Inetlab.MMS.MM7

.NET implementation of MM7 protocol for two-way MMS messaging

Table of Contents

Getting Started with MM7Client

MM7 is the interface between MMSC and a value-added service provider (VASP). The MM7 interface is used to send MMS from 3rd party providers (e.g., a bank sending a statement or an advertiser sending publicity).

Send MMS from SMIL file

```
_client = new MM7Client(mmscUrl);

_client.AuthId = authId;
_client.AuthSecret = authSecret;
_client.VASPID = vaspid;
_client.VASID = vasid;
_client.SenderAddress = senderAddress;
_client.ServiceCode = serviceCode;
```

- **url** - http or https URL of MMSC
- **authId** - username for Basic authentication on the remote MMS Relay/Server.
- **authSecret** - password for Basic authentication on the remote MMS Relay/Server.
- **vaspid** - your identifier as VASP (Value Added Service Provider)
- **vasid** - identifier of your application
- **senderAddress** - the address of the message originator.
- **serviceCode** - information supplied for billing purposes

```
SubmitRequest req = _client.CreateSubmitRequest();

req.Recipients.To.Add("+79171234567");

req.MessageClass = MessageClass.Personal;
req.DeliveryReport = true;
req.Subject = "Test";
req.Priority = Priorities.Normal;
req.ChargedParty = ChargedParty.Sender;
req.ContentHref = "/mms/mm7/mm7client";
req.TransactionID = DateTime.Now.Ticks.ToString();

try
{
    // Submit MMS Message as VASP
    SubmitResponse resp = await _client.Submit(smilFilePath, req);
    // Check response status
    if (resp.Status.StatusCode == 1000)
    {
        Console.WriteLine("MMS message has been sent successfully");

    }
}
catch (MM7Exception ex)
{
    Console.Error.WriteLine(ex.Message);
    Console.Error.WriteLine(ex.Status.Details);
}
catch (WebException ex)
{
    Console.Error.WriteLine(ex.Message);
}
```

Getting Started with MM7Server

Standalone MMS Server

The MMS messages from MMSC can be received with standalone MM7Server class. This class has several events that will raise as soon as appropriate MM7 message is received. In the event handlers you can save the MMS content or perform any other processing. You need to start the server with the method @Inetlab.MMS.MM7Server.Start().

```
public class MMSServerForVASPExample
{
    private MM7Server _server;

    public MMSServerForVASPExample()
    {

        _server = new MM7Server("http://localhost:9090/");
        _server.ServerMode = MM7ServerMode.VASP;
        _server.AuthScheme = AuthenticationSchemes.Basic;
        _server.Events.OnAuthentication = OnAuthentication;
        _server.Events.OnDeliverRequest = OnDeliverRequest;
    }

    public void Start()
    {
        _server.Start();
    }

    private Task<string> OnAuthentication(AuthenticationEventArgs e)
    {
        //check username
        if (e.UserName == "test")
        {
            string password = "password from db";

            // return password
            return Task.FromResult(password);
        }

        // not authenticated.
        return Task.FromResult<string>(null);
    }


    /// <summary> Raises when MMSC delivers MMS to VASP endpoint.</summary>
    private Task<DeliverResponse> OnDeliverRequest(DeliverReqEventArgs e)
    {
        //MMS received from MMSC on VASP MMS Endpoint.

        // Save all MMS message parts to the Delivered folder
        string path = Path.GetFullPath("Delivered");
        path = Path.Combine(path, e.Request.TransactionID);
        Directory.CreateDirectory(path);

        foreach (MMSPart part in e.Message.Parts)
        {
            part.Save(Path.Combine(path, part.GetFileName()));
        }

        return Task.FromResult(new DeliverResponse(e.Request));
    }
}
```

MMS Server in ASP.NET Core

It is possible to implement ASP.NET Core middleware with MM7Server class. See the *MM7Server.AspNetCore* sample application in the Samples folder.

Changelog

[1.0.0] - 2022-02-16

Added

- .NET Standard 2.0, .NET Core version
- add IHttpContext interface to be able to wrap any HttpContext from HttpListner, ASP.NET or ASP.NET Core
- MM7Client. Added support for web proxy
- added ASP.NET Core sample

Changed

- Simplified ILog interface
- MM7Client implemented with HttpClient and all send methods are Task-based.

Fixed

- System.ArgumentException: Item has already been added. Key in dictionary: 'binary'. when creating MediaMessage

[0.10.0] - 2018-04-20

Added

- added two protected methods GetWebRequest and GetWebResponse.

[0.9.2] - 2018-01-02

Fixed

- MM7Client should process SOAP Fault instead throwing of "Unknown server response".

Version 0.9.1 19.11.2016 -small fixes in demo application

Version 0.9.0 11.03.2016

- added support of Basic and Digest authentication in MM7Client and MM7Server classes.
- added ability to change ServerName in MM7Server.

Version 0.8.2 07.03.2016

- add ability to get request parameters for the event handlers in standalone MM7Server

Version 0.8.1 06.03.2016

- new logging classes
- improved MIME classes
- added ability to use Evaluation version as Full version after adding a license file to the project as embedded resource. You can update nuget package of the library in your project and use always latest version.

Version 0.7.2 06.06.2014

- fixed missing MM7Version in DeliverRequest

Version 0.7.1 30.10.2013

- added workaround for non-standard error responses

Version 0.7.0 27.08.2012

- added ability to send DRM content

Version 0.6.3 22.08.2012

- addition of DeliveryCondition, ApplicID, ReplyApplicID, AuxApplicInfo, DRMContent fields to SubmitRequest

Version 0.6.2 17.08.2012

- fix threading issue in MIME classes

Version 0.6.1 15.08.2012

- adapt XML serialization to schema 5.3.0

Version 0.6.0 27.07.2012

- added support of DeliveryReport and Read-Reply report

Version 0.5.12 31.01.2012

- added ability to send SOAPAction HTTP header
- process server responses without Content-Length HTTP header

Version 0.5.11 02.11.2011

- handle GET requests to MMS server

Version 0.5.10 06.09.2011

- improved parsing HTTP headers

Version 0.5.9 16.05.2011

- added methods that helps to send custom DeliverRequest.

Version 0.5.8 06.12.2010

- fixed charset of transfered text file within SMIL message

Version 0.5.7 01.11.2010

- added ability to change MM7Version and MM7 Schema version
- added ContentClass element for SubmitReq

Version 0.5.6 06.09.2010

- added Submit method for sending MMS message from SMIL file

Version 0.5.5 25.08.2010

- adapted to some vendor specific responses
- fixed basic authentication
- for class MM7Client added new method Submit with SubmitRequest parameter

Version 0.5.4 12.04.2010

- adapted to some vendor specific requests

Version 0.5.3 08.04.2010

- added trace of MMSServer

Version 0.5.2 10.09.2009

- fixed From was empty in MMSMessage

Version 0.5.1 08.09.2009

- fixed TimeStamp parsing

Version 0.5.0 26.08.2009

- added ability to process request in ASHX files

Version 0.4.6 24.08.2009

- fixed server.Stop() bug

Version 0.4.5 05.08.2009

- added ability to change Content-Transfer-Encoding for MMSParts

Version 0.4.4 14.06.2009

- improved MM7Client

Version 0.4.3 16.04.2009

- added support of multipart.mixed MMS messages
- added "Inetlab.MMS.MM7.Switch" to enable detailed information about requests and responses

Version 0.4.2 05.03.2009

- added Timeout property to MM7Client

Version 0.4.1 28.02.2009

- added xml declaration to request and response

Version 0.4 27.02.2009

- fixed SenderAnddress type in the SenderIdentification

Version 0.3 01.10.2008

- added MM7Client.ServiceCode property
- allow to specify Charged Party in the Submit method.