EWS Editor

An Exchange Mailbox Explorer built on the Exchange Web Services Managed API

# Overview

EWS Editor has three goals:

1. Demonstrate the [Exchange Web Services Managed API](http://msdn.microsoft.com/en-us/library/dd633709.aspx) functionality and simplicity to developers
2. Demonstrate the [Exchange Web Services](http://msdn.microsoft.com/en-us/library/bb204119(EXCHG.140).aspx) traffic used to perform actions initiated through an explorer user interface to aid any Exchange Web Services developer.
3. Assist non-developers in debugging and understanding Exchange stores by exploring items, folders, and their properties in depth.

# Contents

1. Release Notes
2. Feature Notes
3. Source Notes

# Release Notes

## 1.5 Release – 11/6/2009

The first public release of EWSEditor! The basic feature set is still be flushed out but it is still a very useful tool. There are three feature areas of EWSEditor: General UI, Support Tools, and API Coverage. The focus of this release was to establish the UI design and flow of the application, solidify some useful support tools, and illustrate the direction of the application.

There are some gaps in CRUD tasks across item and folder properties and items themselves. The major focus of upcoming releases will be to fill in these gaps before moving on to new features.

## 1.6.1 Release – 3/5/2010

API coverage is a major focus of the 1.6.\* releases of EWSEditor. The 1.6.1 release adds an OOF Settings and Availability form to this end. There were several bugs addressed in this release was well as some changes in the error dialog and about information. These changes will make the information displayed and logged by EWSEditor more useful.

# Features

### Exchange Service Profiles

Exchange Service Profiles describe not only the connection information for calls specified in the [ExchangeService](http://msdn.microsoft.com/en-us/library/microsoft.exchange.webservices.data.exchangeservice.aspx) class but also the root folder to use in the tree view of EWS Editor. Multiple ExchangeService objects can be displayed at one time using unique configurations. This allows folders on multiple servers and server versions to be displayed at one time or displaying the same folders using different credentials.

Exchange Service Profiles can be saved to an XML file which contains the configuration information (sans specified passwords) so that it can be reloaded easily. If a profile requires credentials then a prompt will appear when loading the Services Profile to enter the credentials.



### View properties of ExchangeServices, Folders, and Items

The PropertyDetailsGrid display first class properties for all nodes of the tree view starting with the root, ExchangeService node and all the way to the folder contents form, displaying first class properties for Items in a folder. Under the View menu the [PropertySet](http://msdn.microsoft.com/en-us/library/microsoft.exchange.webservices.data.propertyset.aspx) used to retrieve item or folder data can be configured to customize which schema properties and extended properties are displayed in the PropertyDetailsGrid.



**Extended Property Lookup**

A number of known extended MAPI properties have been indexed in EWS Editor. Extended properties can be added to the PropertyDetailsGrid through a look up using a known constant name for the property such as PR\_SUBJECT, PidTagSubject, etc. An Extended Property Lookup tool provides a way to quickly get the property information for a known constant.



### Request and response logging

While EWS Editor is running, all requests and responses are logged to a file. The Chatter Log tool is used to view the history of requests and responses.



# Source Notes

### Understanding the PropertyInformation Namespace

The PropertyInformation namespace’s main purpose is to standardize the textual representation of object values in EWS Editor. This is done through the PropertyInterpretation class which uses TypeValues, SmartViews, and KnownExtendedProperties to return text data describing a given object property.

TypeValues standardize the text to display for the value of specific object types. SmartViews apply Exchange/Outlook business logic to property values to translate the raw value into a more meaningful one. The KnownExtendedProperties class contains lookup methods and data related to documented extended properties.

TypeValues and SmartViews conform to a simple interface which associated the logic implemented in a specific TypeValue or SmartView with specific types (ITypeValue) or known properties (ISmartView). A TypeValueFinder class and SmartViewFinder class retrieve the appropriate class based on an input type or property. To add new ITypeValue or ISmartView implementations to the project simply implement the interface and add the new class to the TypeValueFinder or SmartViewFinder’s Initialize() method. Then anytime the PropertyDetailsGrid or other controls which already use the PropertyInterpretation class will automatically benefit from the translation logic for the type or property referenced.