



Adesso Plug-Ins

Development Guide

Copyright

Neither the documentation nor the software may be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form, in whole or in part, without the prior written consent of Adesso Systems, Inc., except in the manner described in the documentation.

Adesso Plug-Ins Development Guide, version 1.0.0 for Adesso Mobility Platform 3.0.0

©Copyright 2005. Adesso Systems, Inc., 1 Liberty Square, Boston, MA 02109.
All rights reserved.

Contents

Contents	3
1. Introduction	5
1.1. Overview.....	5
1.2. Required Form	5
1.3. Supported Platforms.....	5
1.4. Architecture	6
2. Plug-In Interface	6
2.1. Handler.....	7
2.2. GetFieldNames	7
2.3. GetParameterNames	7
3. XML Definition	8
3.1. XSD XML Schema	9
3.2. XML Request Example.....	10
3.3. Adesso Form Controls.....	21
3.3.1. Supported Form Controls	21
3.3.2. Non-Supported Form Controls	21
3.3.3. Supported Data Types	21
3.4. XML Elements & Attributes	22
3.4.1. Color Definition.....	31
4. Sample Plug-In Implementation	32
5. Metadata Extractor Plug-Ins	35
5.1. Sample Metadata Extractor Implementation.....	35
6. Debug Support	35
7. Building Plug-Ins.....	36
8. Security	36
9. Features Not Yet Supported.....	36
10. Plug-In Implementation in the Adesso Client.....	37
10.1. Ensuring Plug-In/Client Compatibility.....	37
10.2. Plug-In Designer.....	38
10.2.1. Plug-In Tab.....	39
10.2.2. Additional Parameters.....	41
10.2.3. Field Filter Tab	42
10.2.4. Field Name Map Tab.....	43
10.2.5. Other Tab	44
10.3. Form Designer.....	45
10.4. Table Designer	45
10.4.1. EXEC Function.....	46
10.5. View Designer	47
Index	48

BACK PAGE OF CONTENTS: Intentionally Blank

1. Introduction

This document describes how to construct and implement a plug-in for the Adesso Client, release 3.0. It includes a complete description of the plug-in interface and the XSD XML Schema, as well as an XML request example. The final section of the document describes how to register a plug-in using the Adesso Client's Plug-In Designer, and how to associate the plug-in with other Adesso design elements via the Client UI.

1.1. Overview

Adesso plug-ins is a general purpose architecture that provides the ability for 3rd party (including Adesso itself) processes to interact with specific Adesso applications. These 3rd party processes should be thought of as "Handlers", or "Custom Extensions", managed by the Adesso plug-in Manager.

These processes can be considered part of a given Adesso distributed application thereby integrating multiple external, heterogeneous applications into a single workflow. The external processes may be tied to multiple Adesso design elements, including expressions, button-pressing events, record open/new events, data validation, form navigation, and so on.

Examples of plug-in uses are as varied as Adesso Applications. They can include an activity as simple as performing a customized print job to interfacing with a Web Service to more complex processes that involve verifying or collecting information for the Adesso application.

1.2. Required Form

For Adesso release 3.0, a plug-in must be created as a .NET assembly. It is recommended that these assemblies be implemented as Smart Device Application or Class Library projects to support the Pocket PC platform as well as Windows.

Plug-ins may be associated with the following Adesso Client constructs and will trigger a plug-in Manager operation:

- Expressions—The EXEC command can be used for Field Selection calculations, Validation Expressions, etc.
- Form Button—Used for triggering external processes, custom printing, interactions with Web Services, etc.
- Events
 - New Record—Used to pre-process/load new record fields and form attributes from either the Adesso Form or an alternative Form Editor.
 - Open Record—Used to pre-process/load existing record fields and form attributes for the Adesso Form or an alternative Form Editor.
- Metadata Extractors—Programs designed to extract metadata from a specific file type. If a metadata extractor is registered with an application, the function GETMETADATA will automatically attempt to run the program.

1.3. Supported Platforms

The plug-in API supports the same platforms supported by Adesso applications:

- Windows
- Pocket PC 2003

To build a plug-in that utilizes web Services, you need to build device-specific plug-ins for .NET 1.1 and Compact Framework 1.0.

To build a plug-in that utilizes .NET 2 (Visual Studio 2005), then you must have a specially built AdessoInterfaces and SHIMs provided to you by Adesso Systems. A release that addresses this will be available when Visual Studio 2005 is officially shipping.

1.4. Architecture

A file-based SHIM approach is employed for both the Desktop and Pocket PC platforms for inter-process communications. All Handler Request and Response operations are done synchronously. The plug-in Assembly and its dependencies are stored as binary properties within the associated application database and synchronized with the Adesso Server.

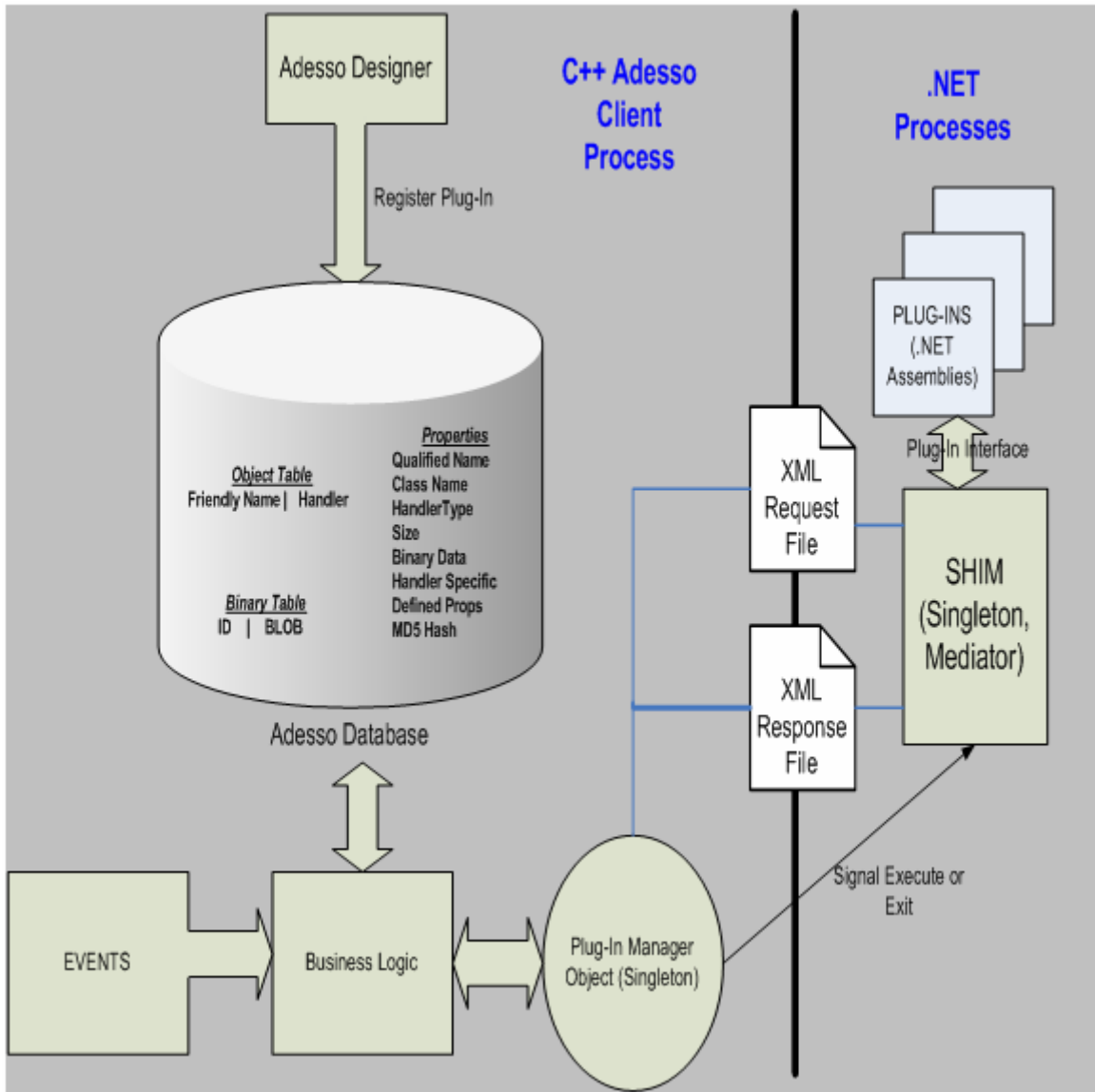


Figure 1 – Adesso Plug-In Architecture

2. Plug-In Interface

The plug-in interface is an abstract interface defined by Adesso that 3rd party plug-in developers must implement. The IExtensionHandler Interface is defined in AdessoInterfaces.dll – a strong name signed assembly that is produced and shipped by Adesso Systems. An implementation of a plug-in must implement the IExtensionHandler interface and should also be implemented as a “Smart Device Class Library” project. By doing this, the Class file and its implementation can be executed on both the .NET Compact Framework as well as the Full .NET Framework. Requests and responses must follow the XSD definition. Requests to the plug-in are contained in the XmlDocument parameter. Responses from the plug-in are returned in the string return value and must contain the complete XML document with changes made by the plug-in.

```
public interface IExtensionHandler
{
    string Handler(System.Xml.XmlDocument xmlDoc);
    string GetFieldNames(XmlDocument xmlDoc);
    string GetParameterNames(XmlDocument xmlDoc);
}
```

2.1. Handler

Used for handling all event types.

2.2. GetFieldNames

Used for obtaining a list of names to be used when designing [Field Name Mappings](#).

Request XML

```
<EventHandler Type="EventsNOUI" Name="SpeechEventHandler"
  Assembly="C:\Documents and Settings\mpalone\My Documents\Visual
  Studio Projects\SpeechEventHandler\bin\Debug\SpeechEventHandler.dll"
  Class="Adesso.Client.SpeechEventHandler" GUID="{0E0FA600-97BB-11D9-305E-
  005E27860124}">
  <FieldNames />
</EventHandler>
```

Response XML

```
<EventHandler Type="EventsNOUI" Name="SpeechEventHandler"
  Assembly="C:\Documents and Settings\mpalone\My Documents\Visual
  Studio Projects\SpeechEventHandler\bin\Debug\SpeechEventHandler.dll"
  Class="Adesso.Client.SpeechEventHandler" GUID="{0E0FA600-97BB-11D9-305E-
  005E27860124}">
  <FieldNames>
    <FieldName>Test1</FieldName>
    <FieldName>Test1</FieldName>
  </FieldNames>
</EventHandler>
```

```
// NOTE: If you not want to return values for GetParameterNames or
GetFieldNames, return xmlDoc.DocumentElement.OuterXml or a blank string (eg. return
"");
string IExtensionHandler.GetParameterNames(XmlDocument xmlDoc)
{
    return xmlDoc.DocumentElement.OuterXml;
}
```

2.3. GetParameterNames

Used for obtaining a list of names to be used when [designing Parameters](#).

Request XML Fragment:

```
<EventHandler Type="EventsNOUI" Name="SpeechEventHandler"
  Assembly="C:\Documents and Settings\mpalone\My Documents\Visual
  Studio Projects\SpeechEventHandler\bin\Debug\SpeechEventHandler.dll"
  Class="Adesso.Client.SpeechEventHandler" GUID="{0E0FA600-97BB-11D9-305E-
  005E27860124}">
  <ParameterNames />
</EventHandler>
```

Response XML Fragment

```
<EventHandler Type="EventsNOUI" Name="SpeechEventHandler"
  Assembly="C:\Documents and Settings\mpalone\My Documents\Visual
  Studio Projects\SpeechEventHandler\bin\Debug\SpeechEventHandler.dll"
  Class="Adesso.Client.SpeechEventHandler" GUID="{0E0FA600-97BB-11D9-305E-
  005E27860124}">
  <ParameterNames>
    <ParameterName>Test1</ParameterName>
    <ParameterName>Test2</ParameterName>
  </ParameterNames>
</EventHandler>
```

```
// NOTE: If you not want to return values for GetParameterNames or
GetFieldNames, return xmlDoc.DocumentElement.OuterXml or a blank string (eg. return
"");
string IExtensionHandler.GetParameterNames(XmlDocument xmlDoc)
{
  return xmlDoc.DocumentElement.OuterXml;
}
```

3. XML Definition

XML messages (fragments) are used for requests to Handlers and for resulting responses back from them. Initially this will be done through file-based SHIM inter-process communications. What is contained in the message is dependent on the type of Handler the message is bound for.

Consider the following XML message:

```
<EventHandler Type="Form Button">
  <Form Name="UserInput" ID="{guid}">
    <Tab Name="Tab1" Enabled="1" Visible="1">
      <Field Name="Field1" Enabled="1" Visible="1">Value1</Field>

      <Field Name="Field2" Enabled="1" Visible="1">Value2</Field>
    </Tab>
  </Form>
</EventHandler>
```

The above fragment would be passed as a parameter to the event handler. The event handler is free to do whatever processing it deems appropriate. In order to return results to the Plug-In Manager, the called handler simply transforms the Xml and returns it.

For example, to disable the 2nd field on the form the handler would return the following fragment:

```
<EventHandler Type="Form Button"
Assembly=C:\Documents and Settings\mpalone\My Documents\My Adesso
Applications\handlers\AdessoEventHandler.dll
Class=Adesso.Client.CosineHandler>
  <Form Name="UserInput" ID="{guid}">
    <Tab Name="Tab1">
      <Field Name="Field1" Enabled="1" Visible="1">Value1</Field>
      <Field Name="Field2" Enabled="0" Visible="1">Value2</Field>
    </Tab>
  </Form>
</EventHandler>
```

For example, to return a different value for a field, the handler would return the following fragment:

```
<EventHandler Type="Form Button"
Assembly=C:\Documents and Settings\mpalone\My Documents\My Adesso
Applications\handlers\AdessoEventHandler.dll
Class=Adesso.Client.CosineHandler>
  <Form Name="UserInput" ID="{guid}">
    <Tab Name="Tab1" Enabled="1" Visible="1">
      <Field Name="Field1" Enabled="1" Visible="1" >New Value</Field>
      <Field Name="Field2" Enabled="1" Visible="1"/>Value2</Field>
    </Tab>
  </Form>
</EventHandler>
```

3.1. XSD XML Schema

The Example that follows defined the XML Schema using XSD. XML Requests are sent to the Plug-In through the Plug-In Manager. XML Responses follow this same schema and are sent from the Plug-In to the Plug-In Manager.

3.2. XML Request Example

Note: XML Responses take on this same format with the addition of the optional <error>, <Save>, <FormAction>, and <Result> tags. The return response may also be an empty string ("").

```
<?xml version="1.0" encoding="UTF-8" ?>
<root>
  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <!-- definition of simple elements -->
    <xs:element name="USERNAME" type="xs:string" />
    <xs:element name="USERID" type="xs:string" />
    <xs:element name="LOCATION" type="xs:string" />
    <xs:element name="RADIUS" type="xs:integer" />
    <xs:element name="TOTALRECORDS" type="xs:string" />
    <xs:element name="CURRENTRECORDINDEX" type="xs:string" />
    <!-- definition of attributes -->
    <xs:attribute name="Name" type="xs:string" />
    <xs:attribute name="DisplayName" type="xs:string" />
    <xs:attribute name="Type" type="xs:string" />
    <xs:attribute name="DataType" type="xs:string" />
    <xs:attribute name="Value" type="xs:string" />
    <xs:attribute name="TableName" type="xs:string" />
    <xs:attribute name="ControlType" type="xs:string" />
    <xs:attribute name="Assembly" type="xs:string" />
    <xs:attribute name="Class" type="xs:string" />
    <xs:attribute name="GUID" type="xs:string" />
    <xs:attribute name="ID" type="xs:string" />
    <xs:attribute name="bgcolor" type="xs:string" />
    <xs:attribute name="textcolor" type="xs:string" />
    <xs:attribute name="height" type="xs:integer" />
    <xs:attribute name="width" type="xs:integer" />
    <xs:attribute name="Enabled" type="xs:boolean" />
    <xs:attribute name="Visible" type="xs:boolean" />
    <xs:attribute name="IsDirty" type="xs:boolean" />
    <xs:attribute name="multiselect" type="xs:boolean" />
    <xs:attribute name="modify" type="xs:boolean" />
    <xs:attribute name="Selected" type="xs:boolean" />
    <xs:attribute name="robcolor" type="xs:string" />
    <xs:attribute name="rottextcolor" type="xs:string" />
    <xs:attribute name="fontsize" type="xs:integer" />
    <xs:attribute name="bold" type="xs:boolean" />
    <xs:attribute name="align" type="xs:string" />
    <xs:attribute name="xpos" type="xs:integer" />
    <xs:attribute name="ypos" type="xs:integer" />
    <xs:attribute name="Common" type="xs:boolean" />
    <xs:simpleType name="DataTypes">
      <xs:restriction base="xs:string">
        <xs:pattern value="Text|Integer|Float|Bool|Date" />
      </xs:restriction>
    </xs:simpleType>
    <xs:attribute name="ResultType" type="DataTypes" />
    <xs:simpleType name="ActionTypes">
      <xs:restriction base="xs:string">
        <xs:pattern
value="Delete|Duplicate|Next|Previous|First|Last|New|Initialize" />
      </xs:restriction>
    </xs:simpleType>
    <xs:attribute name="FormActionType" type="ActionTypes" />
  </xs:schema>

```

Adesso Plug-Ins: Development Guide

```
<!-- definition of complex elements -->
<xs:element name="Globals">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="USERNAME" />
      <xs:element ref="USERID" />
      <xs:element ref="LOCATION" />
      <xs:element ref="RADIUS" />
      <xs:element ref="TOTALRECORDS" minOccurs="0" maxOccurs="1" />
      <xs:element ref="CURRENTRECORDINDEX" minOccurs="0" maxOccurs="1" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="Param">
  <xs:complexType>
    <xs:attribute ref="Name" use="required" />
    <xs:attribute ref="Type" use="required" />
    <xs:attribute ref="Value" use="required" />
  </xs:complexType>
</xs:element>
<xs:element name="Params">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Param" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="li">
  <xs:complexType mixed="true">
    <xs:attribute ref="Name" use="required" />
    <xs:attribute ref="DisplayName" use="required" />
    <xs:attribute ref="Selected" use="optional" />
  </xs:complexType>
</xs:element>
<xs:element name="List">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="li" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
    <xs:attribute ref="multiselect" use="required" />
    <xs:attribute ref="modify" use="optional" />
  </xs:complexType>
</xs:element>
<xs:element name="LabelPos">
  <xs:complexType>
    <xs:attribute ref="xpos" use="required" />
    <xs:attribute ref="ypos" use="required" />
    <xs:attribute ref="width" use="required" />
    <xs:attribute ref="height" use="required" />
  </xs:complexType>
</xs:element>
<xs:element name="FieldPos">
  <xs:complexType>
    <xs:attribute ref="xpos" use="required" />
    <xs:attribute ref="ypos" use="required" />
    <xs:attribute ref="width" use="required" />
    <xs:attribute ref="height" use="required" />
  </xs:complexType>
</xs:element>
```

```

<xs:element name="Label">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="LabelPos" maxOccurs="1" />
    </xs:sequence>
    <xs:attribute ref="Name" use="required" />
    <xs:attribute ref="ID" use="required" />
    <xs:attribute ref="bgcolor" use="required" />
    <xs:attribute ref="textcolor" use="required" />
    <xs:attribute ref="align" use="required" />
    <xs:attribute ref="fontsize" use="required" />
    <xs:attribute ref="bold" use="required" />
  </xs:complexType>
</xs:element>
<xs:element name="FormField">
  <xs:complexType mixed="true">
    <xs:sequence>
      <xs:element ref="List" minOccurs="0" maxOccurs="1" />
      <xs:element ref="FieldPos" maxOccurs="1" />
      <xs:element ref="Label" maxOccurs="1" />
    </xs:sequence>
    <xs:attribute ref="Name" use="required" />
    <xs:attribute ref="TableName" use="required" />
    <xs:attribute ref="ID" use="required" />
    <xs:attribute ref="Type" use="required" />
    <xs:attribute ref="ControlType" use="required" />
    <xs:attribute ref="Enabled" use="required" />
    <xs:attribute ref="Visible" use="required" />
    <xs:attribute ref="bgcolor" use="required" />
    <xs:attribute ref="textcolor" use="required" />
    <xs:attribute ref="robgcolor" use="required" />
    <xs:attribute ref="rotextcolor" use="required" />
    <xs:attribute ref="fontsize" use="required" />
    <xs:attribute ref="bold" use="required" />
  </xs:complexType>
</xs:element>
<xs:element name="Tab">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="FormField" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
    <xs:attribute ref="Common" use="required" />
    <xs:attribute ref="Name" use="required" />
    <xs:attribute ref="ID" use="required" />
    <xs:attribute ref="Enabled" use="required" />
    <xs:attribute ref="Visible" use="required" />
    <xs:attribute ref="bgcolor" use="required" />
    <xs:attribute ref="textcolor" use="required" />
  </xs:complexType>
</xs:element>
<xs:element name="Field">
  <xs:complexType mixed="true">
    <xs:attribute ref="Name" use="required" />
    <xs:attribute ref="ID" use="required" />
    <xs:attribute ref="DataType" use="required" />
    <xs:attribute ref="TableName" use="required" />
    <xs:attribute ref="IsDirty" use="optional" />
    <xs:attribute ref="modify" use="optional" />
  </xs:complexType>
</xs:element>

```

Adesso Plug-Ins: Development Guide

```
<xs:element name="Form">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Tab" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
    <xs:attribute ref="Name" use="required" />
    <xs:attribute ref="GUID" use="required" />
    <xs:attribute ref="bgcolor" use="required" />
    <xs:attribute ref="textcolor" use="required" />
    <xs:attribute ref="width" use="required" />
    <xs:attribute ref="height" use="required" />
  </xs:complexType>
</xs:element>
<xs:element name="Table">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Field" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
    <xs:attribute ref="Name" use="required" />
    <xs:attribute ref="GUID" use="required" />
  </xs:complexType>
</xs:element>
<xs:element name="Result">
  <xs:complexType mixed="true">
    <xs:attribute ref="ResultType" use="required" />
  </xs:complexType>
</xs:element>
<xs:element name="Save">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="Y|Yes|YES|N|No|NO" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>
<xs:element name="error" />
<xs:element name="FormAction">
  <xs:complexType mixed="true">
    <xs:attribute ref="FormActionType" use="optional" />
  </xs:complexType>
</xs:element>
<xs:element name="FieldName">
  <xs:complexType mixed="true" />
</xs:element>
<xs:element name="FieldNames">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="FieldName" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="ParameterName">
  <xs:complexType mixed="true" />
</xs:element>
<xs:element name="ParameterNames">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="ParameterName" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Adesso Plug-Ins: Development Guide

```

<xs:element name="EventHandler">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Globals" minOccurs="0" maxOccurs="1" />
      <xs:element ref="Params" minOccurs="0" maxOccurs="1" />
      <xs:element ref="Form" minOccurs="0" maxOccurs="1" />
      <xs:element ref="Table" minOccurs="0" maxOccurs="1" />
      <xs:element ref="Result" minOccurs="0" maxOccurs="1" />
      <xs:element ref="Save" minOccurs="0" maxOccurs="1" />
      <xs:element ref="FormAction" minOccurs="0" maxOccurs="unbounded" />
      <xs:element ref="error" minOccurs="0" maxOccurs="1" />
      <xs:element ref="FieldNames" minOccurs="0" maxOccurs="1" />
      <xs:element ref="ParameterNames" minOccurs="0" maxOccurs="1" />
    </xs:sequence>
    <xs:attribute ref="Type" use="required" />
    <xs:attribute ref="Name" use="required" />
    <xs:attribute ref="Assembly" use="required" />
    <xs:attribute ref="Class" use="required" />
    <xs:attribute ref="GUID" use="required" />
  </xs:complexType>
</xs:element>
</xs:schema>
<EventHandler Type="Expression|EventsNOUI*" Name="Asset Management DEMO"
Assembly="C:\DOCUME~1\mpalone\LOCALS~1\Temp\My Adesso Plug-Ins\{47678850-79D4-11D9-
4509-00160FBA767D}\AdessoEventHandler_9e076796829db2a99027173c0cc6cd8a.dll"
Class="Adesso.Client.TestEventHandler" GUID="{47678850-79D4-11D9-4509-
00160FBA767D}">
  <Globals>
    <USERNAME>Mike Palone</USERNAME>
    <USERID>mpalone@adessosystems.com</USERID>
    <LOCATION />
    <RADIUS>0</RADIUS>
  </Globals>
  <Params />
  <Form Name="Asset" GUID="{4BC7A366-9C25-4F32-89BD-5B96DF8F9B90}"
bgcolor="#00ffffd7" textcolor="#00804000" width="562" height="563">
    <Tab Common="1" Name="" ID="1" Enabled="1" Visible="1" bgcolor="#00ffffd7"
textcolor="#00804000">
      <FormField Name="Bar Code" ID="2" Type="Field" ControlType="Textbox"
TableName="Assets" Enabled="1" Visible="1" bgcolor="#00ffebd7"
textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
fontsize="10" bold="0">
        <FieldPos xpos="82" ypos="17" width="144" height="17" />
        <Label Name="Bar Code:" ID="3" bgcolor="#00ffffd7"
textcolor="#00804000" align="left" fontsize="10" bold="0">
          <LabelPos xpos="1" ypos="17" width="75" height="17" />
        </Label>
        445443
      </FormField>
      <FormField Name="Employee ID" ID="4" Type="Field" ControlType="Textbox"
TableName="Assets" Enabled="1" Visible="1" bgcolor="#00ffebd7"
textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
fontsize="10" bold="0">
        <FieldPos xpos="82" ypos="34" width="144" height="17" />
        <Label Name="Owner:" ID="5" bgcolor="#00ffffd7" textcolor="#00804000"
align="left" fontsize="10" bold="0">
          <LabelPos xpos="1" ypos="34" width="75" height="17" />
        </Label>
        1364
      </FormField>

```

Adesso Plug-Ins: Development Guide

```
<FormField Name="Name" ID="6" Type="OneToOne" ControlType="Textbox"
  TableName="Employee" Enabled="0" Visible="1" bgcolor="#00ffebd7"
  textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
  fontsize="10" bold="0">
  <FieldPos xpos="82" ypos="51" width="144" height="17" />
  <Label Name="Name:" ID="7" bgcolor="#00ffffd7" textcolor="#00804000"
    align="left" fontsize="10" bold="0">
    <LabelPos xpos="1" ypos="51" width="75" height="17" />
  </Label>
  Megan Randall
</FormField>
</Tab>
<Tab Common="0" Name="General" ID="8" Enabled="1" Visible="1"
  bgcolor="#00ffffd7" textcolor="#00804000">
  <FormField Name="Type" ID="9" Type="Field" ControlType="List"
    TableName="Assets" Enabled="1" Visible="1" bgcolor="#00ffebd7"
    textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
    fontsize="10" bold="0">
    <List multiselect="0">
      <li Name="" DisplayName="" />
      <li Name="Desktop PC" DisplayName="Desktop PC" />
      <li Name="Laptop" DisplayName="Laptop" />
      <li Name="Tablet PC" DisplayName="Tablet PC" />
      <li Name="Pocket PC" DisplayName="Pocket PC" />
      <li Name="Server" DisplayName="Server" />
      <li Name="Cell Phone" DisplayName="Cell Phone" />
    </List>
    <FieldPos xpos="82" ypos="0" width="144" height="17" />
    <Label Name="Type:" ID="10" bgcolor="#00ffffd7" textcolor="#00804000"
      align="left" fontsize="10" bold="0">
      <LabelPos xpos="1" ypos="0" width="75" height="17" />
    </Label>
    Laptop
  </FormField>
  <FormField Name="Vendor" ID="11" Type="Field" ControlType="List"
    TableName="Assets" Enabled="1" Visible="1" bgcolor="#00ffebd7"
    textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
    fontsize="10" bold="0">
    <List multiselect="0">
      <li Name="" DisplayName="" />
      <li Name="Compaq" DisplayName="Compaq" />
      <li Name="Dell" DisplayName="Dell" />
      <li Name="Fujitsu" DisplayName="Fujitsu" />
      <li Name="Nokia" DisplayName="Nokia" />
      <li Name="Toshiba" DisplayName="Toshiba" />
      <li Name="IBM" DisplayName="IBM" />
    </List>
    <FieldPos xpos="82" ypos="17" width="144" height="17" />
    <Label Name="Vendor:" ID="12" bgcolor="#00ffffd7" textcolor="#00804000"
      align="left" fontsize="10" bold="0">
      <LabelPos xpos="1" ypos="17" width="75" height="17" />
    </Label>
    Dell
  </FormField>
```

Adesso Plug-Ins: Development Guide

```
<FormField Name="Model" ID="13" Type="Field" ControlType="Textbox"
  TableName="Assets" Enabled="1" Visible="1" bgcolor="#00ffebd7"
  textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
  fontsize="10" bold="0">
  <FieldPos xpos="82" ypos="34" width="144" height="17" />
  <Label Name="Model:" ID="14" bgcolor="#00ffffd7" textcolor="#00804000"
    align="left" fontsize="10" bold="0">
    <LabelPos xpos="1" ypos="34" width="75" height="17" />
  </Label>
  D45
</FormField>
<FormField Name="Notes" ID="15" Type="Field" ControlType="Memo"
  TableName="Assets" Enabled="1" Visible="1" bgcolor="#00ffebd7"
  textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
  fontsize="10" bold="0">
  <FieldPos xpos="82" ypos="51" width="144" height="51" />
  <Label Name="Notes:" ID="16" bgcolor="#00ffffd7" textcolor="#00804000"
    align="left" fontsize="10" bold="0">
    <LabelPos xpos="1" ypos="51" width="75" height="17" />
  </Label>
  This is some text within my memo field. This is very very cool isn't it?
  Thanks. Fred
</FormField>
<FormField Name="FOO" ID="17" Type="Field" ControlType="Image"
  TableName="Assets" Enabled="1" Visible="1" bgcolor="#00ffebd7"
  textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
  fontsize="10" bold="0">
  <FieldPos xpos="82" ypos="102" width="144" height="85" />
  <Label Name="Photo:" ID="18" bgcolor="#00ffffd7" textcolor="#00804000"
    align="left" fontsize="10" bold="0">
    <LabelPos xpos="1" ypos="102" width="75" height="17" />
  </Label>
</FormField>
<FormField Name="Test" ID="19" Type="Field" ControlType="List"
  TableName="Assets" Enabled="1" Visible="1" bgcolor="#00ffffd7"
  textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
  fontsize="8" bold="0">
  <List multiselect="1">
    <li Name="Dell" DisplayName="Dell" Selected="0" />
    <li Name="Fujitsu" DisplayName="Fujitsu" Selected="0" />
    <li Name="IBM" DisplayName="IBM" Selected="0" />
    <li Name="Nokia" DisplayName="Nokia" Selected="0" />
    <li Name="Toshiba" DisplayName="Toshiba" Selected="0" />
  </List>
  <FieldPos xpos="82" ypos="187" width="144" height="17" />
  <Label Name="Test:" ID="20" bgcolor="#00ffffd7" textcolor="#00804000"
    align="left" fontsize="8" bold="0">
    <LabelPos xpos="1" ypos="187" width="75" height="17" />
  </Label>
</FormField>
</Tab>
```

Adesso Plug-Ins: Development Guide

```
<Tab Common="0" Name="Financial" ID="21" Enabled="1" Visible="1"
bgcolor="#00ffffd7" textcolor="#00804000">
  <FormField Name="Purchase Date" ID="22" Type="Field" ControlType="Date"
    TableName="Assets" Enabled="1" Visible="1" bgcolor="#00ffebd7"
    textcolor="#00000000" robgcolor="#00ffffd7" rotextcolor="#00804000"
    fontsize="10" bold="0">
    <FieldPos xpos="82" ypos="0" width="144" height="17" />
    <Label Name="Purchase Date:" ID="23" bgcolor="#00ffffd7"
      textcolor="#00804000" align="left" fontsize="10" bold="0">
      <LabelPos xpos="1" ypos="0" width="75" height="17" />
    </Label>
    03/22/05
  </FormField>
  <FormField Name="Purchase Price" ID="24" Type="Field"
    ControlType="Textbox" TableName="Assets" Enabled="1" Visible="1"
    bgcolor="#00ffebd7" textcolor="#00804000" robgcolor="#00ffffd7"
    rotextcolor="#00804000" fontsize="10" bold="0">
    <FieldPos xpos="82" ypos="17" width="144" height="17" />
    <Label Name="Purchase Price:" ID="25" bgcolor="#00ffffd7"
      textcolor="#00804000" align="left" fontsize="10" bold="0">
      <LabelPos xpos="1" ypos="17" width="75" height="17" />
    </Label>
    $ 687.00
  </FormField>
  <FormField Name="Depreciation/Year" ID="26" Type="Field"
    ControlType="Textbox" TableName="Assets" Enabled="1" Visible="1"
    bgcolor="#00ffebd7" textcolor="#00804000" robgcolor="#00ffffd7"
    rotextcolor="#00804000" fontsize="10" bold="0">
    <FieldPos xpos="82" ypos="34" width="144" height="17" />
    <Label Name="Depreciation/Yr:" ID="27" bgcolor="#00ffffd7"
      textcolor="#00804000" align="left" fontsize="10" bold="0">
      <LabelPos xpos="1" ypos="34" width="75" height="17" />
    </Label>
    $ 250.00
  </FormField>
  <FormField Name="Current Value" ID="28" Type="Field"
    ControlType="Expression" TableName="Assets" Enabled="1" Visible="1"
    bgcolor="#00ffebd7" textcolor="#00804000" robgcolor="#00ffffd7"
    rotextcolor="#00804000" fontsize="10" bold="0">
    <FieldPos xpos="82" ypos="51" width="144" height="17" />
    <Label Name="Current Value:" ID="29" bgcolor="#00ffffd7"
      textcolor="#00804000" align="left" fontsize="10" bold="0">
      <LabelPos xpos="1" ypos="51" width="75" height="17" />
    </Label>
    $ 687.00
  </FormField>
</Tab>
<Tab Common="0" Name="Owner" ID="30" Enabled="1" Visible="1"
bgcolor="#00ffffd7" textcolor="#00804000">
  <FormField Name="Title" ID="31" Type="OneToOne" ControlType="Textbox"
    TableName="Employee" Enabled="0" Visible="1" bgcolor="#00ffebd7"
    textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
    fontsize="10" bold="0">
    <FieldPos xpos="82" ypos="0" width="144" height="17" />
    <Label Name="Title:" ID="32" bgcolor="#00ffffd7" textcolor="#00804000"
      align="left" fontsize="10" bold="0">
      <LabelPos xpos="1" ypos="0" width="75" height="17" />
    </Label>
    Associate Consultant
  </FormField>
</Tab>
```

Adesso Plug-Ins: Development Guide

```
<FormField Name="Email" ID="33" Type="OneToOne" ControlType="Hyperlink"
  TableName="Employee" Enabled="0" Visible="1" bgcolor="#00ffffd7"
  textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
  fontsize="10" bold="0">
  <FieldPos xpos="82" ypos="17" width="144" height="17" />
  <Label Name="Email:" ID="34" bgcolor="#00ffffd7" textcolor="#00804000"
    align="left" fontsize="10" bold="0">
    <LabelPos xpos="1" ypos="17" width="75" height="17" />
  </Label>
  mrandall@abc.com
</FormField>
<FormField Name="Phone" ID="35" Type="OneToOne" ControlType="Textbox"
  TableName="Employee" Enabled="0" Visible="1" bgcolor="#00ffebd7"
  textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
  fontsize="10" bold="0">
<FieldPos xpos="82" ypos="34" width="144" height="17" />
  <Label Name="Phone:" ID="36" bgcolor="#00ffffd7" textcolor="#00804000"
    align="left" fontsize="10" bold="0">
    <LabelPos xpos="1" ypos="34" width="75" height="17" />
  </Label>
  617-321-2453
</FormField>
<FormField Name="Department" ID="37" Type="OneToOne" ControlType="List"
  TableName="Employee" Enabled="0" Visible="1" bgcolor="#00ffebd7"
  textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
  fontsize="10" bold="0">
  <List multiselect="0">
    <li Name=" " DisplayName=" " />
    <li Name="Sales" DisplayName="Sales" />
    <li Name="Marketing" DisplayName="Marketing" />
    <li Name="Consulting" DisplayName="Consulting" />
    <li Name="Development" DisplayName="Development" />
    <li Name="Customer Service" DisplayName="Customer Service" />
    <li Name="Human Resources" DisplayName="Human Resources" />
  </List>
  <FieldPos xpos="82" ypos="51" width="144" height="17" />
  <Label Name="Department:" ID="38" bgcolor="#00ffffd7"
    textcolor="#00804000" align="left" fontsize="10" bold="0">
    <LabelPos xpos="1" ypos="51" width="75" height="17" />
  </Label>
  Consulting
</FormField>
<FormField Name="Manager" ID="39" Type="OneToOne" ControlType="Textbox"
  TableName="Employee" Enabled="0" Visible="1" bgcolor="#00ffebd7"
  textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
  fontsize="10" bold="0">
  <FieldPos xpos="82" ypos="68" width="144" height="17" />
  <Label Name="Manager:" ID="40" bgcolor="#00ffffd7" textcolor="#00804000"
    align="left" fontsize="10" bold="0">
    <LabelPos xpos="1" ypos="68" width="75" height="17" />
  </Label>
  Grace Lowell
</FormField>
```

Adesso Plug-Ins: Development Guide

```

<FormField Name="AttTest" ID="41" Type="Field" ControlType="Attachment"
  TableName="Assets" Enabled="1" Visible="1" bgcolor="#00ffffd7"
  textcolor="#00804000" robgcolor="#00ffffd7" rotextcolor="#00804000"
  fontsize="8" bold="0">
  <FieldPos xpos="82" ypos="102" width="144" height="51" />
  <Label Name="AttTest:" ID="42" bgcolor="#00ffffd7" textcolor="#00804000"
    align="left" fontsize="8" bold="0">
    <LabelPos xpos="1" ypos="102" width="75" height="17" />
  </Label>
</FormField>
</Tab>
<Tab Common="0" Name="Maintenance" ID="43" Enabled="1" Visible="1"
  bgcolor="#00ffffd7" textcolor="#00804000" />
</Form>
<Table Name="Assets" GUID="{C2414789-1675-421D-9D4F-DA0EF9D5D380}">
  <Field Name="Bar Code" ID="45" DataType="Text"
    TableName="Assets">445443</Field>
  <Field Name="Employee ID" ID="46" DataType="Text"
    TableName="Assets">1364</Field>
  <Field Name="Model" ID="47" DataType="Text" TableName="Assets">D45</Field>
  <Field Name="Type" ID="48" DataType="Text" TableName="Assets">Laptop</Field>
  <Field Name="Purchase Date" ID="49" DataType="Date"
    TableName="Assets">03/22/05</Field>
  <Field Name="Notes" ID="50" DataType="Memo"
    TableName="Assets">This is some text within my memo field. This is very
    very cool isn't it? Thanks. Fred</Field>
  <Field Name="FOO" ID="51" DataType="Binary" TableName="Assets" />
  <Field Name="Purchase Price" ID="52" DataType="Float"
    TableName="Assets">$ 687.00</Field>
  <Field Name="Depreciation/Year" ID="53" DataType="Float"
    TableName="Assets">$ 250.00</Field>
  <Field Name="Age" ID="54" DataType="Integer" TableName="Assets">78</Field>
  <Field Name="Current Value" ID="55" DataType="Float" TableName="Assets">
    $ 687.00</Field>
  <Field Name="Signature" ID="56" DataType="Binary" TableName="Assets" />
  <Field Name="Vendor" ID="57" DataType="Text" TableName="Assets">Dell</Field>
  <Field Name="Test" ID="58" DataType="Text" TableName="Assets" />
  <Field Name="AttTest" ID="59" DataType="Binary" TableName="Assets" />
  <Field Name="Exec Test" ID="60" DataType="Text" TableName="Assets" />
  <Field Name="Employee ID" ID="61" DataType="Text"
    TableName="Employee">1364</Field>
  <Field Name="Name" ID="62" DataType="Text" TableName="Employee">Megan
    Randall</Field>
  <Field Name="Email" ID="63" DataType="Text"
    TableName="Employee">mrandall@abc.com</Field>
  <Field Name="Phone" ID="64" DataType="Text"
    TableName="Employee">617-321-2453</Field>
  <Field Name="Address" ID="65" DataType="Text" TableName="Employee">1000
    Summer Street</Field>
  <Field Name="City" ID="66" DataType="Text"
    TableName="Employee">Worthington</Field>
  <Field Name="State" ID="67" DataType="Text" TableName="Employee">MA</Field>
  <Field Name="Zip Code" ID="68" DataType="Text"
    TableName="Employee">01342</Field>
  <Field Name="Department" ID="69" DataType="Text"
    TableName="Employee">Consulting</Field>
  <Field Name="Manager" ID="70" DataType="Text" TableName="Employee">Grace
    Lowell</Field>
  <Field Name="Title" ID="71" DataType="Text" TableName="Employee">Associate
    Consultant</Field>

```

Adesso Plug-Ins: Development Guide

```
<Field Name="Date of Hire" ID="72" DataType="Date" TableName="Employee" />
<Field Name="Birthday" ID="73" DataType="Date" TableName="Employee" />
</Table>
</EventHandler>
</root>
```

3.3. Adesso Form Controls

3.3.1. Supported Form Controls

- Listbox
- Memo
- HyperLink
- Checkbox
- Date/Time
- Radio
- AutoIncrement
- Spinner
- Attachment
- Image
- OneToOne
- Expression
- File Link
- Button

3.3.2. Non-Supported Form Controls

- Audio
- Signature
- OneToMany
- Survey Answer

3.3.3. Supported Data Types

- Text
- Memo
- Integer
- Float
- Yes/No
- Date/Time
- Binary

3.4. XML Elements & Attributes

The following elements and attributes will be supported in the plug-in message. Note that all character strings are limited to a maximum of 255 characters for all fields except memo fields.

Items below that are underlined> can be modified by the Plug-In.

EventHandler

The main Plug-In XML node.

Sub-Elements:

Globals, Params, Form, Table, Result, Save, FormAction, error, FieldNames, ParameterNames

Attributes:

Assembly

Provides the qualified name associated with the registered Handler.

Class

Provides the name of the Class within the Assembly to execute

Type

Indicates to the called Handler the context under which it was called.

Accepted Values: Any combination of the following strings: "Expression", "Form Button", or "Events" separated by '|'. An asterisk ('*') is placed next to the type currently being used.

Name

Provides the current "user friendly" name of the application.

GUID

Provides the internal globally unique identifier assigned to this application.

Globals

Global constant values.

Sub-Elements:

USERNAME, USERID, LOCATION, RADIUS, TOTALRECORDS, CURRENTRECORDINDEX

Attributes:

N/A

USERNAME

User name of user currently logged into Adesso Client.

Sub-elements:

N/A

Attributes:

N/A

USERID

User ID of user currently logged into Adesso Client in the form of an email address.

Sub-elements:

N/A

Attributes:

N/A

LOCATION

Postal code (in the United States, this is the zip code).

Sub-elements:

N/A

Attributes:

N/A

RADIUS

The setting representing the greatest radius, in miles, over which the user wants to download binary files.

Sub-elements:

N/A

Attributes:

N/A

TOTALRECORDS

Number of records in the current view.

Sub-elements:

N/A

Attributes:

N/A

CURRENTRECORDINDEX

Zero based current record index of the record being processed within the view. Helps Alternate forms editors implementing Next, Previous, First, or Last FormActions.

Sub-elements:

N/A

Attributes:

N/A

Params

Additional Plug-In parameters defined within the Plug-In Designer or EXEC command declaration.

Sub-Elements:

Param

Attributes:

N/A

Param

Plug-In parameter defined within the Plug-In Designer or EXEC command declaration.

Sub-Elements:

N/A

Attributes:

Name

The name of the parameter.

Type

Text, Integer, Float, Yes/No, Date/Time, Binary

Value

The value associated with the parameter. May be blank.

Form

This tag contains all the Form information for the current record being processed. If the Plug-In Type is an Expression, no Form Data is sent

Sub-Elements:

Tab

Attributes:

Name

The name of the form as defined by the Adesso Designer.

GUID

Internal globally unique identifier that was assigned by the Adesso designer to this specific form.

bgcolor

The background color of the form.

textcolor

The text color used on the form.

width

The width of the form in pixels.

height

The height of the form in pixels.

Tab

This tag contains all the Tab information for a tab that exists on the Form being processed.

Sub Elements:

FormField

Attributes:

Common

Indicates that the fields present in this tab are common and visible from all tabs. Set to 1 if the tab is common. There may be only one common tab.

Accepted Values: "1" or "0"

Name

The name of the tab as defined by the Adesso Designer.

ID

Internally generated unique identifier for the element.

Enabled

Indicates if the tab is on or off. (Note: Has no effect currently)

Accepted Values: "1" or "0"

Visible

Indicates if the tab is visible.

Accepted Values: "1" or "0"

bgcolor

The background color of the tab.

Textcolor

The text color used on the tab.

FormField

This tag contains all the Form Field information within a tab that exists on the Form being processed. The FormField element's inner content is a string and is modifiable.

Sub-Elements:

List, FieldPos, Label

Attributes:

Name

The name of the form field as defined by the Adesso Designer.

TableName

The name of the table from which this field resides as defined by the Adesso Designer.

ID

Internally generated unique identifier for the element.

Enabled

Indicates if the form field is set to read only ("0")

Accepted Values: "1" or "0"

Visible

Indicates if the form field is visible. (Note: Has no effect currently)

Accepted Values: "1" or "0"

bgcolor

The background color of the form field.

textcolor

The text color used on the form field.

robicolor

The background color used on the form field when the field is set to read only.

rotextcolor

The text color used on the form field when the field is set to read only.

fontsize

The font size of the text that appears in the form field. Currently only sizes 8 or 10 are supported.

bold

Indicates if the text in the form field is bold ("1")

Accepted Values: "1" or "0"

List

This tag contains the List information within a Form Field that exists on the Form being processed.

Sub-Elements:

li

Attributes:

multiselect

The type of Adesso List as defined by the Adesso Designer. Accepts multiple entries displayed in a drop-down format.

Accepted Values: "1" or "0"

modify

Response signal. The list contents has been changed by the plug-in.

Accepted Values: "1" or "0"

li

This tag contains the List Item information within a List that exists on the Form being processed.

Sub Elements:

N/A

Attributes:

Name

The name for this List Item.

DisplayName

The display name for this List Item.

Selected

Indicates that this item is currently selected.

Accepted Values: "1" or "0"

FieldPos

This position of the Form Field on the Form being processed.

Sub Elements:

N/A

Attributes:

xpos

The horizontal position of the Form Field in number of pixels from an upper left hand 0, 0 origin on the form.

ypos

The vertical position of the Form Field in number of pixels from an upper left hand 0, 0 origin on the form.

Width

The width of the Form Field in pixels.

height

The height of the Form Field in pixels.

Label

This tag contains Label information within a Form Field that exists on the Form being processed.

Sub Elements:

LabelPos

Attributes:

Name

The text of the Label.

ID

Internally generated unique identifier for the element.

bgcolor

The background color of the Label.

textcolor

The text color used on the Label.

align

The alignment of the Label.

Accepted Values: "left", "center", or "right"

fontsize

The font size of the text that appears in the Label. Currently only sizes 8 or 10 are supported.

bold

Indicates if the text in the Label is bold ("1")

Accepted Values: "1" or "0"

LabelPos

This position of the Label on the Form being processed.

Sub Elements:

N/A

Attributes:

xpos

The horizontal position of the Label in number of pixels from an upper left hand 0, 0 origin on the form.

ypos

The vertical position of the Label in number of pixels from an upper left hand 0, 0 origin on the form.

width

The width of the Label in pixels.

height

The height of the Label in pixels.

Table

Contains all fields within the current Table being processed for the record.

SubElements:

Field

Attributes:

Name

The name of the Table as defined by the Adesso Designer.

GUID

Internal, globally unique identifier assigned by the Adesso Designer to the Table.

Field

Contains field information within the current Table being processed for the record. The field element's inner content is a string and is modifiable.

Sub Elements:

N/A

Attributes:

Name

The name of the Field in the table as defined by the Adesso Table Designer.

ID

Internally generated unique identifier for the element.

DataType

The Data Type of the Field in the table as defined by the Adesso Table Designer.

TableName

The name of the Table as defined by the Adesso Designer where this field resides.

IsDirty

Indicates that this Field's value has been changed since the form has been opened ("1" indicates a change has occurred. This attribute is applicable for applicable for Expression and Form Button type Plug-Ins.

Accepted Values: "1" or "0"

modify

Response signal for Attachment Binary DataTypes only. If set to "1" the existing attachment list will be deleted and refreshed with the list from the Plug-In.

Accepted Values: "1" or "0"

Result

Result tag responses are used for EXEC Expression results. The element's inner text is a string and is modifiable.

Sub Elements:

N/A

Attributes:

ResultType

The Data Type of the return value.

Accepted Values: "Text", "Integer", "Float", "Bool", or "Date"

Save

Save tag responses are used to indicate whether or not changes to the record should be saved persistently to the Adesso Database or not and are processed only when used in conjunction with Alternate Forms Editors. The element's inner text is a string, must be one of "Y", "Yes", "YES", "N", "No", or "NO", and is modifiable.

Sub Elements:

N/A

Attributes:

N/A

FormAction

FormAction tag responses are used to indicate navigational user actions that occur within the context of an Alternate Forms Editor. Alternate Form Editor Plug-Ins must be Modal and return control immediately after such user actions as Delete, Duplicate, Next Record, Previous Record, First Record, or Last Record. Plug-Ins will be called again immediately after the Plug-In Manager receives a FormActionType of Next, Previous, First, or Last.

Sub Elements:

N/A

Attributes:

FormActionType—The action type requested by the Plug-In.

Accepted Values: "Delete", "Duplicate", "Next", "Previous", "First", "Last", "New", or "Initialize".

Note that "New" is used only within the context of a request XML message from the Plug-In Manager to the Plug-In to indicate that this is a new record being processed. A "New" FormAction Type is not processed if it is present in the XML response.

Note that "Initialize" is used only within the context of a request XML message from the Plug-In Manager to the Plug-In to indicate that the record is initially being opened and the Adesso Form is being used. A "Initialize" FormAction Type is not processed if it is present in the XML response.

error

Error tag responses are used to indicate that error has occurred in the Plug-In and optionally may contain text that will be displayed by the Plug-In Manager. The element's inner error text is a string and is modifiable.

Sub Elements:

N/A

Attributes:

N/A

FieldNames

Contains information in response to the GetFieldNames interface call used for obtaining a list of names to be used when designing Field Name Mappings.

Sub Elements:

FieldName

Attributes:

N/A

FieldName

A name that the Plug-In wants included in the available list of names when designing [Field Name Mappings](#).

Sub-Elements:

N/A

Attributes:

N/A

ParameterNames

Contains information in response to the GetParameterNames interface call used for obtaining a list of names to be used when creating Additional Parameters in the Plug-In Designer.

Sub Elements:**ParameterName****Attributes:**

N/A

ParameterName

A name that the Plug-In wants included in the available list of names when creating [Additional Parameters](#) in the Plug-In Designer.

Sub Elements:

N/A

Attributes:

N/A

3.4.1. Color Definition

All color attribute values are in the form of HHRRGGBB where:

- "HH" is designated for Auto Color ("01" or Transparency ("FFFFFFF").
- "RR" is the Red portion of the RGB value.
- "GG" is the Green portion of the RGB value.
- "BB" is the Blue portion of the RGB value

Sample XML Fragment for Expression Return Value:

```
<Result ResultType="Text">Value1</Result>
```

Sample XML Response Fragment for Save Return Value used by Alternate Forms Editors only:

```
<!--Excepted values are ="Y|Yes|YES|N|No|NO" - >
<Save>YES</Save>
```

Sample XML Response Fragment for FormAction Value used by Alternate Forms Editors only (Note: New is used in Requests to indicate that the record is being newly created and is not processed if present in a response. Initialize is used in Requests to indicate that the record is initially being opened and the Adesso Form is being used and is not processed if present in a response.):

```
<!--Excepted values are "Delete|Duplicate|Next|Previous|First|Last|New|Initialize" -
->
<FormAction>Next</FormAction>
```

Sample XML Response Fragment for Error Return Value:

```
<Error>Error Text</Error>
```

4. Sample Plug-In Implementation

```
using SpeechLib;
using System;
using System.Xml;
using System.Windows.Forms;

namespace Adesso.Client
{
    /// <summary>
    /// Summary description for SpeechEventHandler.
    /// Note that this plug-in requires adding a dependency in the Plug-In Designer
    (Interop.SpeechLib.dll)
    /// </summary>
    public class SpeechEventHandler : IExtensionHandler
    {
        SpVoice _voice = new SpVoice();

        string IExtensionHandler.Handler(XmlDocument xmlDoc)
        {
            try
            {
                XmlNode node;

                node = xmlDoc.SelectSingleNode(@"//FormField[@Name='Rate']");

                SetSpeechRate(node.InnerText);

                node = xmlDoc.SelectSingleNode(@"//FormField[@Name='Text']");

                SpeakText(node.InnerText);

                // Since we're not sending anything back, return empty string
                return "";
            }
            catch (Exception ex)
            {
                MessageBox.Show("SpeechEventHandler: " + ex.Message.ToString());
            }

            return "<error/>";
        }

        private void SpeakText(String text)
        {
            _voice.Speak(text, SpeechVoiceSpeakFlags.SVSFDefault);
        }

        private void SetSpeechRate(String rate)
        {
            _voice.Rate = int.Parse(rate);
        }
    }
}
```

Adesso Plug-Ins: Development Guide

```
// NOTE: If you not want to return values for GetParameterNames or
GetFieldNames, return xmlDoc.DocumentElement.OuterXml or a blank string (eg. return
"");)
string IExtensionHandler.GetParameterNames(XmlDocument xmlDoc)
{
    return xmlDoc.DocumentElement.OuterXml;
}

string IExtensionHandler.GetFieldNames(XmlDocument xmlDoc)
{
    try
    {
        do
        {
            XmlNode node = null;
            XmlElement elem;

            XmlNode root = xmlDoc.DocumentElement;
            if (!root.HasChildNodes)
                break;

            XmlNode eventNode = root.LastChild;

            // Get to the FieldNames Node.
            if (eventNode.HasChildNodes)
            {
                for (int i=0; i<eventNode.ChildNodes.Count; i++)
                {
                    node = eventNode.ChildNodes[i];
                    if (node.Name == "FieldNames")
                    {
                        // Replace entries.
                        node.RemoveAll();

                        // Create a few new nodes.
                        elem = xmlDoc.CreateElement("FieldName");
                        elem.InnerText = "Rate";

                        // Add the node to the document.
                        node.AppendChild(elem);

                        elem = xmlDoc.CreateElement("FieldName");
                        elem.InnerText = "Speech";

                        // Add the node to the document.
                        node.AppendChild(elem);

                        break; // !Done
                    }
                }
            }
        }
        while (false);

        // Return results.
        return xmlDoc.DocumentElement.OuterXml;
    }
    catch(Exception ex)
    {

```

Adesso Plug-Ins: Development Guide

```
string error;
error = "SpeechEventHandler GetFieldNames: " + ex.Message.ToString();
System.Diagnostics.Debug.WriteLine("SpeechEventHandler GetFieldNames: " +
ex.Message.ToString());
    MessageBox.Show (error, "SpeechEventHandler");
}

// Do nothing
return "<error/>";
}
}
}
```

5. Metadata Extractor Plug-Ins

Metadata extractor plug-ins obtain data from the Params section of the XML. The parameters sent for the metadata extractor are:

- InputFile
- Attributes

With Attributes being an optional parameter.

When registering a metadata extractor plug-in, the file extension(s) for the files that are the target of the extractor must be entered on the [Other tab of the Plug-In Designer](#).

5.1. Sample Metadata Extractor Implementation

```

if (node.Name=="Params")
{
    IEnumerator ienum = node.GetEnumerator();
    if (node.HasChildNodes)
    {
        while (ienum.MoveNext ())
        {
            paramNode = (XmlNode) ienum.Current;
            attributeCollection = paramNode.Attributes;
            attribute = attributeCollection["Name"];
            paramName = attribute.Value;
            if (paramName == "InputFile")
            {
                attribute = attributeCollection["Value"];
                paramValue = attribute.Value;
                string display = "InputFile: " + paramValue;
                MessageBox.Show (display, "CrackerTest1EventHandler");
            }
            if (paramName == "Attributes")
            {
                attribute = attributeCollection["Value"];
                paramValue = attribute.Value;
                string display = "Attributes: " + "CrackerTest1EventHandler");
            }
        }
    }
}

```

6. Debug Support

Registry settings are available for:

- If the registry setting \HKEY_CURRENT_USER\Software\Adesso Systems\PlugIns\Debug is set to "1", log everything to a Log file (AdessoPlugInLog.txt) for 3rd party remote debugging.
- If the registry \HKEY_CURRENT_USER\Software\Adesso Systems\PlugIns\NoClearSHIMFiles is set to "1", XML request and response files will NOT be cleared after use.

The location of these files are typically in the system Temp directory. (eg. C:\Documents and Settings\\Local Settings\Temp)

To debug a plug-in, make sure you both the Adesso Client and Visual Studio available, and then follow these steps:

1. Launch the plug-in once from the Adesso Client.
2. Switch back to the Visual Studio 2003 IDE where you are developing the plug-in, and then select **Debug | Processes**.
3. Attach to the **AdessoShimFullFramework** process.
4. Set the breakpoints.
5. Re-launch the plug-in.
At your breakpoints, you will be returned to the debugger.

7. Building Plug-Ins

When you are setting up your working environment for creating a plug-in, note the following:

- You must make sure that you set the default namespace to Adesso.Client. (Working in Visual Studio 2003, in the Solution Explorer, right-click "ProjectName"; go to "Properties" and set the Default Namespace to Adesso.Client). This ensures that all Web references will compile correctly.
- If you have a problem registering VB plug-ins in Adesso, clear the "Root namespace" field. (Working in Visual Studio 2003, in the Solution Explorer, right-click "ProjectName"; go to "Properties")

When building a .NET plug-in assembly, add a reference to the AdessoInterfaces.dll file.

8. Security

- XML Request and Response file will be cleared after each synchronous operation with the plug-in unless the NoClearSHIMFiles registry setting is set to 1.
- All .NET Assemblies should be strong named signed Assemblies.

9. Features Not Yet Supported

- One-to-Many fields
- One-to-One fields for New Record Creation—List of possible records from related table(s) not provided in XML outbound. Existing record field setting from related table is provided outbound. Changes to inbound related table record field values cannot be made since they are read-only. XML indicates these are ENABLED=0.
- View specific information—Only one record processed by Plug-In Manager at a time
- Audio and Signature binary field processing—outbound and inbound.
- File Link binary processing—inbound
- Form and Button Background Image Data—outbound and inbound
- Adding or Deleting Table Fields
- Hooks into the Adesso Expression Engine
- Existing record Attachment and Image deletes. New record creation inbound attachment and image processing is supported.

10. Plug-In Implementation in the Adesso Client

Once the design of a plug-in is complete, the plug-in must be registered with each Adesso application that it is intended to work with. Registering the plug-in is a multi-step process which involves identifying the plug-in in the Adesso Plug-In Designer and also associating the plug-in with a specific design element (for example, the view from which a new record event will cause the plug-in to run).

This section documents the Adesso interface elements used to register the plug-in and associate it with the application design.

Notes:

- If you are a developer who does not usually work with the Adesso client, but you will be responsible for registering the plug-in, make sure that the Application Manager gives you the permission necessary to change the design of the application (set on the Adesso server).
- Registering the plug-in is not supported on the Pocket PC, you must use a desktop or laptop machine.
- Step-by-step instructions for registering an Adesso plug-in, as well as for creating other required design elements to support plug-ins are available in Adesso Client Online Help and the Adesso Users Guide for Release 3.0

10.1. Ensuring Plug-In/Client Compatibility

Plug-ins can only be used by Adesso Clients running release 3.0 or higher. To prevent Client devices running an earlier version of the software from downloading and using an application that employs a plug-in, the Application Manager may want to specify a minimum version of the Client that can download the application.

To enforce the release 3.0 minimum on Adesso Clients intended to use the application, the "Require minimum client version" field (see figure below) should be set to 3.0 before the design changes that include the plug-in registration are uploaded to the server.

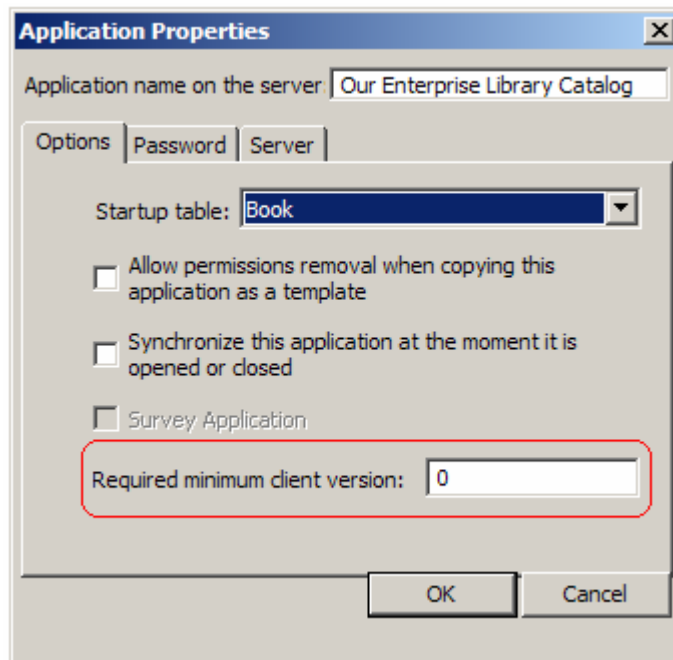


Figure 2: Application Properties Dialog—Setting a Minimum Client Version

10.2. Plug-In Designer

The Plug-In Designer is where an Adesso user registers the .NET plug-in.

The Plug-In Designer is accessed by selecting Plug-Ins from the Application Designer and then clicking the New button.

10.2.1.Plug-In Tab

Users create a new plug-in or open a selected plug-in, they will be presented with the Plug-In Designer dialog:

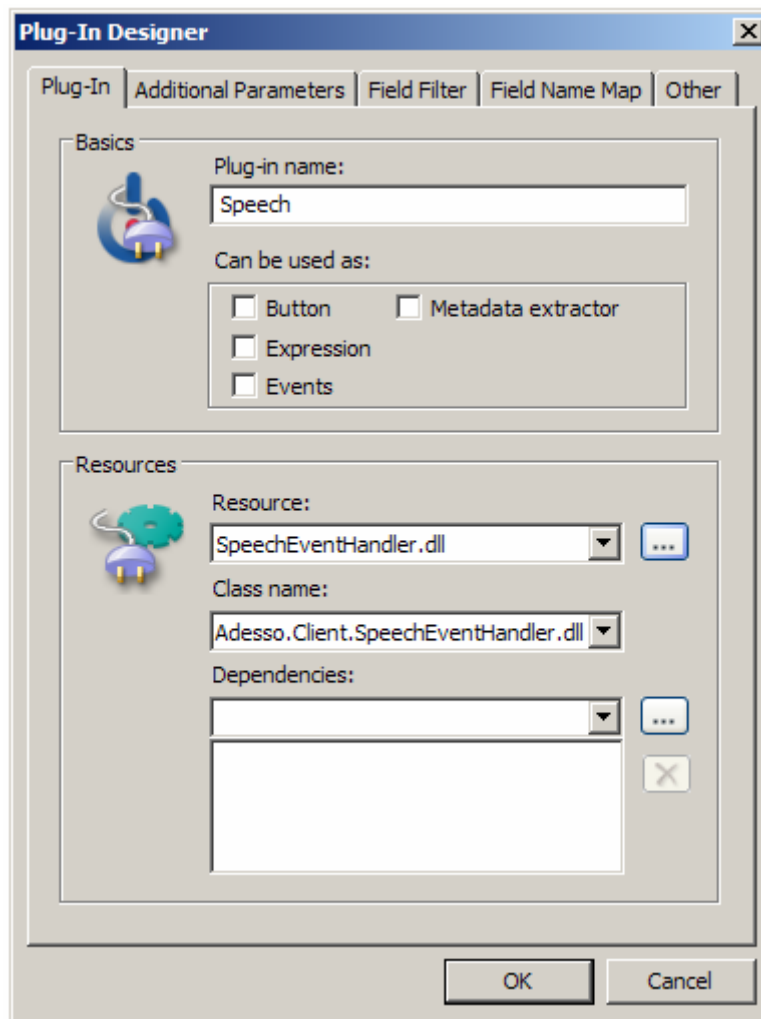


Figure 3: Plug-In Designer Plug-In Tab

10.2.1.1. Data Entry for the Plug-In Tab

- **Plug-in name**—The "user friendly" display name that may be changed at any time.
- **Can be used as**—A list of supported Types for a plug-in. At least one checkbox must be selected if the plug-in is to be saved without warning. If "Metadata extractor" is selected, typically, it is the only selection.
- **File**—The File Path of the plug-in Assembly on Disk. The Designer can choose from existing Assembly Resources or click/tap the file selection button to open a new/updated plug-In Assembly from disk.
- **Class Name**—The namespace used within the implementation of the plug-in Assembly. It is recommended that Designers qualify the namespace name with their company name. The Adesso Plug-In Manager will look for namespaces that start with "Adesso.Client", so in the implementation of the plug-in, 3rd party class definitions would look like the one on the following page. Note that this field is automatically filled in for you after selecting the Plug-In Assembly File from the File Selection Dialog.

```
namespace Adesso.Client.Acme
{
    class WebServiceEventHandler : IExtensionHandler
    {
```

The Class Name entry in the Adesso Extension Designer should therefore be "Acme.WebServiceEventHandler". "Adesso.Client" may or may not be pre-pended. The Plug-In Manager will insert that if not entered.

- **Dependencies** – A list control containing additional resource files that the plug-in requires to function correctly. The Designer can add or delete from the list using the buttons to the right of the list.

10.2.2. Additional Parameters

The Additional Parameters Tab displays the list of User Defined Parameters that the Designer wishes to include in the <Params> section of the XML ([Described Previously](#)). The Designer can create new, edit existing, or delete entries from the list. Names cannot include commas (,), periods (.), brackets ([]s), or pipes (|). Values cannot include commas (,), brackets ([]s), or pipes (|).

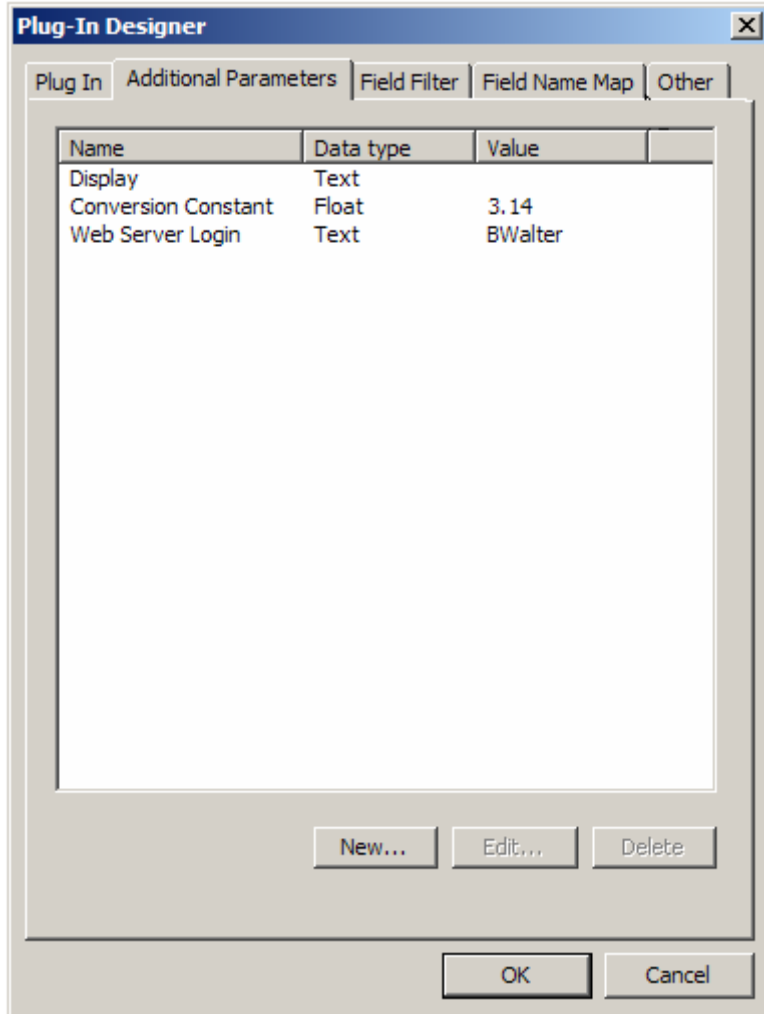


Figure 4: Plug-In Designer—Additional Parameters Tab

10.2.2.1. Plug-In Parameters Data Entry

This dialog is displayed when the Designer clicks the New or Modify buttons from the Additional Parameters Tab. The user must enter a Name for the Parameter, select a data type, and specify the value to be entered.

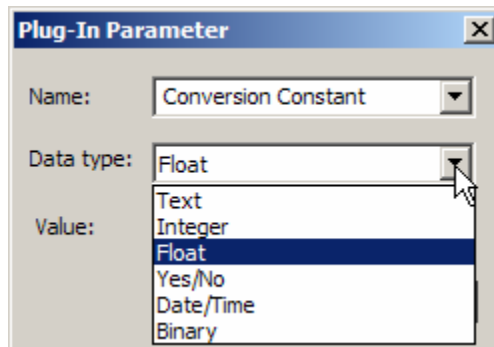


Figure 5: Plug-In Parameter Dialog--Selecting a Data Type

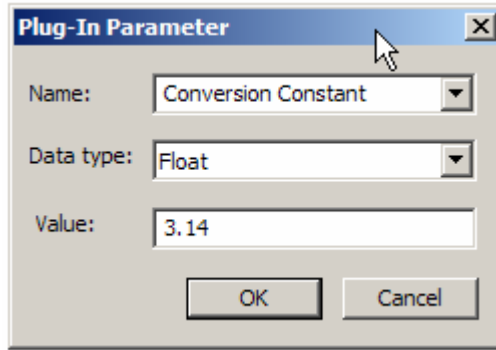


Figure 6 : Plug-In Parameter Dialog—Entry Complete

10.2.3.Field Filter Tab

The Field Filter Tab displays the list of available Fields from a selected Table (Available in the drop-down list in the bottom left of the dialog) and allows the Designer to add or delete 1 or more of them into the list of fields that will be included in the XML request sent to the plug-in Assembly. If no items are moved from the Available Fields list to Selected Fields list, then all fields will be sent.

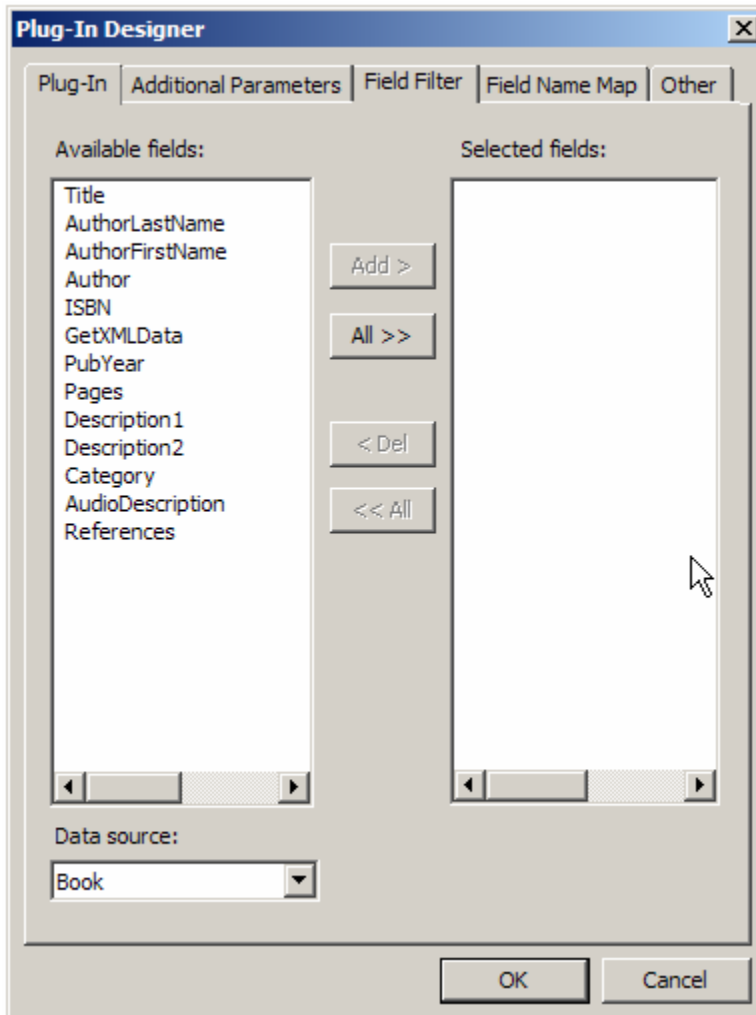


Figure 7: Plug-In Designer—Field Filter Tab

10.2.5. Other Tab

The Other tab is used to set, in seconds, the length of time that Adesso will attempt to run the plug-in before timing out. The default value is 0.

If the plug-in is a metadata extractor, this tab is also used to identify the file types from which the plug-in can extract data.

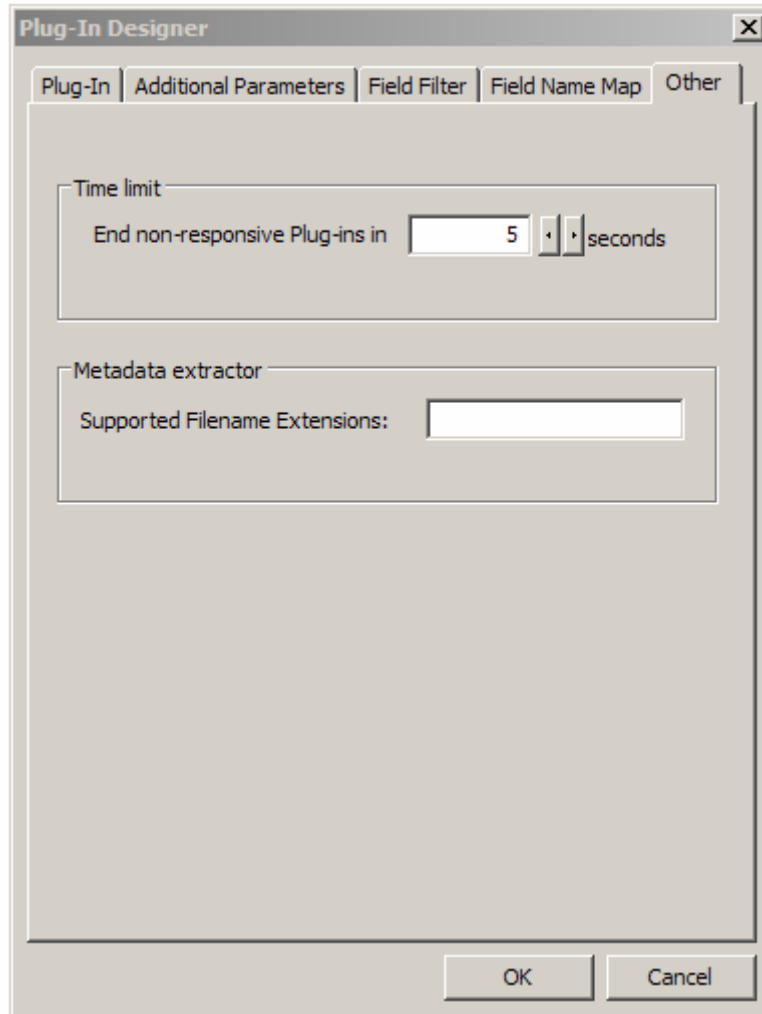


Figure 9: Plug-In Designer—Other Tab

10.2.5.1. Data Entry for the Other Tab

- **End non-responsive Plug-ins in...seconds** – The number of seconds that Adesso will attempt to run the plug-in before timing out.
- **Supported Filename Extension**—For metadata extractors only, a list of the extensions of supported file types. Extensions should be entered as alpha/numeric characters only, without a preceding period. Multiple entries are separated by commas.

An example entry: bmp, gif.

10.2.5.2. Associating Plug-Ins with Adesso Design Elements

After a given plug-in has been registered, it may be associated with various Adesso Client constructs. You do this within the Adesso Designer in the following locations.

10.3. Form Designer

- Tools / Form Validation Condition... / Form Validation Dialog / New / Validation Dialog
 - In the Select function drop-down list the user is presented with a list of each matching Plug-In Type (Expression Types) from the list of Registered Plug-Ins (**Note** – Additional EXEC Expression parameters may be entered as a variable argument list):
 - EXEC(“Friendly Name for Plug-In”, OnTheFly Parameter List...) where OnTheFly Parameter List... could be “A”, “B”, “3.14” for example.
- Select Row Context Menu – There is an entry for “Add Plug-In Button”.
 - When selected, you are presented with a Properties Dialog allowing the selection of one of the available plug-ins of the appropriate type.

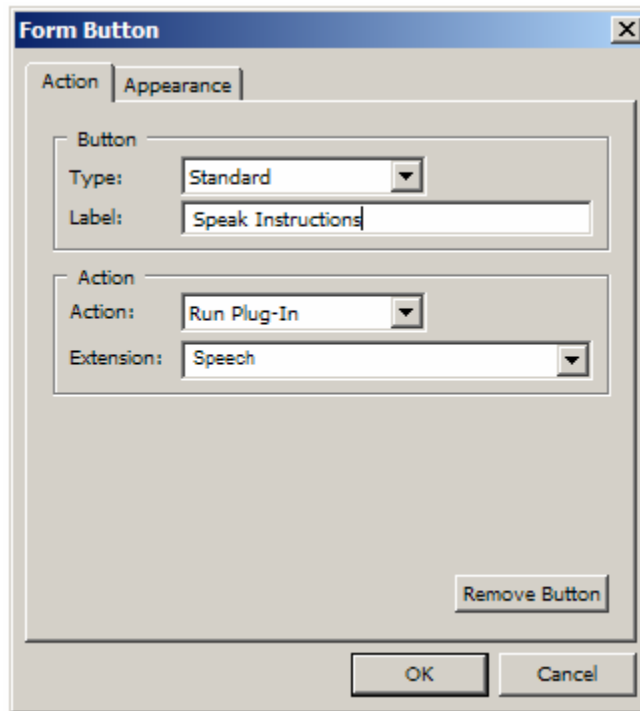


Figure 10: Form Control Dialog

10.4. Table Designer

- Field Properties/Display Control/Expression Control
 - In the Select function drop-down list additional functions are added for each matching plug-In Type (Expression Types) from the list of Registered Plug-Ins (**Note** – Additional EXEC Expression parameters may be entered as a variable argument list):
 - EXEC(“Friendly Name for Handler X” , OnTheFly Parameter List...) where OnTheFly Parameter List... could be “A”, “B”, “3.14” for example.

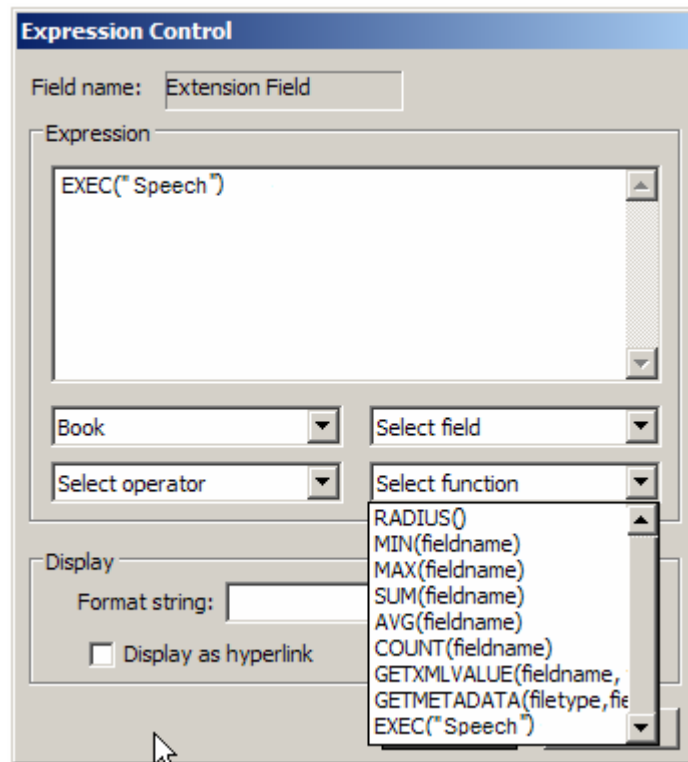


Figure 11: Expression Control Dialog

10.4.1.EXEC Function

Support for a new EXEC command is implemented in the Adesso Expression Language. The format of this command will take two forms:

- EXEC("User Plug-In Name")
- EXEC("User Plug-In Name", VariableArgument1, VariableArgument2, ...VariableArgumentN)

Where "User Plug-In Name" is the name associated with the plug-in in the System Object Table created during plug-in registration.

The variable argument list (or OnTheFly Parameters) allows for the entry of additional parameters to control operations with the Plug-In Manager's interactions with the plug-in. These arguments take the form of control flags or Plug-In Property values that must be passed through the system. They are listed in the <Params> Section of the XML.

10.5. View Designer

From the View Designer Other Tab you can associate plug-ins of Type "Events" with the ability to create new records or open existing records with either an Adesso Form or an alternate Form Editor.

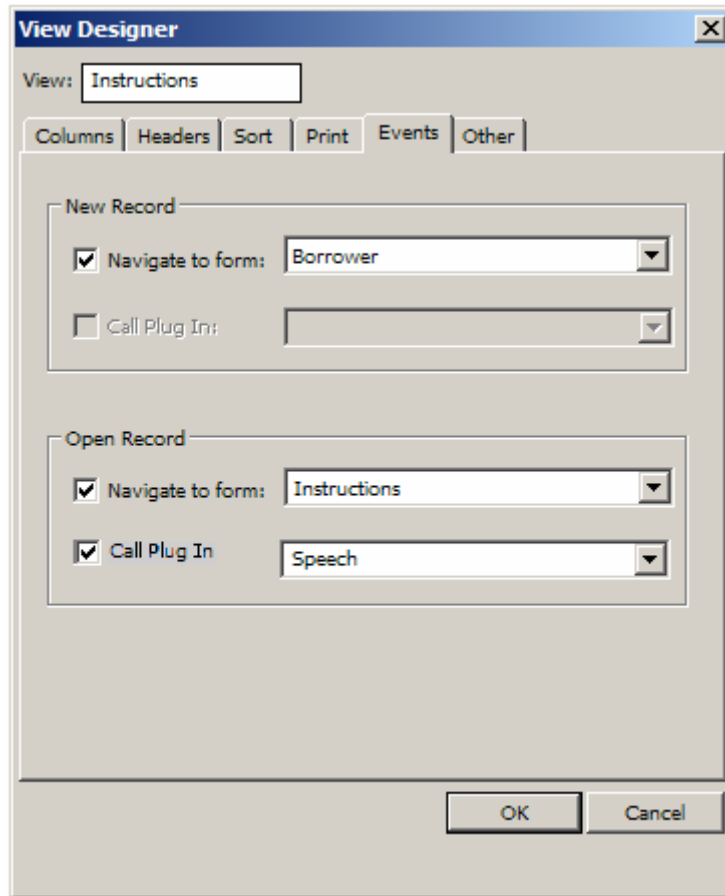


Figure 12: View Designer

Index

Adesso Client		LabelPos	28
plug-in compatibility	37	li	27
Color Definition	31	List	26
CURRENTRECORDINDEX	23	LOCATION	23
development working environment	36	metadata extractor	5
error	30	Param	24
EventHandler	22	ParameterName	31
Events	5	ParameterNames	31
New Record	5	Params	23
EXEC command	5	Plug-In Designer	38
Field	29	RADIUS	23
FieldName	31	registering a plug-in	37
FieldNames	30	Result	29
FieldPos	27	sample implementation	
Form	24	metadata extractor	35
form controls		plug-in	32
non-supported	21	Save	30
supported	21	Tab	25
FormAction	30	Table	29
FormField	26	TOTALRECORDS	23
GetFieldNames	7	USERID	22
GetParameterNames	7	USERNAME	22
Globals	22	XLM Attributes	22
Handler	7	XML definition	8
Handler Request	6	XML Elements	22
Handler Response	6	XML request	10
IExtensionHandler Interface	6	XML response	10
Label	28	XSD XML Schema	9