



GUPTA

Roadshow 2015

Team Developer 6.3 .NET

Helmut Reimann

Team Developer 6.3 .Net

- **.Net Assemblies einbinden**
- **Exception Handling**
- **Web Services**

.Net Assemblies

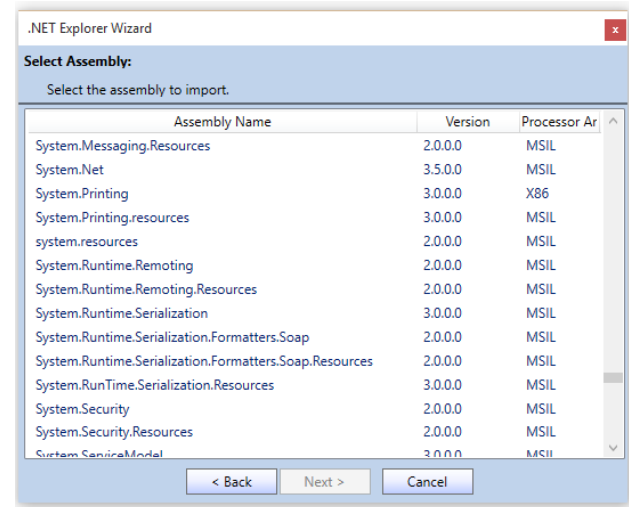
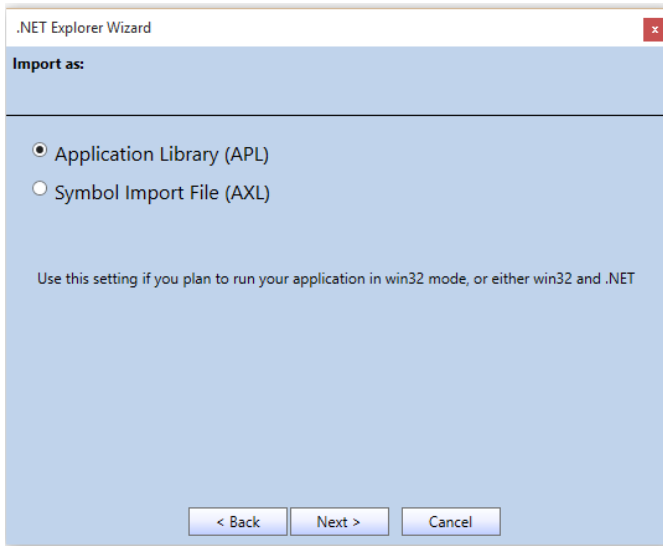
- .Net Assemblies können in Win32 und in .Net Anwendungen eingebunden werden
 - .Net Explorer:

Achtung: In der Windows Version des Team Developers ist der .Net Explorer ein separates Tool!



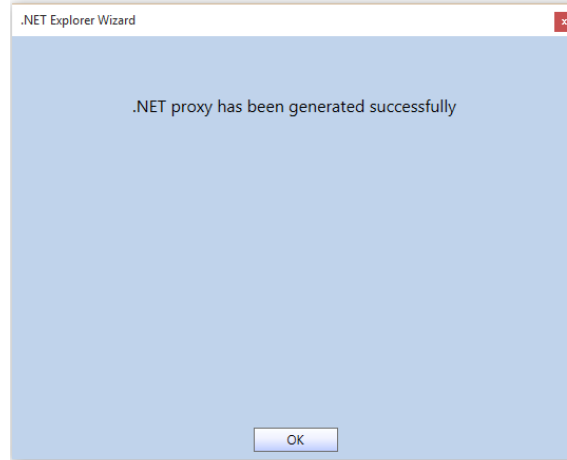
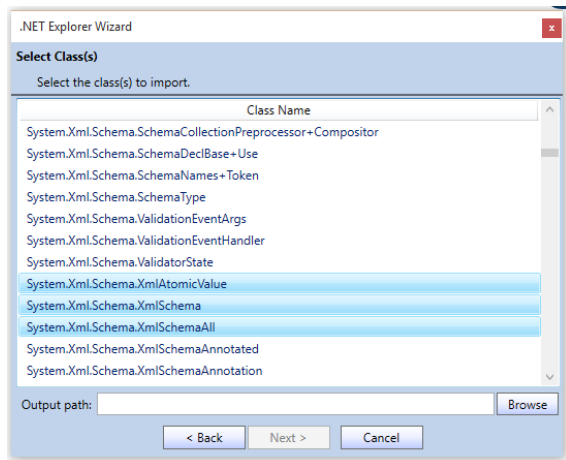
.Net Assemblies

- Optionen:



.Net Assemblies

- **Klassen auswählen**



<input type="checkbox"/>	System.Xml.apl	09.11.2015 15:57	Application Library
<input checked="" type="checkbox"/>	System.Xml.axl	09.11.2015 15:57	TD.NET Symbol I...

.Net Explorer

- ◆ Application Description: Gupta SQLWindows
Standard Application Template
- ◆ Libraries
 - ◇ File Include: System.Xml.apl
 - ◇ File Include: GAILBase.apl
- ◆ Global Declarations
 - ◆ Window Defaults
 - ◆ Formats
 - ◆ External Functions
 - ◆ External Assemblies
 - ◇ Symbol Import: System.Xml.axl
 - ◆ Constants
 - ◇ Resources
 - ◆ Variables
 - ◆ Internal Functions
 - ◇ Named Exceptions
 - ◇ Named Toolbars
 - ◆ Named Menus
 - ◆ Class Definitions

Je nach Compiler Setting wird
die APL oder die AXL
eingebunden

- ◆ Class Definitions
 - ◆ Functional Class: System_Xml_Schema_XmlSchemaAll
 - ◆ Functional Class: System_Xml_Schema_XmlAtomicValue
 - ◆ Functional Class: System_Xml_Schema_XmlSchema
 - ◆ Functional Class: System_Xml_Schema_XmlSchemaObjectCollection
 - ◆ Functional Class: System_Xml_Schema_XmlSchemaGroupBase
 - ◆ Functional Class: System_Xml_Schema_XmlSchemaType
 - ◆ Functional Class: System_Xml_IXmlNamespaceResolver

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

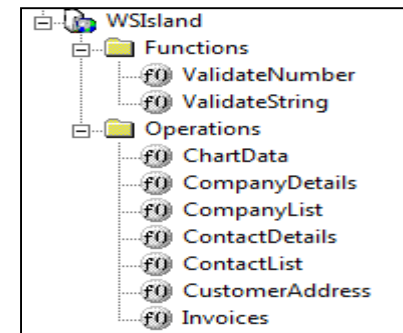
Web Services

- Verlagern von Business Logik auf einen zentralen MS-IIS Server
 - Stateless Datenbankzugriffe

```
▾ Actions
  ▮ Set sSQL = 'select T_ID, Typ, Termin, Bezeichnung, Verein from Ter
    :oTermin.nTerminID, :oTermin.nTerminTyp, :oTermin.dDatum, :oTer
    where T_ID = :npTerminID '
  ▮ Set ngSQL_ID = 1421
  ▮ If GConnect()
    ▮ If SqlPrepareAndExecute(ghSQL, sSQL)
      ▮ Call SqlFetchNext(ghSQL, nRet)
      ▮ Call SqlDisconnect(ghSQL)
    ▮ Return oTermin
```

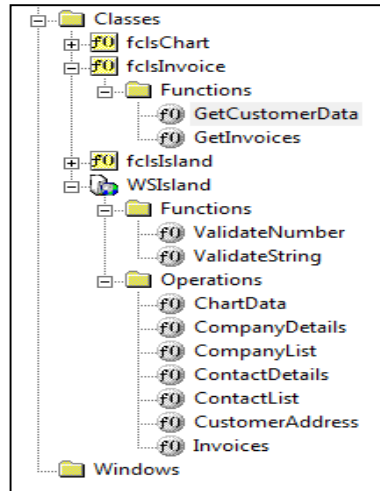

Neuer Klassen Typ: Web Service

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



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.sClsCorporate_Url
 - ◇ Return TRUE
 - ◆ Else
 - ◇ Return FALSE

Einfache Operation (Arrays von Receive Parameter)

• Programm Code mit Arrays als Receive Parameter:

- ◆ 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 variables 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

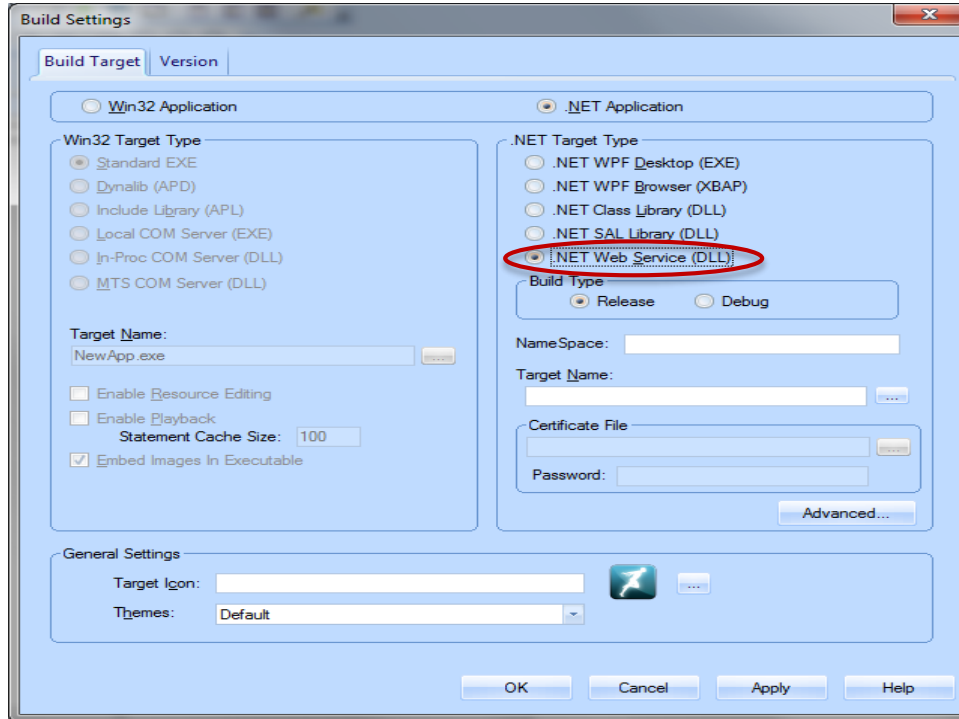
Einfache Operation (Return UDV)

• 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.)

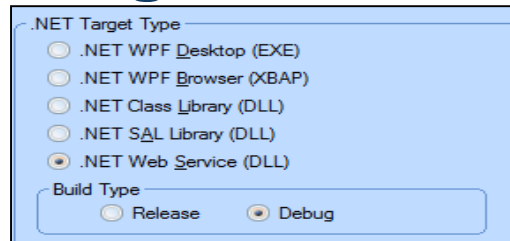
Nutzt **SalThrowSoapFault()**, um Exceptions oder SQL Fehler zum Client zu senden.

.NET Compiler Einstellungen

- **Compiler generierte Files:**

 WS_Island	10/10/2013 2:30 PM	68 KB
 WS_Island.dll	10/10/2013 2:30 PM	36 KB
 WS_Island	10/10/2013 2:30 PM	28 KB
 WSIsland	10/10/2013 2:30 PM	1 KB

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



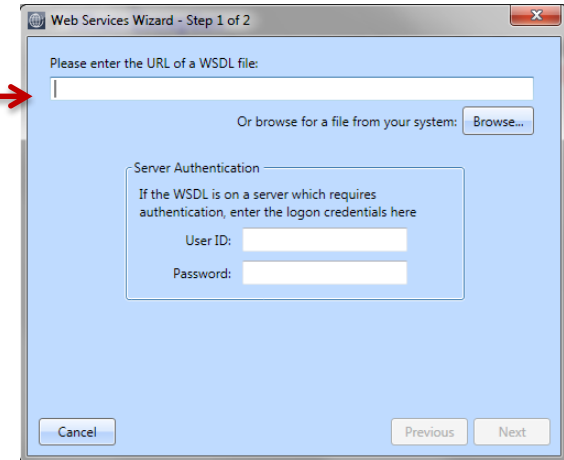
Option: ‚Debug‘:
Generiert zusätzlich
WS_Island.pdb

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: SalThrowSoapFault(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 )
```

1a. SQL Fehler ermitteln und Throw...

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

1b. Throwing einer ,User' Exception zum Client

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

1c. 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

Deploying Web Services im IIS

- **File Struktüre 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 in IIS Manager

The screenshot shows the IIS Manager interface. On the left, the 'Verbindungen' pane shows the tree structure with 'Anwendungspools' circled in red. The main pane displays a list of application pools:

Name	Status	.NET Framework-Version	Verwalteter Pip...
ASP.NET v4.0	Gestartet	v4.0	Integriert
ASP.NET v4.0 Classic	Gestartet	v4.0	Klassisch
Classic .NET AppPool	Gestartet	v2.0	Klassisch
DefaultAppPool	Gestartet	v2.0	Integriert

The 'DefaultAppPool' is circled in red. Below it, the 'Erweiterte Einstellungen' dialog box is open, showing the 'Allgemein' tab. The '32-Bit-Anwendungen aktivieren' checkbox is checked and highlighted in blue, with a red arrow pointing to it from the text on the right.

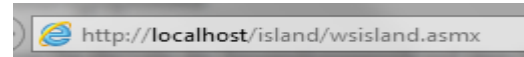
Erweiterte Einstellungen	
(Allgemein)	
.NET Framework-Version	v2.0
32-Bit-Anwendungen aktivieren	<input checked="" type="checkbox"/> True
Automatisch starten	True
Name	DefaultAppPool
Verwalteter Pipelinemodus	Integrated
Warteschlangenlänge	1000
CPU	
Affinitätsmaske für Prozessor	4294967295
Limit	0
Limitaktion	NoAction
Limitintervall (Minuten)	5
Prozessoraffinität aktiviert	False
Prozessmodell	
Benutzerprofil laden	True
Identität	ApplicationPoolIdentity
Leerlaufzeit (Minuten)	20
Maximale Anzahl von Arbeitsprozessen	1
Maximale Ping-Antwortzeit (Sekunden)	90

32-Bit-Anwendungen aktivieren
[enable32BitAppOnWin64] Beim Wert "true" für einen Anwendungspool auf einem 64-Bit-Betriebssystem werden die Arbeitsprozesse für den Anwendungspool im WOW64 (Windows on Windows64)-Modus ausgeführt. Prozesse im WOW64-Modus sind 32-Bit-Prozesse, die nur 32-Bit-Anwendungen laden.

Diese Einstellung wird benötigt, um TD 6.3 Web Services im IIS verwenden zu können!

Test Web Service im Browser

- Öffnen der URL in browser:



WSIsland

Folgende Vorgänge werden unterstützt. Eine ausführliche [Definition finden](#) Sie in der [Dienstbeschreibung](#).

- [ChartData](#)
- [CompanyDetails](#)
- [CompanyList](#)
- [ContactDetails](#)
- [ContactList](#)
- [CustomerAddress](#)
- [Invoices](#)

WSIsland

Klicken Sie [hier](#), um die vollständige Vorgangsliste anzuzeigen.

CompanyDetails

Test

Das Testformular ist nur für Methoden mit primitiven Typen wie Parametern verfügbar.

SOAP 1.1

Es folgt ein Beispiel für eine SOAP 1.1-Anforderung und -Antwort. Die angezeigten Werte sind nur für die Darstellung und sind nicht für die Ausführung geeignet.

```
POST /island/wsisland.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "WS_Island32/CompanyDetails"
```

Select Operation to run

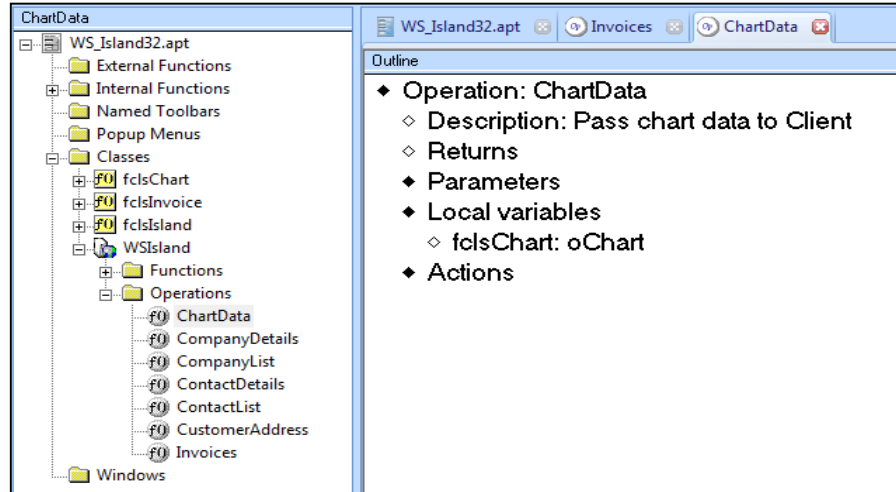
WSIsland WSDL

```
<?xml version="1.0" encoding="utf-8" ?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://microsoft.com/wsdl/mime/textMatch/" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:base64="http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" targetNamespace="WS_Island32" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" >
  <wsdl:types >
    <xs:import elementFormDefault="qualified" targetNamespace="WS_Island32" />
    <xs:complexType name="CompanyList">
      <xs:sequence>
        <xs:element minOccurs="0" maxOccurs="1" name="nOutCompanyIDs" type="tns:ArrayOfSalDecimal" />
        <xs:element minOccurs="0" maxOccurs="1" name="nOutCompanyNames" type="tns:ArrayOfString" />
        <xs:element minOccurs="0" maxOccurs="1" name="nOutCompanyCities" type="tns:ArrayOfString" />
        <xs:element minOccurs="0" maxOccurs="1" name="nOutCompanyPhones" type="tns:ArrayOfString" />
        <xs:element minOccurs="1" maxOccurs="1" name="nOutResultCount" type="tns:SalDecimal" />
      </xs:sequence>
    </xs:complexType>
  </wsdl:types>
  <wsdl:binding name="ArrayOfSalDecimal" type="tns:ArrayOfSalDecimal">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="SalDecimal" type="tns:SalDecimal" />
    </xs:sequence>
  </wsdl:binding>
  <wsdl:binding name="SalDecimal" type="tns:SalDecimal">
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="1" name="Value" type="tns:SalDecimal" />
    </xs:sequence>
  </wsdl:binding>
  <wsdl:binding name="ArrayOfString" type="tns:ArrayOfString">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="string" nillable="true" type="xs:string" />
    </xs:sequence>
  </wsdl:binding>
  <wsdl:binding name="CompanyListResponse" type="tns:CompanyListResponse">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="nOutCompanyIDs" type="tns:ArrayOfSalDecimal" />
      <xs:element minOccurs="0" maxOccurs="1" name="nOutCompanyNames" type="tns:ArrayOfString" />
      <xs:element minOccurs="0" maxOccurs="1" name="nOutCompanyCities" type="tns:ArrayOfString" />
      <xs:element minOccurs="0" maxOccurs="1" name="nOutCompanyPhones" type="tns:ArrayOfString" />
      <xs:element minOccurs="1" maxOccurs="1" name="nOutResultCount" type="tns:SalDecimal" />
    </xs:sequence>
  </wsdl:binding>
  <wsdl:service name="CompanyListService" base="tns:CompanyListServiceBase" >
    <wsdl:operation name="GetCompanyList" binding="tns:ArrayOfSalDecimal" />
    <wsdl:operation name="GetSalDecimal" binding="tns:SalDecimal" />
    <wsdl:operation name="GetArrayOfString" binding="tns:ArrayOfString" />
    <wsdl:operation name="GetCompanyListResponse" binding="tns:CompanyListResponse" />
  </wsdl:service>
</wsdl:definitions>
```

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

Debuggen von Web Services

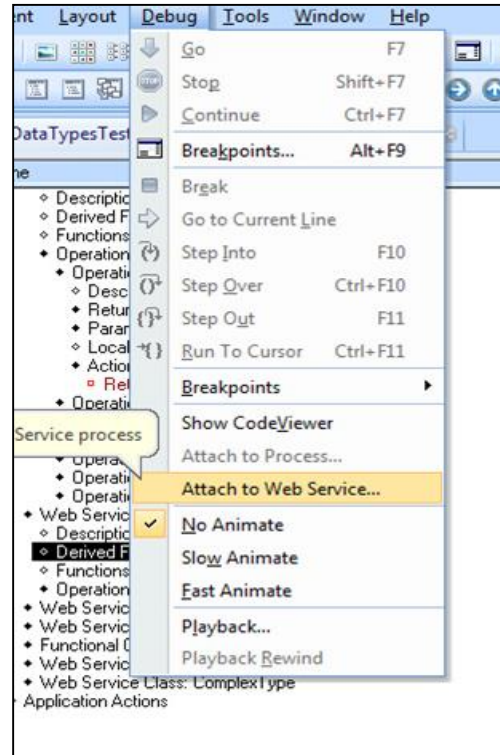
- **Beispiel: Debuggen eines .NET Web Service:**



Starte TD6.3 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 ValidateNumber(Number: nParIn)
Validate In-Parameter
 - ◇ Set oCompanyDetails.nClsCompany_ID = nInCompanyID

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.....

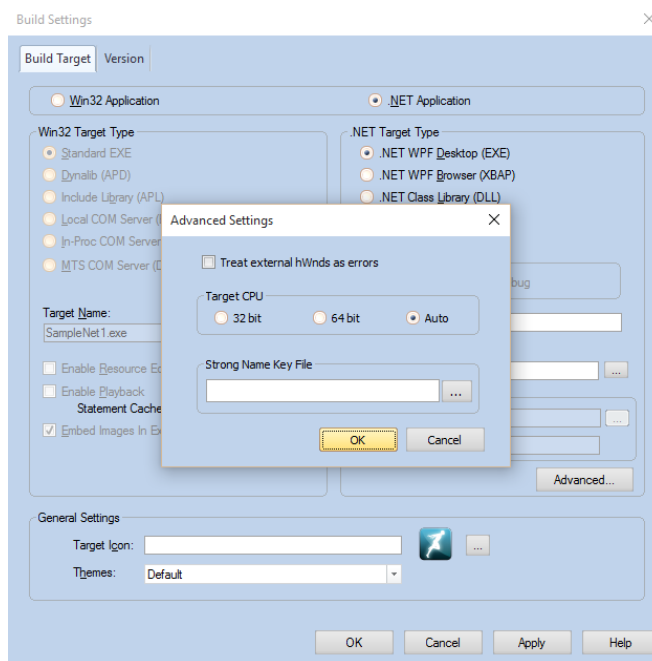
Exception handling Win32 Client

- ◆ Application Actions
 - ◆ On SAM_NetException
 - ◇ Set nButton = SalMessageBox('.NET Exception happens', '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

.Net 64 Bit

- **Compiler Setting**



Achtung: Bitness muss
übereinstimmen!

Team Developer 6.3 .Net Anwendungen

- **Ja, das ist realisierbar!**
 - **Es sind allerdings einige Vorarbeiten zu erledigen!**
 - **Auskunft gibt der Upgrade-Report des TD6.x**

Sal Web Reporting Funktionen

- **Zwei Sal Funktionen stellen eine Web-Reporting Funktionalität innerhalb einer TD Mobile oder Team Developer .Net Anwendung zur Verfügung:**
 - **SalWebReportCreateToFile()**
 - **SalWebReportCreate()**
- **Anwendungsbeispiele**
 - **Generierung einer Rechnung und Versenden als Anhang einer Mail**
 - **Generieren eines Dokumentenarchivs für eine spätere Nutzung**

SalWebReportCreateToFile()

- **Generiert einen Reportfile basierend auf Daten und einem Report Template**
 - **Ausgabe kann entweder PDF oder HTML sein, abhängig vom letzten Argument der entsprechenden Sal Funktion**
- **Das generierte Dokument verbleibt auf dem IIS Server**
- **Gibt dem Programmierer die Möglichkeit, Report zu erzeugen um sie dann SalFile* Funktionen zu verarbeiten**

SalWebReportCreateToFile()

Syntax

- `bOk = SalWebReportCreateToFile(sRPXFile, sOutputFile, reportVars, reportData, nOutputType, sLang)`
- **Arguments:**
 - `sRPXFile`: (String), the name of the RPX file which defines the report layout. This file is expected to be found on the web server and the path, if any, is relative to the home location of the web service which is running
 - `sOutputFile`: (String), the name of the output file which will be generated. This file will be generated on the web server and its path, if any, is relative to the home location of the web service which is running
 - `reportVars`: (UDV) an instance of a class whose member fields contain any “report variables” to be used by the report engine. The names of the class fields must match the names of the report variables as defined in the RPX file.
 - `reportData`: (UDV array) an array of class objects whose member fields contain the report “data items”. The names of the class fields must match the names of the report variables as defined in the RPX file.
 - `nOutputType`: (Number) a constant specifying the kind of report file to generate:
 - `WRPT_OutputPDF`: A PDF file
 - `WRPT_OutputHTML`: An HTML file
 - `sLang`: (String) The language to use when generating the report (e.g. “en-US”)
- **Example:**
 - Call `SalWebReportCreateToFile("Invoice.rpx", "report.pdf", reportVars, inputItems, WRPT_OutputPDF, "en-US")`

SalWebReportCreate()

- **Einfache Funktion zum Erzeugen eines serverseitigen Reports und direkter Übertragung als Binary Datenstream**
- **Sinnvoll, wenn der Anwender keine Schreibberechtigung im APP-Pool auf dem IIS hat**

SalWebReportCreate() Syntax

- `bOk = SalWebReportCreate(sRPXFile, binOutputFile, reportVars, reportData, nOutputType, sLang)`
- **Arguments:**
 - `sRPXFile`: (String), the name of the RPX file which defines the report layout. This file is expected to be found on the web server and the path, if any, is relative to the home location of the web service which is running
 - `binOutputFile`: (Receive Binary), the generated file, either PDF or HTML, in binary format
 - `reportVars`: (UDV) an instance of a class whose member fields contain any “report variables” to be used by the report engine. The names of the class fields must match the names of the report variables as defined in the RPX file.
 - `reportData`: (UDV array) an array of class objects whose member fields contain the report “data items”. The names of the class fields must match the names of the report variables as defined in the RPX file.
 - `nOutputType`: (Number) a constant specifying the kind of report file to generate:
 - `WRPT_OutputPDF`: A PDF file
 - `WRPT_OutputHTML`: An HTML file
 - `sLang`: (String) The language to use when generating the report (e.g. “en-US”)
- **Example:**
 - Call `SalWebReportCreate("Invoice.rpx", bMyReport, reportVars, inputItems, WRPT_OutputPDF, “en-US”)`



Fragen?