

SMS Broker System
Software Integration

Our system uses
WCF
(Windows Communication Foundation)
&
.NET

Annex. Importer Security Filing classes and interfaces.

Table of Contents

(ver. from 5/26/2026)

1. Sample. Context initialization.	1
2. Sample. Importer Security Filing objects reading from database.	1
3. Sample. New ISF object creating.	2
4. Sample. How to put ISF submission to queue.	6

1. Sample. Context initialization.

```
public class Common
{
    public static void InitializeServiceContext()
    {
        string url = @"http://someservice";
        string username = "somename";
        string database = "somedatabase";
        string password = "somepassword";

        // ClientContext - class that provides interaction between client and server
        // part of the application.
        // By InitializeContext() method the follow operations will be made:
        // Connect to service by uri,
        // login,
        // determine Transport type (encrypted or not),
        // get User properties ( ClientContext.User class will be initialized).
        ClientContext.InitializeContext(url, username + "@" + database, password);

        var userId = ClientContext.User.Id;           // Id in database
        var userCode = ClientContext.User.Code;       // Unique code (isf) used for login
        var userName = ClientContext.User.Name;       // Full name
    }
}
```

2. Sample. Importer Security Filing objects reading from database.

Here we read a list of ISF documents from database. `Common.InitializeServiceContext()` is a method defined in `Common` class in a previous sample.

```
protected void btGetList_Click(object sender, EventArgs e)
{
    Common.InitializeServiceContext();

    // Server part contains a realization of numerous managers (Service
    // Contracts). SMS.Broker.Services.ServiceTypes.Services is a collection
    // of the full list of managers. Each manager has a name (a part of uri)
```

```

        // and an interface (Service Contract).

var mngr =
ClientContext.ServicesFactory.GetManager<SMS.Broker.ServiceContracts.Documents.IImporterSecurityFilingManager>();

EntityPageQuery q = new EntityPageQuery();

q.Include = "Importer";           // Read from a database also property Importer by
                                // Importer_Id (Entity Framework "include")
q.PageSize = 10;                 // Get first 10 records (the big value can
                                // decrease an operation performance or cause a
                                // timeout exception)
q.QueryTotalCount = true;        // The result will have a total number of records
                                // (It requires an additional SQL query and
                                // decrease the operation performance)
q.RequestFullList = false;       // Get all records (It is only for small tables)
q.Position = 0;                 // Position of the first record in query result.

// q.Navigation = NavigationType.First;    // Request will return the first
//                                         // page list starting with the first
//                                         // record
// q.Navigation = NavigationType.Last;     // Request will return the last
//                                         // page list starting with the last
//                                         // record - q.PageSize
// q.Navigation = NavigationType.Next;     // Request will return the next page
//                                         // starting with q.Position +
//                                         // q.PageSize
// q.Navigation = NavigationType.Previous; // Request will return the previous
//                                         // page starting with q.Position -
//                                         // q.PageSize
q.Navigation = NavigationType.Refresh;    // Request will return a list
//                                         // starting with q.Position

// Order by Number descending
q.Order.Add(new OrderItem() { Name = "Number", IsDesc = true });

// Filter (where [Id] < 100L)
q.Criteria = "[Id] < 100L";

// Make a request:
var result = mngr.GetPage(q);

// Total count of records in full query result
// If q.QueryTotalCount = true
// result.TotalCount

// Number of record in Current page
// result.EntityPageCount

// true if it can be got the next non empty page
// result.HasNext

// true if it can be got the previous non empty page
// result.HasPrevious

// List of records in current page
// result.Entities;

// Query that is used to get this page
// result.Query;

GridView1.DataSource = result.Entities;
GridView1.DataBind();
}

```

3. Sample. New ISF object creating.

Here we create a new ISF object, fill its properties and collections with some values and save it to database. About navigation property and [PossibleValues](#) please see *02. Annex. Getting Started With SMS API, 1. Navigation property.* and *05. Base, common usage classes and interfaces.pdf, 9. Possible Values.* In this sample there are a lot of hardcoded string values are used. It is done specially in order not to complicate the samples. Real work, of course, suggests a work with a user interface or another source of an input data.

```

protected void btAddNewISF_Click(object sender, EventArgs e)
{
    Common.InitializeServiceContext();

    IImporterSecurityFilingManager mngrISF =
    ClientContext.ServicesFactory.GetManager<SMS.Broker.ServiceContracts.Documents.IImporterSecurityFilingManager>();

    IContactManager mngrContact =
    ClientContext.ServicesFactory.GetManager<SMS.Broker.ServiceContracts.Directories.IContactManager>();

    ICarrierManager mngrCarrier =
    ClientContext.ServicesFactory.GetManager<SMS.Broker.ServiceContracts.Directories.ICarrierManager>();

    IShipmentManager mngrShipment =
    ClientContext.ServicesFactory.GetManager<SMS.Broker.ServiceContracts.Documents.IShipmentManager>();

    #region prepare data to make a new ifs
    // It is suggested that really this data is taken from a user interface or some another source.
    // This sample has no a sophisticated user interface. Its purpose is only to show as it works.

    string SubmissionType = "1";
    string ShipmentType = "01";
    string EstimatedQuantityUnitOfMeasure = "PCS";
    string EstimatedWeightQualifiers = "K";

    string BondType = "8";
    string BondActivityCode = "01";
    string TransportationMode = "11";

    //Some user data
    string ClientRef = "ISF005";

    //SCAC code
    string CarrierCode = "MAEU";

    // Unique code of Contact in our database (Generated by our system on adding a new contact)
    // in this example is used as if the contact already was in a database
    string ImporterCode = "FRENEW";

    string ImporterName = "FREY NEWNAN";
    string ImporterAdress1 = "10 RAYMOND HILL ROAD";
    string ImporterAdress2 = "";
    string ImporterAdress3 = "";
    string ImporterCountry = "US";
    string ImporterState = "GA";
    string ImporterCity = "NEWNAN";
    string ImporterZIP = "30265";
    string ImporterIRS = "93-031218300";

    string BillIssuer = "JAPT";
    string BillINumber = DateTime.Now.ToString("MMddyyHHmmss"); // This "trick" is only for this sample to avoid
"duplicate BOL#"
// message from server.

    string BillType = "House"; // It can be House or Regular

    // Each party is a Contact type like Importer
    // so they can be processed in the same manner.
    // To simplify the sample only we use the same Contact Code as for Importer
    string ConsigneeCode = "FRENEW";
    string SellerCode = "FRENEW";
    string BuyerCode = "FRENEW";
    string ShipToCode = "FRENEW";
    string StuffingLocationCode = "FRENEW";
    string ConsolidatorCode = "FRENEW";

    string TariffNumber = "842820";
    string ManufacturerCode = "FRENEW";
    string CountryOfOrigin = "JP";
    #endregion

    // Get a new isf with filled Date and Number.
    // ISF class has a related ISF sequence class that is used to increment the next number.
    // Starting sequence Number can be adjusted (can begin not only from zero).

```

```

ImporterSecurityFiling newisf = mngrISF.New(null);

// Get the next sequence number:
var isfNumber = newisf.Number;

newisf.ClientRef = ClientRef;

// Some properties have a list of possible values (see doc).
// It is a static property with some name + "Values" (static List<PossibleValue> SubmissionTypeValues on this
case) // You can get a whole list of possible values. It is convenient if you want to use a lookup for user.
var listOfSubmissionTypePossibleValues = ImporterSecurityFiling.SubmissionTypeValues;

// On our simple sample it is hardcoded SubmissionType = "01" defined before:
newisf.SubmissionType = SubmissionType;

// String fields have a StringLength attribute with a field length.
// [StringLength(2)]
// public string ShipmentType { get; set; }
newisf.ShipmentType = ShipmentType;

newisf.EstimatedQuantityUnitOfMeasure = EstimatedQuantityUnitOfMeasure;
newisf.EstimatedWeightQualifiers = EstimatedWeightQualifiers;
newisf.BondType = BondType;
newisf.BondActivityCode = BondActivityCode;
newisf.TransportationMode = TransportationMode;

// Get Carrier from a database by Carrier Code
SMS.Broker.DataContracts.Directories.Carrier Carrier = mngrCarrier.GetByCode(CarrierCode, "");

// Important item - Carrier is a navigation property and we need to assign Carrier.Id to newisf.Carrier_Id.
// newisf.Carrier_Id has a higher priority against newisf.Carrier and exactly its value will be saved to database
// on save operation. Other words newisf.Carrier = Carrier is incorrect for save operation.
if (Carrier != null)
    newisf.Carrier_Id = Carrier.Id;

SMS.Broker.DataContracts.Directories.Contact Importer;
if (string.IsNullOrEmpty(ImporterCode)) // If contact is new.
{
    Importer = new SMS.Broker.DataContracts.Directories.Contact();

    Importer.Name = ImporterName;
    Importer.Address1 = ImporterAddress1;
    Importer.Address2 = ImporterAddress2;
    Importer.Address3 = ImporterAddress3;
    Importer.Country = ImporterCountry;
    Importer.State = ImporterState;
    Importer.City = ImporterCity;
    Importer.ZIP = ImporterZIP;
    Importer.IRSNumber = ImporterIRS;

    // Manufacturer/Supplier Code is required if contact is a manufacturer.
    // Use Customs rule to generate this code.
    Importer.GenerateManufacturerSupplierCode();
}
else
{
    Importer = mngrContact.GetByCode(ImporterCode, "");

    // You can use also Id if you know it:
    // Importer = mngrContact.Get(ImporterId, "");
}

// Return updated Contact from the database.
// It is a new entry and the result will have an Id and Code be generated on adding operation.
Importer = mngrContact.Save(Importer);

// For a single bond:
// newisf.BondType = "9";
// newisf.BondHolder = Importer;
// newisf.BondActivityCode = "16";
// newisf.BondReferenceNumber = "1111111111"
// newisf.SuretyCode = "123";

```

```

// Importer code can be used later for some additional operations.
ImporterCode = Importer.Code;

// Importer is a navigation property also.
newisf.Importer_Id = Importer.Id;

#region Add bill
// Get a new shipment with filled Date and Number.
// Shipment class has a related Shipment sequence class that is used to increment the next number.
// Starting sequence Number can be adjusted (can begin not only from zero).
Shipment shipment = mngrShipment.New(null);

// Carrier and Importer are navigation properties.
shipment.Carrier_Id = Carrier.Id;
shipment.Importer_Id = Importer.Id;

shipment.TransportationMode = TransportationMode;

if (BillType == "House")
{
    //For house bill #
    SMS.Broker.DataContracts.Directories.Carrier CarrierOnHouse = mngrCarrier.GetByCode(BillIssuer, "");

    // shipment.HouseBillIssuer is a navigation property.
    shipment.HouseBillIssuer_Id = CarrierOnHouse.Id;

    shipment.HouseBillNumber = BillINumber;
}
else if (BillType == "Regular")
{
    //For Regular bill #
    SMS.Broker.DataContracts.Directories.Carrier CarrierOnRegular = mngrCarrier.GetByCode(BillIssuer, "");

    // shipment.MasterBillIssuer is a navigation property.
    shipment.MasterBillIssuer_Id = CarrierOnRegular.Id;

    shipment.MasterBillNumber = BillINumber;
}

// Assign commercial Entities. They all are navigation properties.
SMS.Broker.DataContracts.Directories.Contact Consignee = mngrContact.GetByCode(ConsigneeCode, "");
if (Consignee != null)
    shipment.Consignee_Id = Consignee.Id;           // Assign Consignee's Id to Consignee_Id.

SMS.Broker.DataContracts.Directories.Contact Seller = mngrContact.GetByCode(SellerCode, "");
if (Seller != null)
    shipment.Seller_Id = Seller.Id;

SMS.Broker.DataContracts.Directories.Contact Buyer = mngrContact.GetByCode(BuyerCode, "");
if (Buyer != null)
    shipment.Buyer_Id = Buyer.Id;

SMS.Broker.DataContracts.Directories.Contact ShipTo = mngrContact.GetByCode(ShipToCode, "");
if (ShipTo != null)
    shipment.ShipTo_Id = ShipTo.Id;

SMS.Broker.DataContracts.Directories.Contact StuffingLocation = mngrContact.GetByCode(StuffingLocationCode, "");
if (StuffingLocation != null)
    shipment.StuffingLocation_Id = StuffingLocation.Id;

SMS.Broker.DataContracts.Directories.Contact Consolidator = mngrContact.GetByCode(ConsolidatorCode, "");
if (Consolidator != null)
    shipment.Consolidator_Id = Consolidator.Id;

// Add shipment to new ISF
newisf.Shipments.Add(shipment);
#endregion Add bill

#region Add Commodity
SMS.Broker.DataContracts.Documents.ImporterSecurityFilingCommodity commodity = new
    SMS.Broker.DataContracts.Documents.ImporterSecurityFilingCommodity();
commodity.HarmonizedTariffNumber = TariffNumber;

```

```

// Manufacturer is a navigation property.
SMS.Broker.DataContracts.Directories.Contact Manufacturer = mngrContact.GetByCode(ManufacturerCode, "");
if (Manufacturer != null)
    commodity.Manufacturer_Id = Manufacturer.Id;

commodity.CountryOfOrigin = CountryOfOrigin;

// Add commodity to new ISF
newisf.Commodities.Add(commodity);
#endregion Add Commodity

// save an new ISF to database with Shipments (Bills) and Commodities
newisf = mngrISF.Save(newisf);

// Get Id of new Isf in database
var NewIsfId = newisf.Id;
}

```

4. Sample. How to put ISF submission to queue.

Here we put to queue ISF document (SF-application) to send it to Customs (Add, Replace and Delete submissions). ISF object's Id is taken from tbISFId control. RblISF is a radiobutton control and action parameter of PutToQUEUEOneWay() method depends on its selection.

```

protected void btISFQue6_Click(object sender, EventArgs e)
{
    if (tbISFId.Text.Trim() == "")
    {
        tbISFMessage.Text = "ISF Id is absent.";
        return;
    }

    long ISFId = Convert.ToInt64(tbISFId.Text.Trim());

    Common.InitializeServiceContext();

    SMS.Broker.ServiceContracts.Documents.IImporterSecurityFilingManager mngrImporterSecurityFiling =
    ClientContext.ServicesFactory.GetManager<SMS.Broker.ServiceContracts.Documents.IImporterSecurityFilingManager>();

    ImporterSecurityFiling isf;
    isf = mngrImporterSecurityFiling.Get(ISFId, "");

    if (isf == null)
    {
        tbISFMessage.Text = "ISF is absent on database.";
        return;
    }

    ImporterSecurityFiling.ActionQUE action = 0;

    if (rblISF.Items[0].Selected) // ISF Add
    {
        action = ImporterSecurityFiling.ActionQUE.Add;
    }
    else if (rblISF.Items[1].Selected) // ISF Replace
    {
        action = ImporterSecurityFiling.ActionQUE.Replace;
    }
    else if (rblISF.Items[2].Selected) // ISF Delete
    {
        action = ImporterSecurityFiling.ActionQUE.Delete;
    }

    // This method places a request to que on our server.
    mngrImporterSecurityFiling.PutToQUEUEOneWay(ISFId, action);
}

```