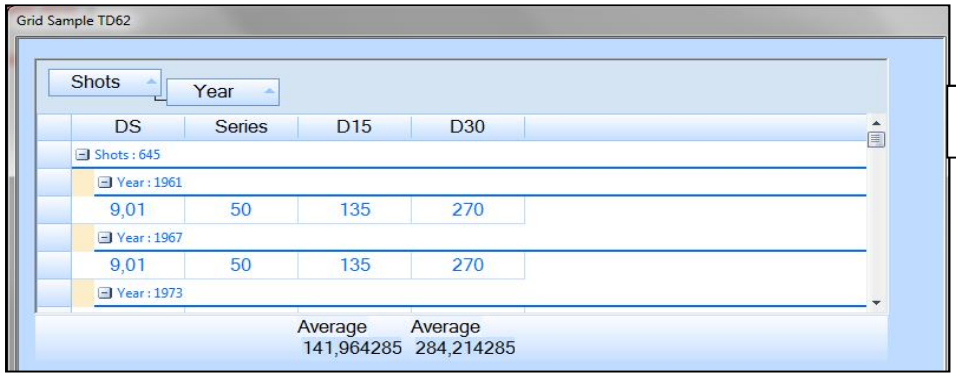
hWndGrid : Window Handle. Das Handle (oder Name) des Grids.

sXMLFile: String. Name des XML Files um das Schema zu speichern.

Hinweis: Folgende Informationen & Attribute können gespeichert werden:

Group By
Order By
Group by area on/off
Summary on/off, which columns, which statistics
Hidden and visible columns
Row Height

Grid Erweiterungen




```
On SAM_Click
    Set bOk =  SalGridWriteState( grid1, 'GridState.xml' )
```

GridState.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<TeamDeveloperGrid62 Version="1">

  <TDGrid IsGroupArea="true" IsSummaryEnabled="true" RowHeight="26">
    <GridColumn ColID="32768" IsGrouped="true" IsSortEnabled="false" SortDir="asc" SortIdx="1" SummaryType="0" VisibleIdx="5">Year</GridColumn>
    <GridColumn ColID="32770" IsGrouped="true" IsSortEnabled="false" SortDir="asc" SortIdx="0" SummaryType="3" VisibleIdx="4">Shots</GridColumn>
    <GridColumn ColID="32769" IsGrouped="false" IsSortEnabled="true" IsVisible="true" SummaryType="0" VisibleIdx="0">DS</GridColumn>
    <GridColumn ColID="32771" IsGrouped="false" IsSortEnabled="true" IsVisible="true" SummaryType="0" VisibleIdx="1">Series</GridColumn>
    <GridColumn ColID="32772" IsGrouped="false" IsSortEnabled="true" IsVisible="true" SummaryType="4" VisibleIdx="2">D15</GridColumn>
    <GridColumn ColID="32773" IsGrouped="false" IsSortEnabled="true" IsVisible="true" SummaryType="4" VisibleIdx="3">D30</GridColumn>
  </TDGrid>

</TeamDeveloperGrid62>
```

Grid Erweiterungen

bOk = SalGridApplyState(hWndGrid, sXMLFile)

Rekonstruiert ein Grid Layout basierend auf einen XML File

Parameter:

hWndGrid : Window Handle. Name oder Handle des Grids.

sXMLFile: String. Der XML file mit den Layout Informationen

Return Value:

Hinweis: Folgende Informationen können aus dem Layout File gelesen werden:

- Group By
- Order By
- Group by area on/off
- Summary on/off, which columns, which statistics
- Hidden and visible columns
- Row Height

Grid Erweiterungen

GridState.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<TeamDeveloperGrid62 Version="1">

  <TDGrid IsGroupArea="true" IsSummaryEnabled="true" RowHeight="26">
    <GridColumn ColID="32768" IsGrouped="true" IsSortEnabled="false" SortDir="asc" SortIdx="1" SummaryType="0" VisibleIdx="5">Year</GridColumn>
    <GridColumn ColID="32770" IsGrouped="true" IsSortEnabled="false" SortDir="asc" SortIdx="0" SummaryType="3" VisibleIdx="4">Shots</GridColumn>
    <GridColumn ColID="32769" IsGrouped="false" IsSortEnabled="true" IsVisible="true" SummaryType="0" VisibleIdx="0">DS</GridColumn>
    <GridColumn ColID="32771" IsGrouped="false" IsSortEnabled="true" IsVisible="true" SummaryType="0" VisibleIdx="1">Series</GridColumn>
    <GridColumn ColID="32772" IsGrouped="false" IsSortEnabled="true" IsVisible="true" SummaryType="4" VisibleIdx="2">D15</GridColumn>
    <GridColumn ColID="32773" IsGrouped="false" IsSortEnabled="true" IsVisible="true" SummaryType="4" VisibleIdx="3">D30</GridColumn>
  </TDGrid>

</TeamDeveloperGrid62>
```

- On SAM_Click
  - Set bOk = SalGridApplyState(grid1, 'GridState.xml')

Grid Sample TD62

Shots | Year

DS	Series	D15	D30
Shots : 645			
Year : 1961			
9,01	50	135	270
Year : 1967			
9,01	50	135	270
Year : 1973			
		Average 141,964285	Average 284,214285

Grid Erweiterungen

bOk = SalTblSetCellBackColor(hWndGrid, nColor, bSet)

Parameter:

hWndGrid : Window Handle. Das Handle (oser Name) des Grid.

nColor: Vordefinierte Farbe. BeispielCOLOR_LightAqua.

bSet: Boolean. TRUE oder FALSE.

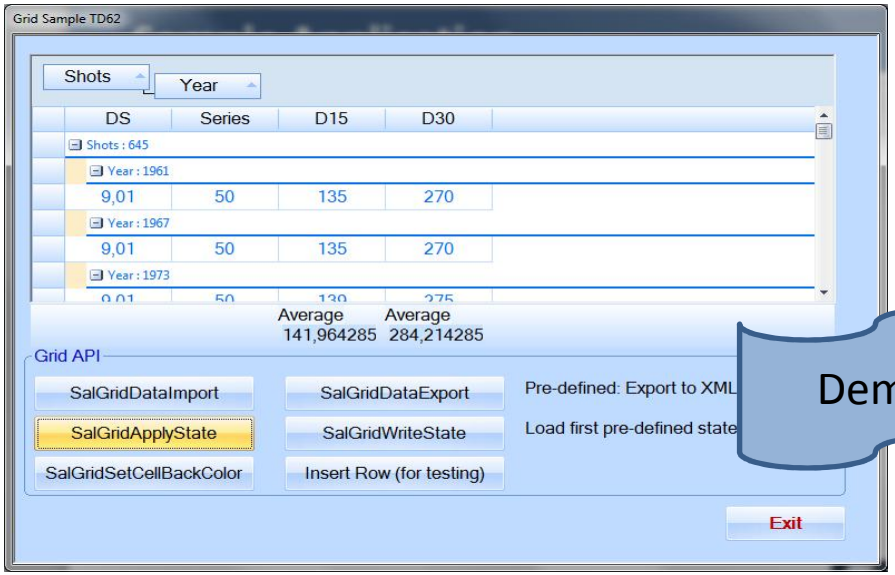
Return Value:

bOk ist *TRUE* wenn die Funktion erfolgreich war

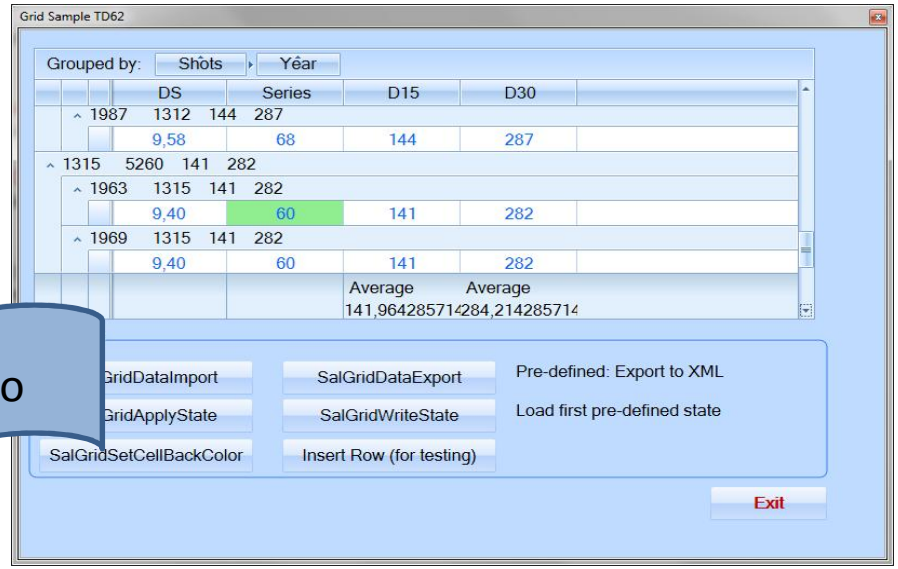
```
◆ On SAM_Click
  ◇ Call SalTblSetContext( grid1, 2 )
  ◇ Call SalTblSetCellBackColor( grid1.colSeries, COLOR_LightGreen, TRUE )
```

s	Series	D15
	50	135
	45	139
5	60	141
2	68	144
5	65	145

Grid Erweiterungen



Demo



Win32

.NET WPF

Tree Erweiterungen

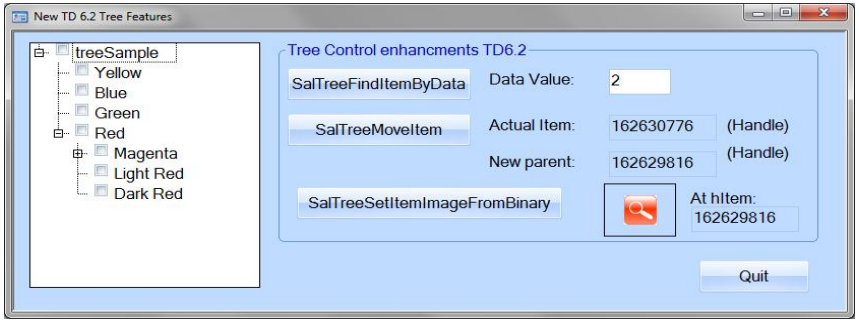
- **Neue API Funktionen**
  - **SalTreeMoveItem( )**
  - **SalTreeFindItemByData( )**
  - **SalTreeSetItemImageFromBinary( )**

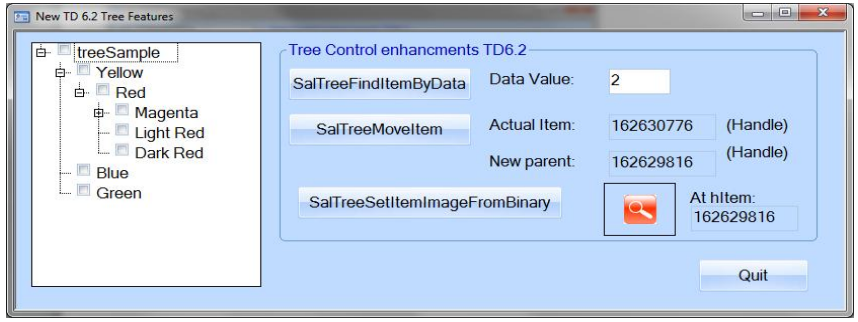
Tree Erweiterungen

bOk = SalTreeMoveItem(hWndTree, hTreeItem, hTreeParentItem, nPositon)

Hängt einen Teilbaum des *Trees* um.

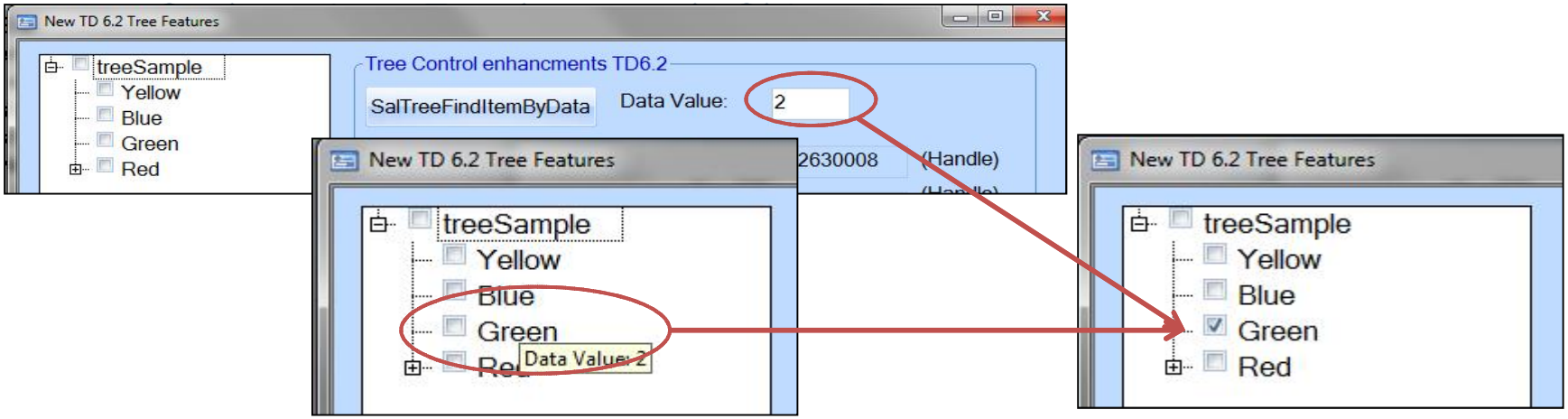
```
◆ On SAM_Click
  ◇ Call SalTreeMoveItem( treeSample, dfActualPos , dfNewPos, 0 )
```





Tree Erweiterungen

hItem = SalTreeFindItemByData(hWndTree, hParent, nData)



- On SAM_Click
  - Set hTreeItem = SalTreeFindItemByData(treeSample, hFirstItem, dfTreeItemValue)

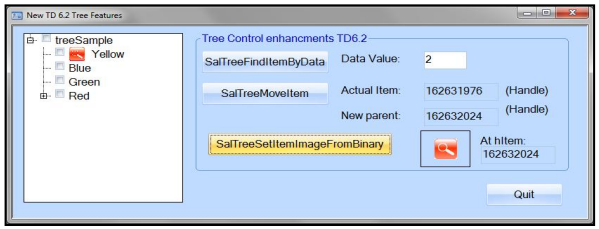
Tree Erweiterungen

bOk = SalTreeSetItemImageFromBinary(hWndTree, hItem, bSelected, nFormat, binPicture)

Diese Funktion ermöglicht es, ein Icon aus einer binären Variablen an ein *Tree Item* zu binden

```
◇ Binary: binPicture
```

```
◆ On SAM_Click
  ◇ Call SalPicGetBinary( picIcon, PIC_FormatBitmap, binPicture )
  ◇ !
  ◇ Call SalTreeSetItemImageFromBinary( treeSample, dfNewPic, FALSE, PIC_FormatBitmap, binPicture )
```

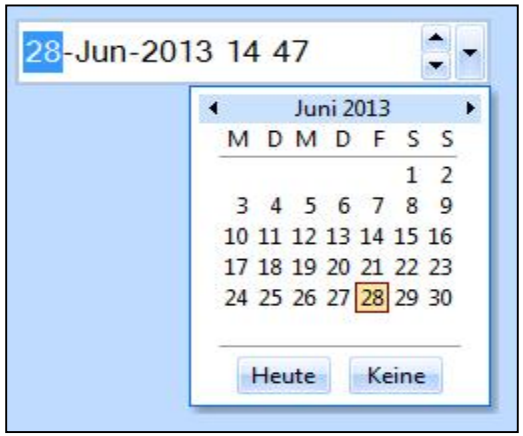


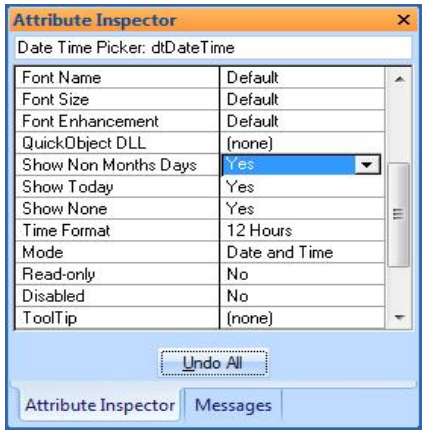
Demo

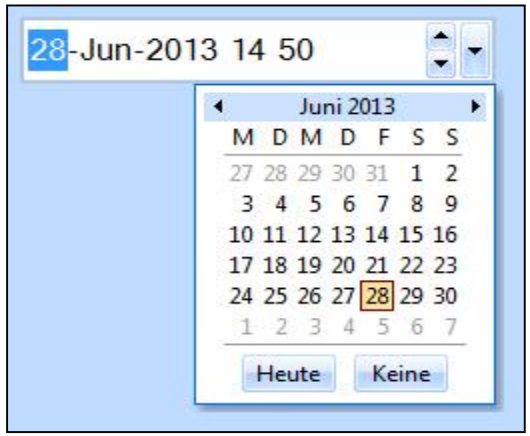
GUPTA Roadshow 2013

DateTime Picker Erweiterungen

- „Show Non **Month Days**“

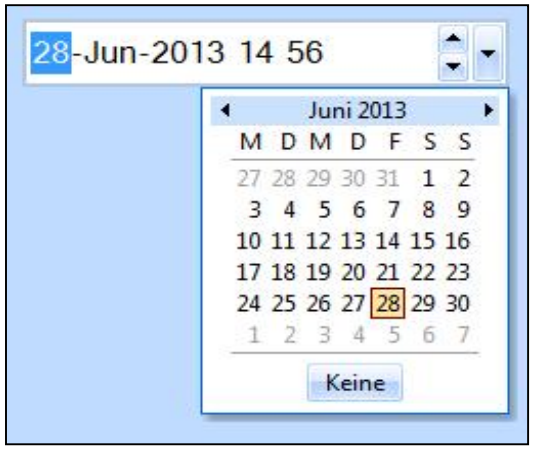


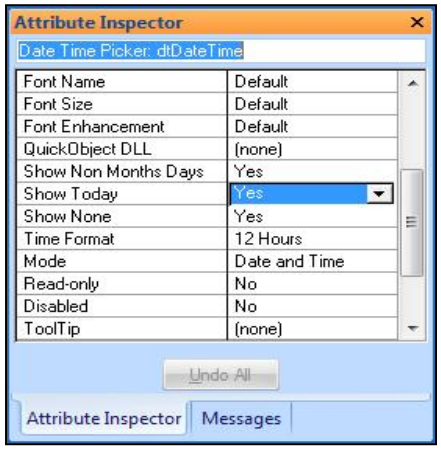


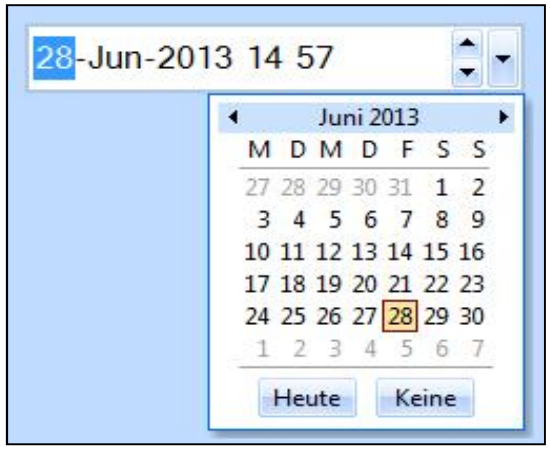


DateTime Picker Erweiterungen

- „Show Today“

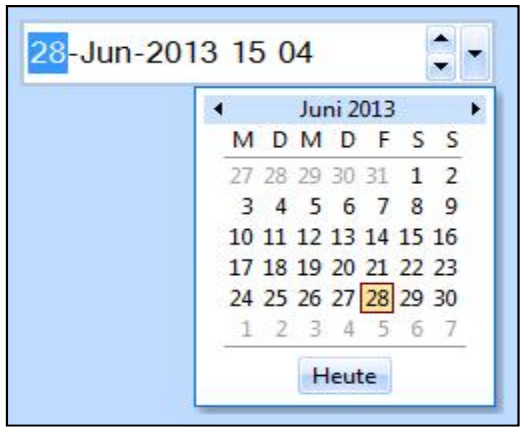


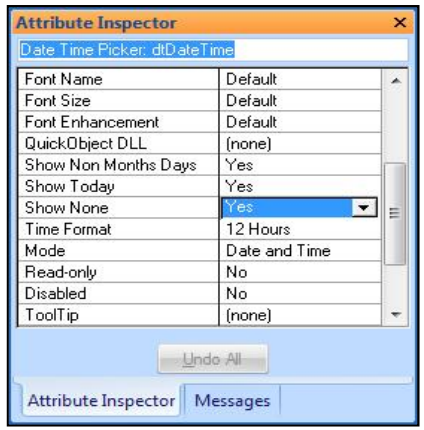


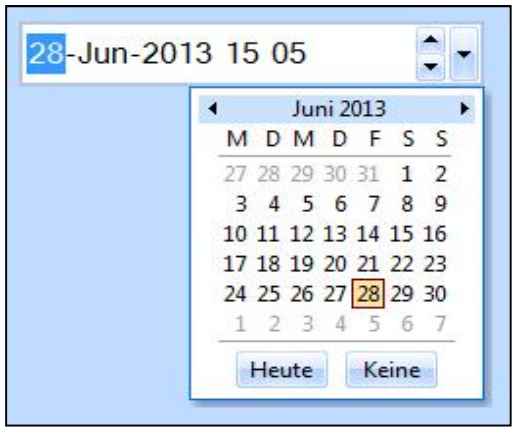


DateTime Picker Erweiterungen

- „Show None“

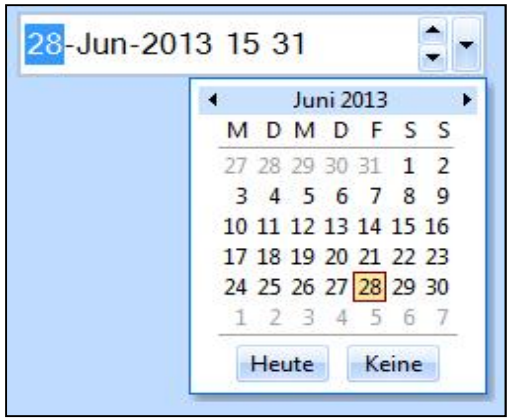


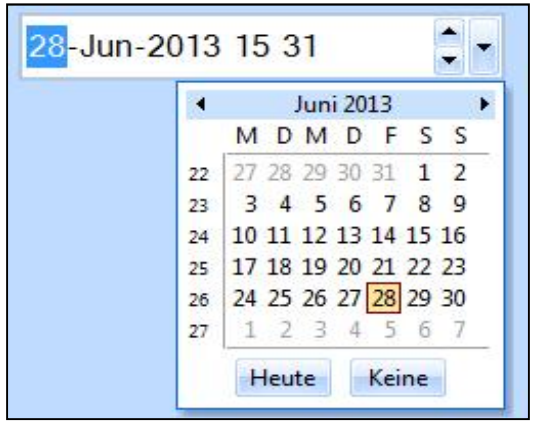




DateTime Picker Erweiterungen

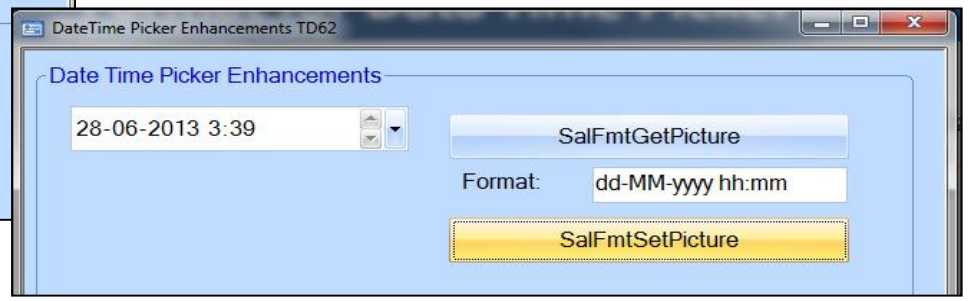
- „Show Week Numbers“





DateTime Picker Erweiterungen

- **Setzen des Formats innerhalb des *DateTime Pickers***
  - Im *Attribute Inspector*
  - Zur Laufzeit mit **SalFmtGetPicture( )** und **SalFmtSetPicture( )**



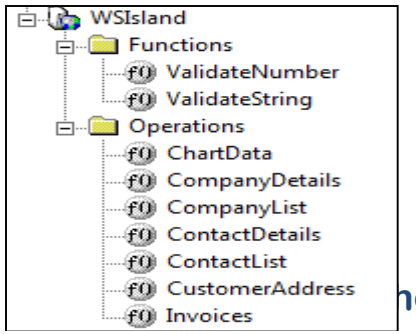
Demo

Team Developer Web Services

- Entwickeln von .NET Web Services
  - .NET Compiler Einstellungen
  - Neue WS-Class: *Operations*
  - Soap Fault / Exception handling
  - Web Service Sicherheit
  - Veröffentlichen von Web Services
  - Testen von Web Services
  - Debuggen von Web Services

Neuer Klassen Typ: Web Service

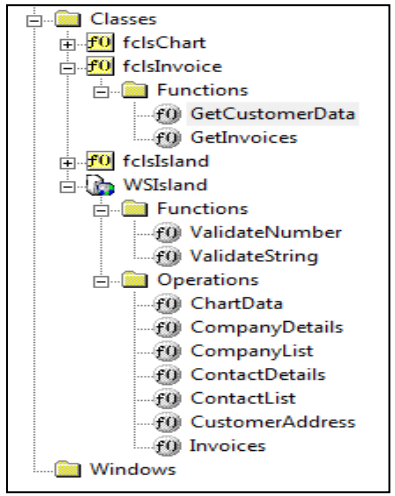
- Web Service Klasse hat
  - Lokale Funktionen
    - Nicht sichtbar für den Anwendungsentwickler
  - Operations
    - Operations sind die Schnittstellen zur Client Anwendung
    - Simple Web Service Struktur:



now 2013

Neuer Klassen Typ: Web Service

- **Web Service Klassen können bestehende *Functional Classes* nutzen:**



Einfache Operation (Receive Parameter)

- **Programm Code einer einfachen Operation:**

```
Operation: CompanyDetails
  Description: Return company details
  Returns
  Parameters
    Number: nInCompanyID
    Receive String: sOutCompany_Name
    Receive String: sOutAddress
    Receive String: sOutCity
    Receive String: sOutState
    Receive String: sOutZIP
    Receive String: sOutCountry
    Receive String: sOutPhone
    Receive String: sOutFax
    Receive String: sOutTerms
    Receive Number: nOutLine
    Receive String: sOutCorporate_URL
  Local variables
    fclsIsland: oCompanyDetails
    Boolean: bOk
```

```
Actions
  ! Check for valid input
  Set bOk = ValidateNumber( nInCompanyID )
  If bOk = TRUE
    ! Set instance variable for select
    Set oCompanyDetails.nClsCompany_ID = nInCompanyID
    ! Fire method
    Call oCompanyDetails.GetCompanyDetails( )
    ! Copy instance variables to receive values
    Set sOutCompany_Name = oCompanyDetails.sClsCompany_Name
    Set sOutAddress = oCompanyDetails.sClsAddress
    Set sOutCity = oCompanyDetails.sClsCity
    Set sOutState = oCompanyDetails.sClsState
    Set sOutZIP = oCompanyDetails.sClsZip
    Set sOutCountry = oCompanyDetails.sClsCountry
    Set sOutPhone = oCompanyDetails.sClsPhone
    Set sOutFax = oCompanyDetails.sClsFax
    Set sOutTerms = oCompanyDetails.sClsTerms
    Set nOutLine = oCompanyDetails.nClsLine
    Set sOutCorporate_URL = oCompanyDetails.sClsCorparate_Url
    Return TRUE
  Else
    Return FALSE
```

Einfache Operation (Arrays von Receive Parameter)

- **Programm Code mit Arrays of Receive Parametern:**

```
Operation: CompanyList
  Description: Return a list of Companies
  Returns
  Parameters
    Receive Number: nOutCompanyIDs[*]
    Receive String: sOutCompanyNames[*]
    Receive String: sOutCompanyCities[*]
    Receive String: sOutCompanyPhones[*]
    Receive Number: nOutResultCount
  Local variables
    fclsIsland: oCompanyList
```

```
Actions
  ! Fire method to fill list; Instance of functional class
  Call oCompanyList.GetCompanyList( )
  ! Map instance varibales to receive variables
  Set nOutCompanyIDs = oCompanyList.nClsCompany_IDs
  Set sOutCompanyNames = oCompanyList.sClsCompany_Names
  Set sOutCompanyCities = oCompanyList.sClsCities
  Set sOutCompanyPhones = oCompanyList.sClsPhones
  Set nOutResultCount = oCompanyList.nClsResultCountCompanyList
  Return TRUE
```

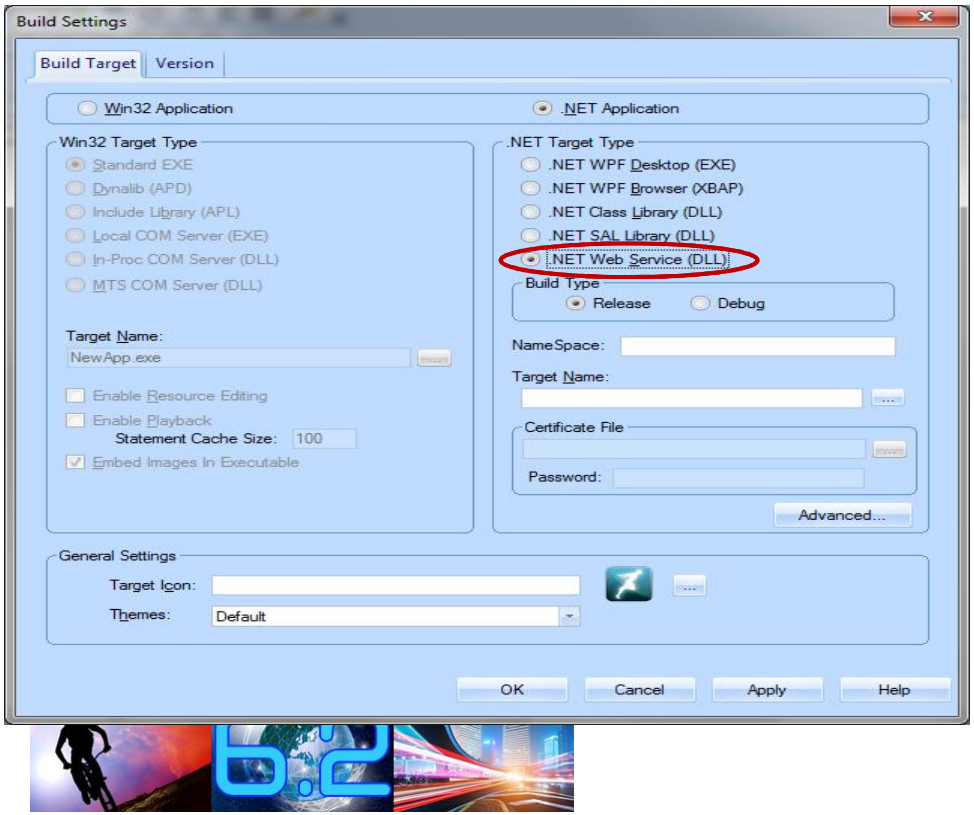
Einfache Operation (Return UDV)

- **Code einer Operation mit UDV als Return Wert:**

```
Operation: CustomerAddress
  Description: Return company details as object
  Returns
    fclsInvoice: oReturn
  Parameters
    Number: nInCompanyID
  Local variables
    fclsInvoice: oLocalCompany
    Boolean: bOk
```

```
Actions
  ! Check for valid input
  Set bOk = ValidateNumber( nInCompanyID )
  If bOk = TRUE
    ! Fire method to fill instance variables
    Call oLocalCompany.GetCustomerData( nInCompanyID )
    ! return whole object
    Return oLocalCompany
  Else
    ! Pass Error as Exception to client
    Call SalThrowSoapFault( 'No valid Company_ID' )
```

.NET Compiler Einstellung



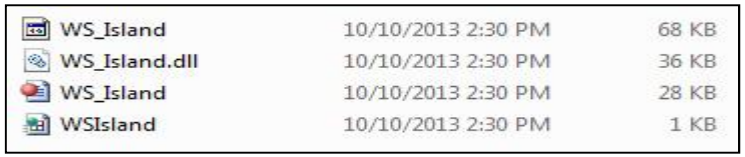
.Net Compiler Einstellung für Web Services

Build Target for .NET Web Service DLL

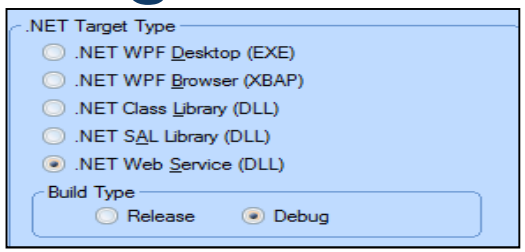
Achtung: Es dürfen KEINE sichtbaren Elemente in einem Web Service „verbaut“ werden (z. B. Message Boxes usw.)
Nutzt ***SalThrowSoapFault( ),*** um Exceptions oder SQL Fehler zum Client zu senden.

.NET Compiler Einstellungen

- **Compiler generierte Files:**



  - **WSIsland.asmx = Web Service Beschreibung für den IIS**
  - **WS_Island.apt = Team Developer Sourcecode**
  - **WS_Island.dll = compiliertes .NET Web Service Assembly**
- **Compiler Einstellung:**



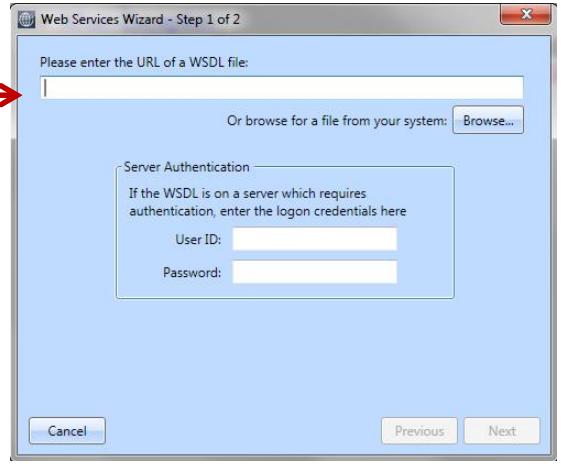
Option: ‚Debug‘: Generiert zusätzlich WS_Island.pdb

Datentypen Serilisierung

- *Binary* Datentypen werden zu *base64 string*
- *Bool* Datentypen werden zu *Boolean*
- *Date/Time* Datentypen werden zu *complex data type: SalDate*
- *Number* Datentypen werden zu complex *data type*: *SalDecimal*
- *String* Datentypen werden zu *Strings*

Web Services Sicherheit

- Web Service Security
  - IIS Security
    - Zugang zur WSDL
    - Zugang zum WS zur Laufzeit




```
◆ Message Actions
  ◆ On SAM_CreateComplete
    ◇ Call oTest.Test61DataTypes()
    ◆ If not SalWSSetCredentials(oTest,'JMG6510','devcon','devcon')
      ◇ Call SalMessageBox("invalid credentials : domain\username\password","Error",0)
```

WS – API Exception Handling

- **WS-API: SalThrowSoupFault(sError)**

  bOk = SalThrowSoapFault (sString)
  Sendet ein SoapFault an den Client
  - **Parameter**
    - **sString: Der Fehlertext**
  - **Return Value**
    - **bOk ist TRUE wenn die Funktion erfolgreich war**

```
When SqlError
  Call SqlGetError( hSQL, nError, sError )
  Call WriteTrace( sError )
  Call SalThrowSoapFault( sError )
```

1. SQL Fehler ermitteln und Throw...

```
Else
  ! Pass Error as Exception to client
  Call SalThrowSoapFault( 'No valid Company_ID' )
```

2. Throwing Exception zum Client

```
When Exception
  ! What should be done in case of an exceptions
  Call SalGetLastException(sgWinName,ngResError,sgErrorMessage,sgArrayStack)
  Call SalThrowSoapFault( sgErrorMessage )
```

3. Throwing a .NET exception to client

Exception handling .Net Client

- **Lokales Exception handling**

```
◆ Actions
  ◇ Set sContactID = SalNumberToStrX( nParContactID, 0 )
  ◆ When Exception
    ◇ ! What should be done in case of an exceptions
    ◇ Call SalGetLastException(sgWinName,ngResError,sgErrorMessage,sgArrayStack)
    ◇ Call SalModalDialog( dlgNetExceptions, hWndMDI )
  ◇ Call oContact.ContactDetails( sContactID, dfFirstName, dfLastName, dfTitle,
                                  dfPhoneCon, dfFaxCon, dfEmail )
  ◇ End Exception
  ◇ Return TRUE
```

Exception handling Win32 Client

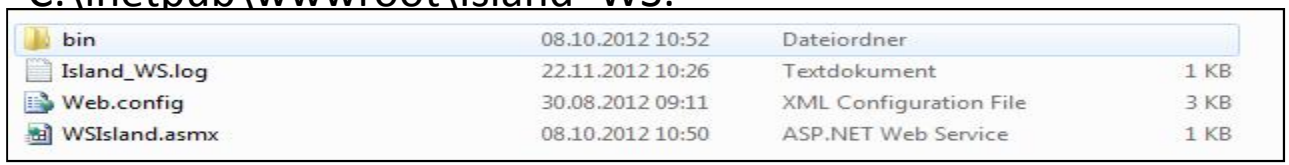
```
◆ Application Actions
  ◆ On SAM_NetException
    ◇ Set nButton =  SalMessageBox( '.NET Exception happends', 'Global Exception Handler', MB_RetryCancel | MB_IconStop )
    ◇ !
    ◇ ! Enter code to handel exception
    ◇ !
    ◇ ! Return TRUE => end error processing
    ◆ If nButton = IDRETRY
      ◇ Call SalMessageBox( 'Return TRUE: End error processing,
          do next steps...', 'Global Exception Handler', MB_Ok | MB_IconInformation )
      ◇ Return TRUE
    ◇ ! Return FALSE => continue error processing
    ◆ If nButton = IDCANCEL
      ◇ Call SalMessageBox( 'Return FALSE: Go ahead with error processing,
          do next steps...', 'Global Exception Handler', MB_Ok | MB_IconInformation )
      ◇ Return FALSE
```

Programmcode um eine .NET Exception in *SAM_NetException* abzufangen

Deploying Web Services im IIS

- **File Strukture im IIS (für das Island WS Beispiel)**

C:\inetpub\wwwroot\Island_WS:



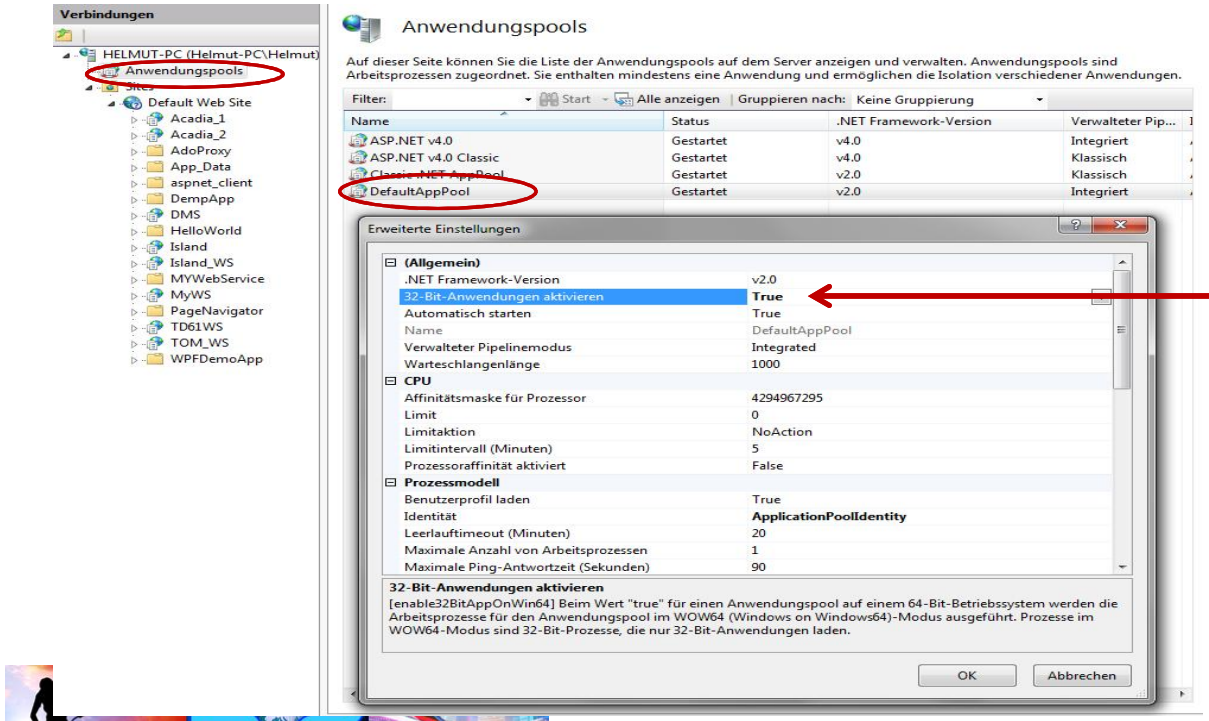
bin	08.10.2012 10:52	Dateiordner	
Island_WS.log	22.11.2012 10:26	Textdokument	1 KB
Web.config	30.08.2012 09:11	XML Configuration File	3 KB
WSIsland.asmx	08.10.2012 10:50	ASP.NET Web Service	1 KB

C:\inetpub\wwwroot\Island_WS\bin:

WS_Island.dll	09.10.2012 16:24	Anwendungserweiterung	36 KB

Hinweis: Island_WS.log = trace file für WS Fehler; generiert mit *SalWriteTrace*( )
WSIsland.asmx = TD Compiler generierte ASP.NET Web Service Beschreibung
Web.config = Konfigurationsfile mit Pfaden zu SQL.ini & TD .NET runtime
WS_Island.dll = TD-Compiler generierte Web Service DLL

Managing IIS imInetMgr.exe

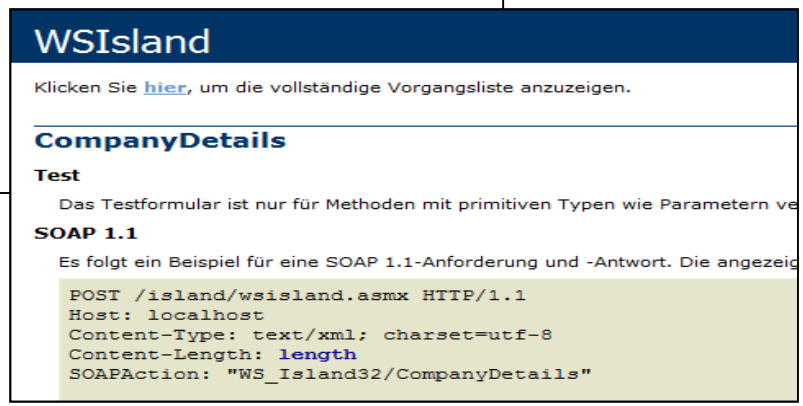


Diese Einstellung wird benötigt, um TD 6.2 Web Services im IIS verwenden zu können.

Test Web Service im Browser

- **Open URL in browser:**

Browser shows all Operations of WS

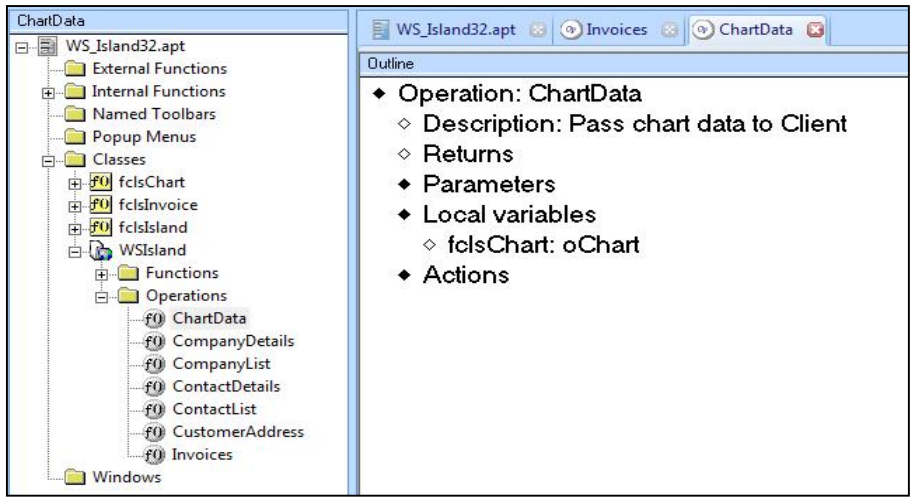


Select Operation to run

Besser: Testanwendung schreiben (Test von Exeption Handling, komplexen Datentypen usw.)

Debuggen von Web Services

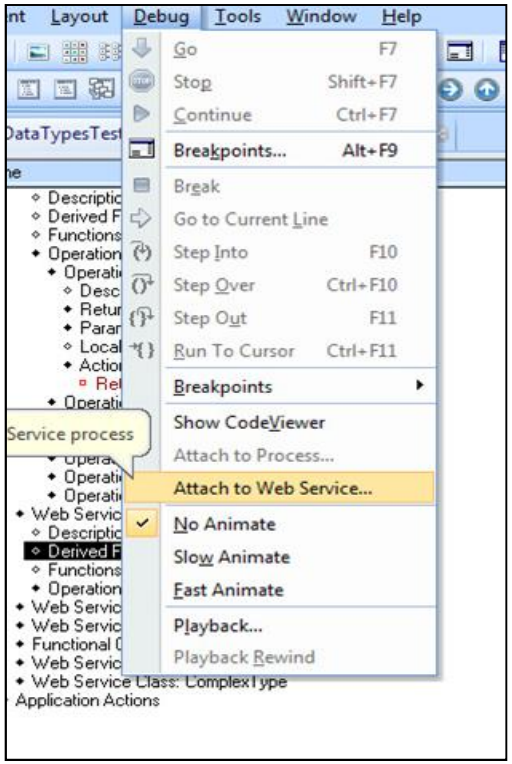
- **Beispiel: debuggen eines .NET Web Service:**



Starte TD6.2 als Administrator und öffne den Sourcecode in der Publish Directory innerhalb des IIS

Debuggen von Web Services

Attach den Sourcecode des Web Services zu dem laufenden WS im IIS



Debuggen von Web Services

- Operation: CompanyDetails
  - Description: Return company details
  - Returns
  - Parameters
  - Local variables
  - Actions
    - ! Check for valid input
    - Set bOk = ValidateNumber(nInCompanyID)
    - If bOk = TRUE
      - ! Set instance variab
      - Set oCompanyDetails.nClsCompany_ID = nInCompanyID

ValidateNumber(Number: nParIn)
Validate In-Parameter

Setze Breakpoint innerhalb der Source des WS

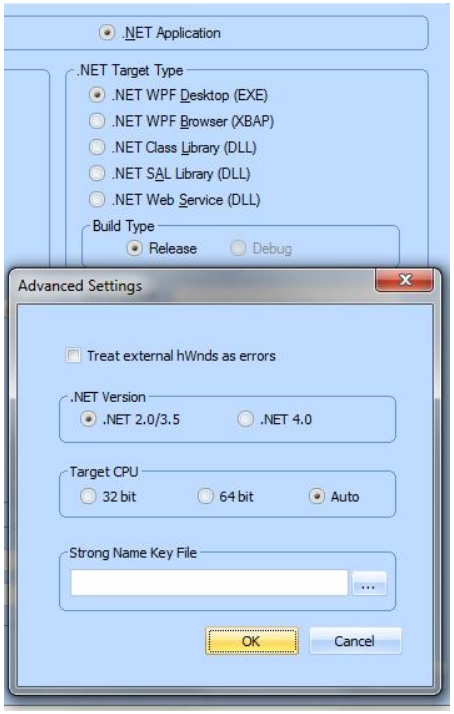
Debuggen von Web Services

```
◆ Operation: CompanyDetails
  ◇ Description: Return company details
  ◇ Returns
  ◆ Parameters
  ◆ Local variables
  ◆ Actions
    ◇ ! Check for valid input
    Set bOk = ValidateNumber( nInCompanyID )
    ◆ If bOk = TRUE
      ◇ ! Set instance variable for select
      ◇ Set oCompanyDetails.nClsCompany_ID = nInCompanyID
      ◇ ! Fire method
      ◇ Call oCompanyDetails.GetCompanyDetails( )
      ◇ ! Copy instance variables to receive values
      ◇ Set sOutCompany_Name = oCompanyDetails.sClsCompany_Name
      ◇ Set sOutAddress = oCompanyDetails.sClsAddress
      ◇ Set sOutCity = oCompanyDetails.sClsCity
      ◇ Set sOutState = oCompanyDetails.sClsState
```

Gehe schrittweise durch den Code des WS.....

64-Bit .NET Anwendungen

- *Advanced Settings* im Compiler Menü:
  - 32 bit
  - 64 bit
  - Auto
- Hängt von den verwendeten Komponenten ab (‚bitness‘ der Anwendung)

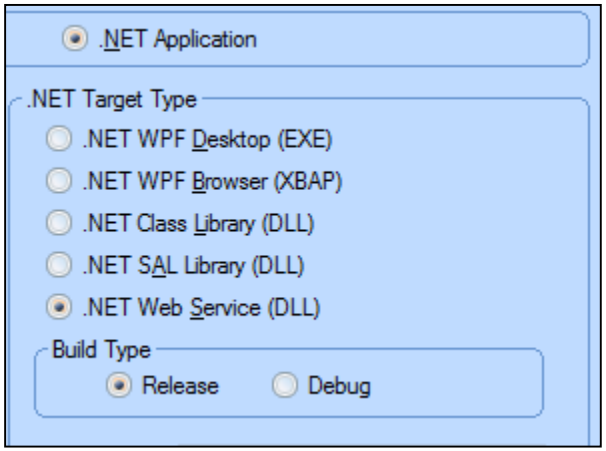


Debuggen Team Developer 6.2

- Neue Debug Möglichkeiten in TD6.2 .NET
  - Debuggen von .NET Anwendungen
  - Debuggen von .NET Anwendungen, die TD .NET SAL Libraries beinhalten
  - Debuggen von .NET Anwendungen, die C# Assemblies beinhalten
  - Debuggen von externen .NET Web Services Anbietern
  - Debuggen von C# Anwendungen, die TD .NET Assemblies beinhalten

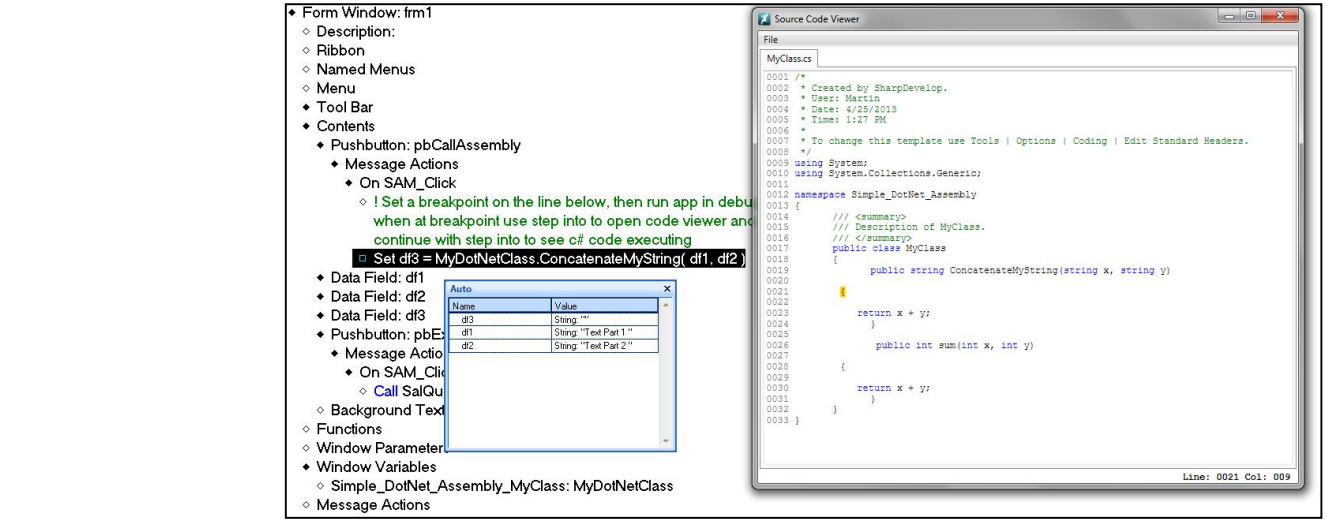
Debuggen Team Developer 6.2

- **Debugger Einstellungen für .Net:**
  - **Erzeugen der *.pdb Files**



Debuggen Team Developer 6.2

- **C# Assembly in Team Developer 6.2:**



Named Exceptions für .NET

- **Definition:**

```
◆ Named Exceptions
  ◆ Exception: eTerribleThingsHaveHappened
    ◇ Description: For demo
    ◆ Exception Variables
      ◇ String: sMessage ! Auto-generated, do not remove
      ◇ ! You can add any number of additional variables here;
          they must be included as parameters when you throw the exception.
      ◇ ! These variables can be accessed with SalGetLastExceptionField
        (see example in frmException.pbHandled On SAM_Click).
      ◇ String: sMoreMessage
      ◇ Number: nErrorNo
```

Name der Exeption

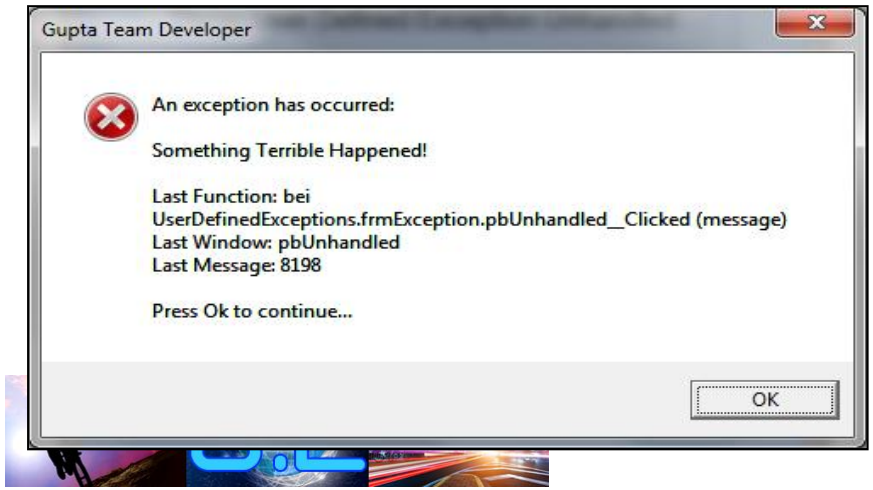
Standard Message

Zusätzliche, optionale Parameter

Beispiel und „Throw“

```
◇ ! Throw an unhandled exception
  (eTerribleThingsHaveHappened defined in Global Declarations | Named Exceptions )
◇ ! 2nd & 3rd Parameter: Additonal information defined in Named Exception for this exception
◇ Throw eTerribleThingsHaveHappened( 'Something Terrible Happened!',
  "This text is for the sMoreMessage variable in our Named Exception", 4711 )
```

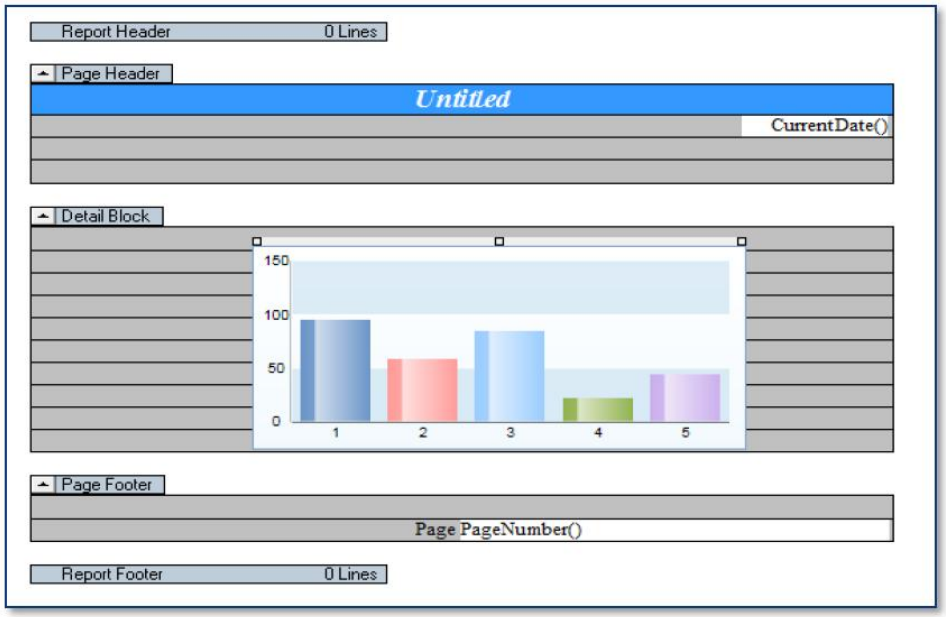
Neues Keyword ***Throw*** feuert die Exeption



Standard .NET Exception Handler Dialog wird angezeigt.

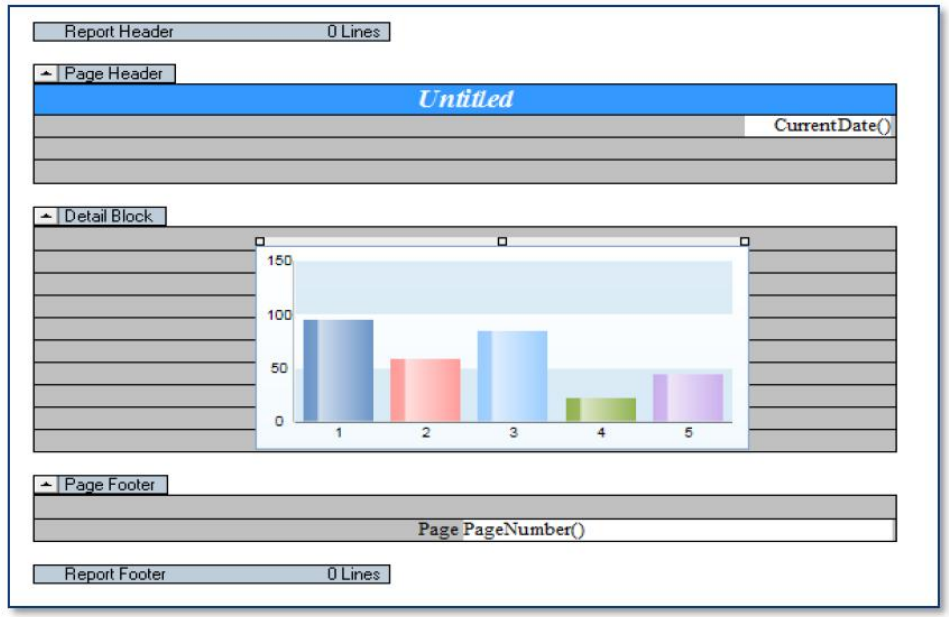
Neues Chart Control in Report Builder 6.2

- **Neues Object im RB 6.2:**



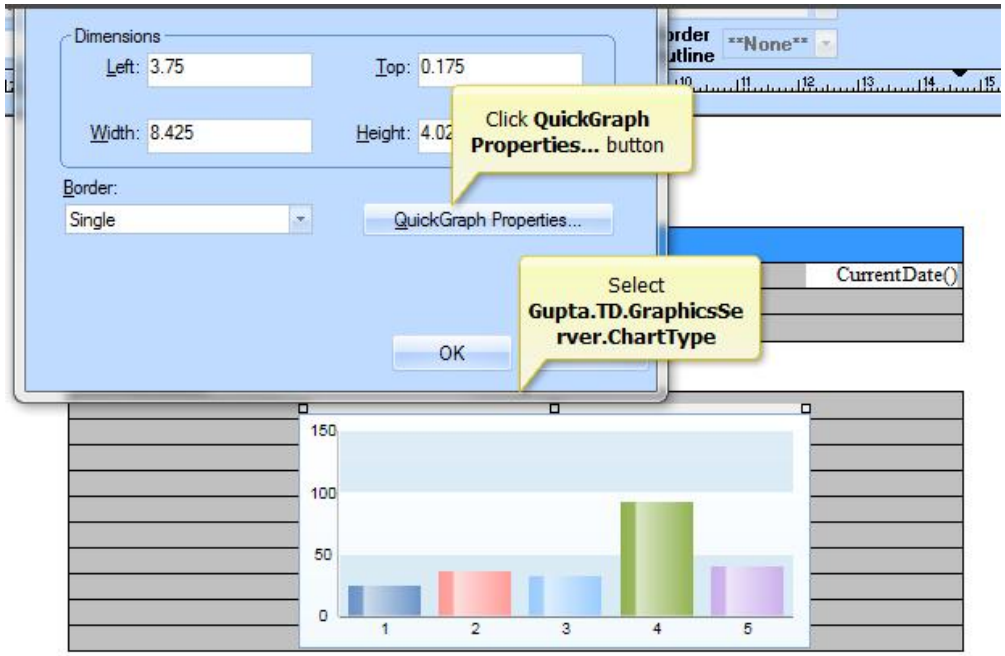
Neues Chart Control in Report Builder 6.2

- **Neues Objekt im RB 6.2:**



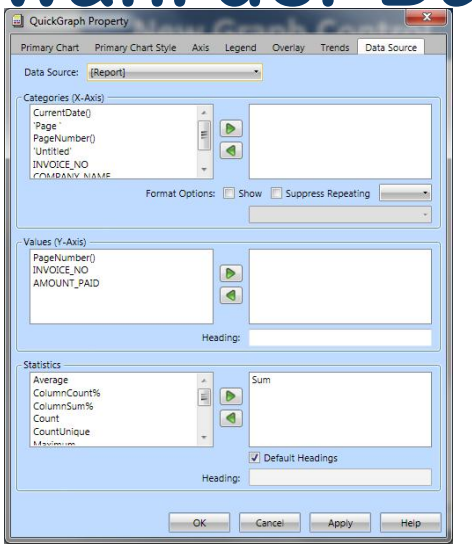
Neues Chart Control in Report Builder 6.2

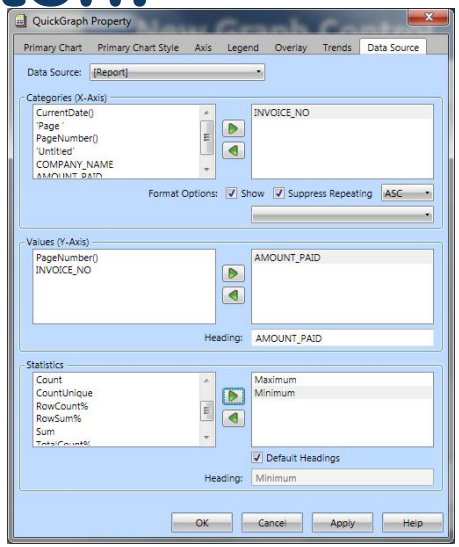
- Öffnen des Property – Dialogs und setzten der Eigenschaften:



Neues Chart Control in Report Builder 6.2

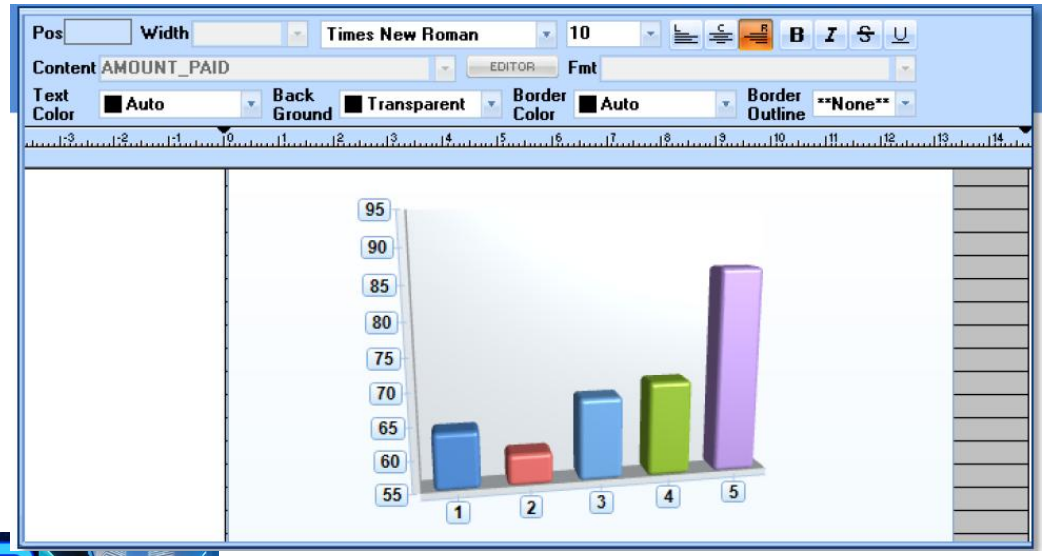
- Auswahl der Daten:





Neues Chart Control in Report Builder 6.2

- **Fertiges Chart Diagramm:**



Was fehlt in der Präsentation?

- .NET
  - Arbeiten mit Resourcen
  - Lokalisieren von Anwendungen
  - Exception Handling (Lokal, Throw, Rethrow, Rethrow global, When Exception / End Exception)
  - Debuggen im Detail
  - Arbeiten mit Web Services (Komplexe Datentypen)

GUPTA

Roadshow 2013

Team Developer 6.2