

Indhold

Introduktion.....	2
Scenarier hvor API'et kan benyttes.....	2
Scenarie 1 – Integration til lagerhotel.....	2
Scenarie 2 – Integration til økonomi system	2
API Modeller	2
Webshop2 API Model v1	3
Webshop2 API Model v2	4
Brugen af API'et.....	5
Oprette forbindelse til Webshop2 via API'et.....	6
C# eksempel.....	6
VB.NET eksempel	7
C#	7
VB.NET.....	7

Senest opdateret:14. marts 2012

Introduktion

Webshop2 API'et gør det muligt at integrere ordredata fra Webshop ind i andre systemer. API er en forkortelse for *Application Programming Interface*.

API'et er baseret på Web Services ved hjælp af SOAP 1.2 standarden, som er en protokol der gør det muligt at udveksle XML, normalt via HTTP.

SOAP er supporteret af de fleste nyere programmeringssprog og udviklingsværktøjer fx. Visual Studio og lignende.

Scenarier hvor API'et kan benyttes

Med API'et kan man udvikle sin egen integration til 3. parts systemer, f.eks. lagerhotel, økonomi system mv.

Scenarie 1 – Integration til lagerhotel

Det er muligt i WebShop2 at oprette bruger definerede orderstatus typer fx 'Klar til pakning', 'Pakket', 'Klar til afsendelse' osv.

Disse brugerdefinerede orderstatustyper kan benyttes via API'et, når man i WebShop2 ændrer status på en ordre til 'Klar til pakning', kan man efterfølgende via API'et hente ordren (og andre ordrer med samme status) på en liste. Senere kan man – igen via API'et - ændre ordrens status til 'Klar til afsendelse', hvilket så vil fremgå af ordrelisten i WebShop2.

Scenarie 2 – Integration til økonomi system

Ved at benytte en ordres tilknyttede metoder *.GetInvoicedListByNotReplicated*, *.SetInvoicedReplicating* og *.SetInvoicedReplicated*, er det muligt at hente ordrer, der er markeret som faktureret i WebShop2, der endnu ikke er blevet replikeret.

.SetInvoicedReplicating er en tilstand der indikerer at man arbejder på ordren og *.SetInvoicedReplicated* sættes når man er færdig med behandle ordren i et økonomi system.

API Modeller

Der er idag flere versioner af API modellen, af den simple grund ikke at bryde kompatibiliten med 3. parts systemer der bebenytter en specifik version. Man kan se forskellen på API modellerne længere nede i dette dokument.

For at kunne bruge API'et skal man benytte URL'erne til Webservice'ne som er:

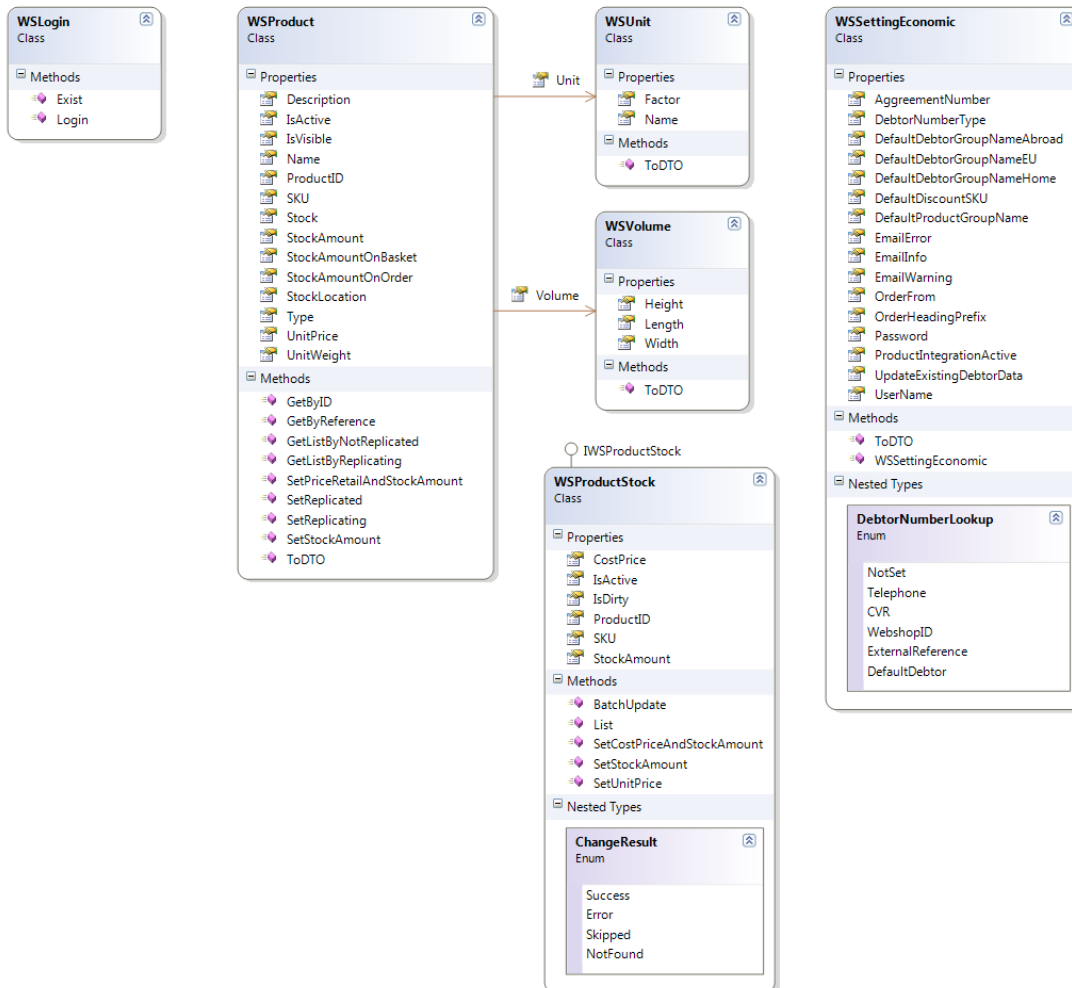
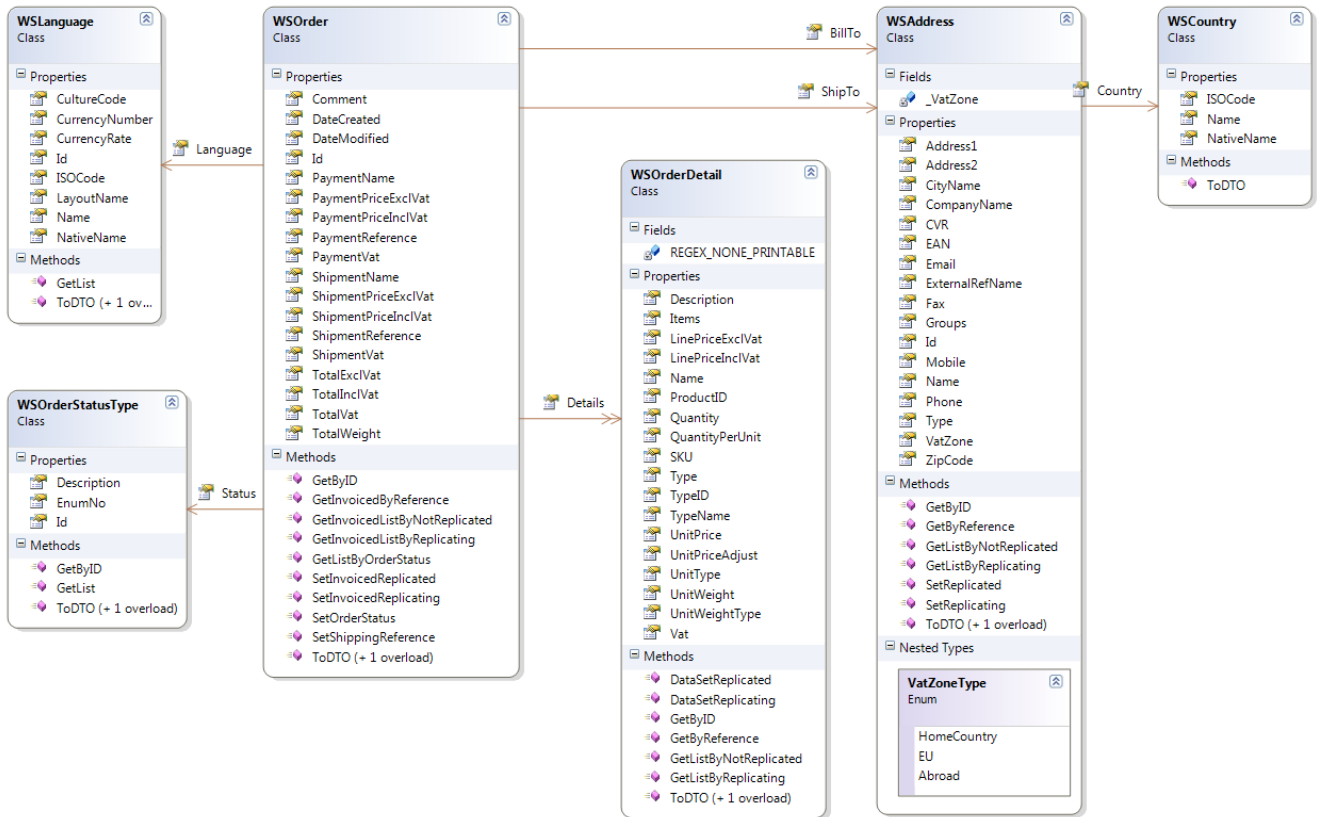
Version 1: [http://\[webshop2domæne\]/service/replicator.asmx](http://[webshop2domæne]/service/replicator.asmx)

Version 2: [http://\[webshop2domæne\]/service/replicatorV2.asmx](http://[webshop2domæne]/service/replicatorV2.asmx)

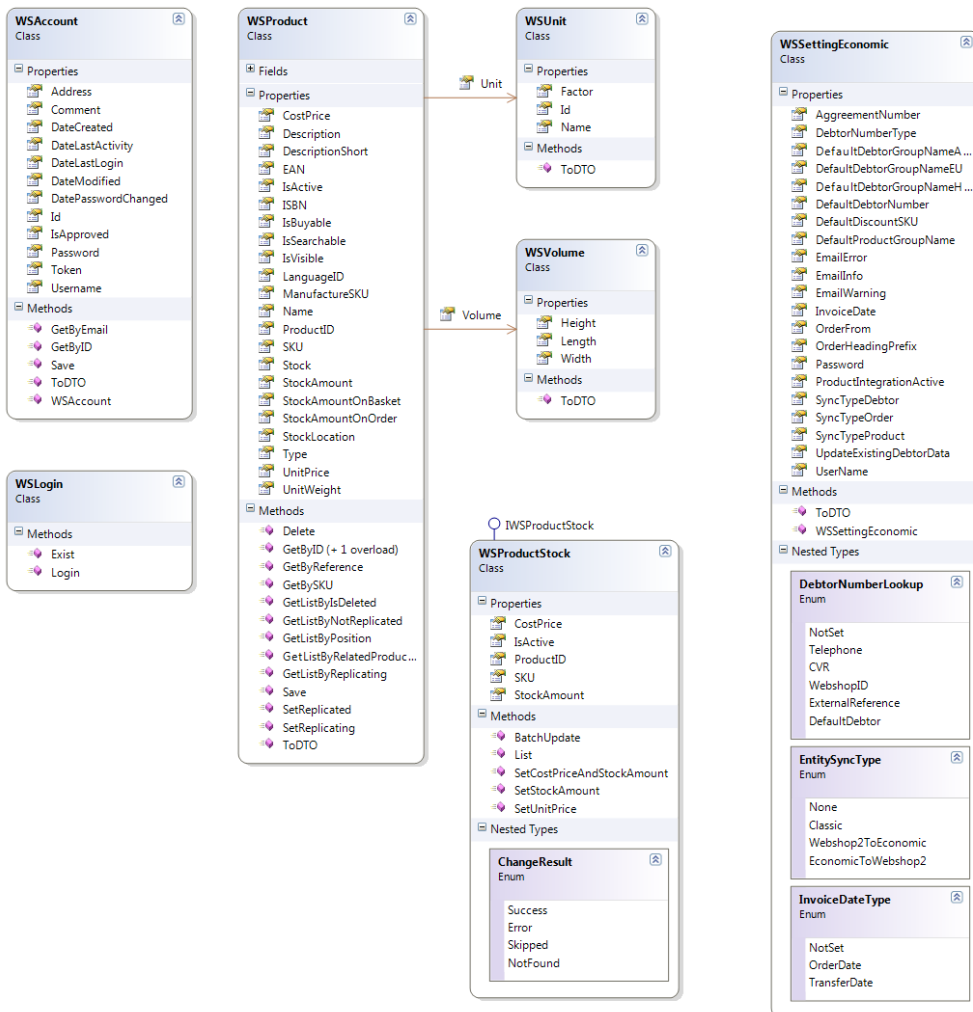
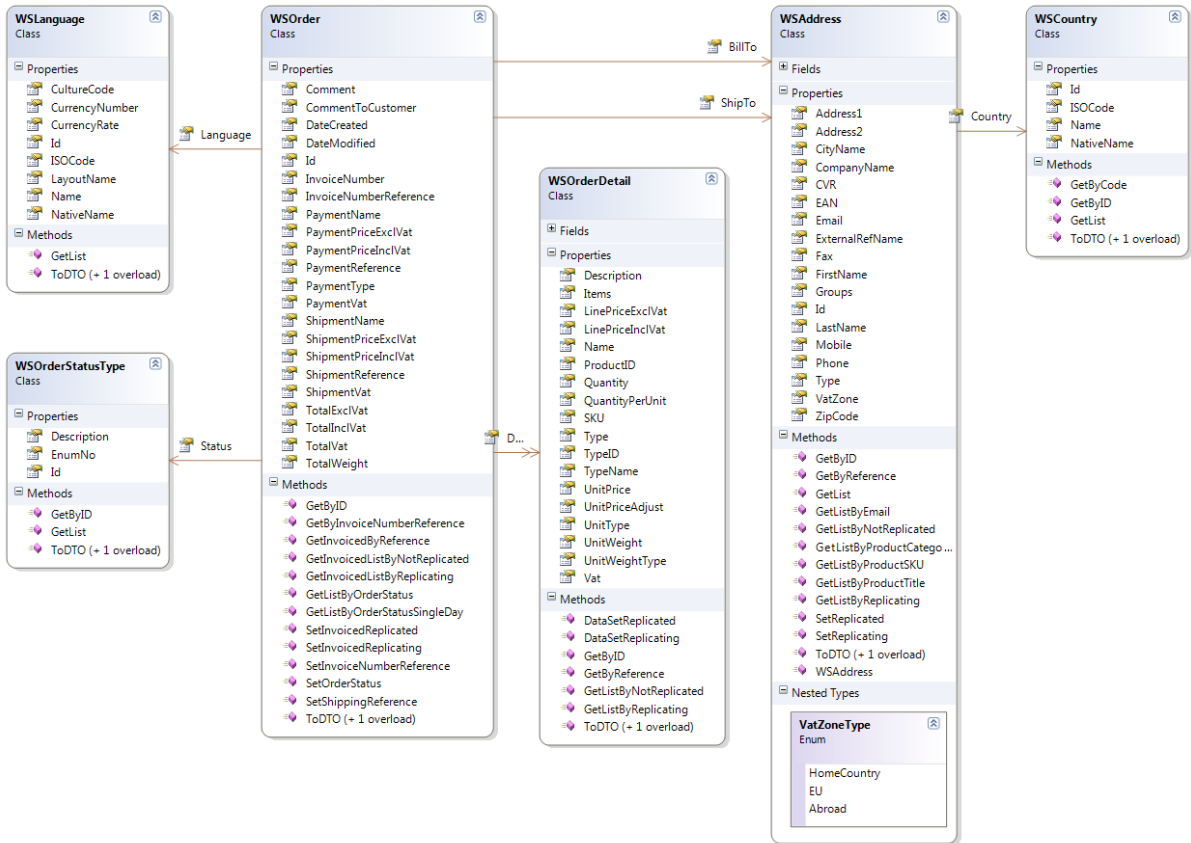
Version 3: [http://\[webshop2domæne\]/service/replicatorV3.asmx](http://[webshop2domæne]/service/replicatorV3.asmx)

I alle tilfælde skal *[webshop2domæne]* erstattes med Webshop2'ens rigtige domæne.

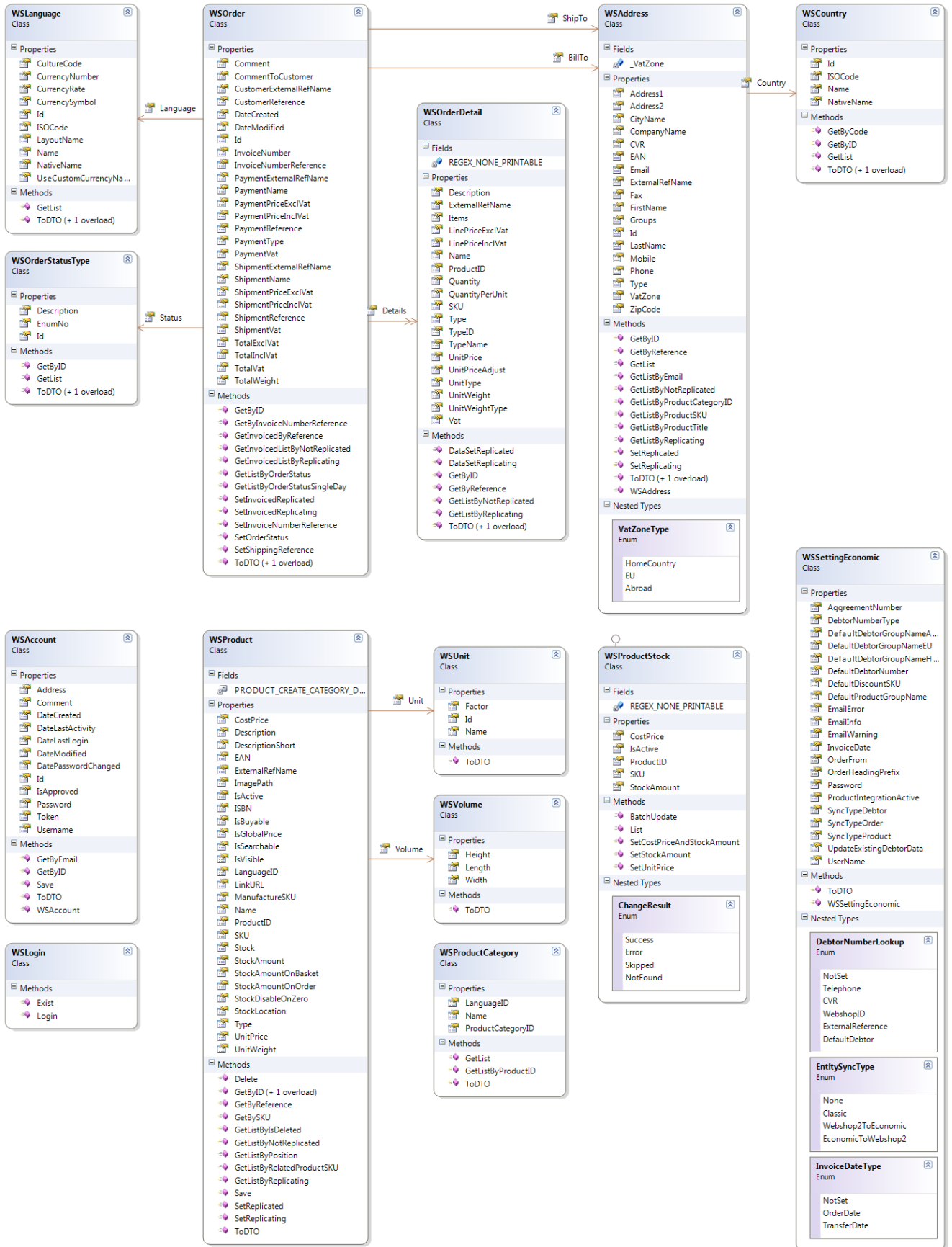
Webshop2 API Model v1



Webshop2 API Model v2 – September 2011



Webshop2 API Model v3 - Marts 2012



Brugen af API'et

Oprette forbindelse til Webshop2 via API'et

Når man skal logge på API'et, benyttes der en Webshop2 administrators brugernavn og adgangskode, og der returneres en Guid (Token), som skal benyttes ved samtlige efterfølgende kald til Webshop2 API'et.

Nedenstående eksempler viser brugen af webservicen, hvor der logges på til WebShop2 API, hentes en orderstatustype liste og efterfølgende en ordre liste på basis af en orderstatus type. Alle ordrer tildeles efterfølgende en ny status 'Tilbagebetaling'.

C# eksempel

```
// Variables
static wsWebshop2.Replicator _WebShop2;
static wsWebshop2.WSOrderStatusType[] _OrderStatusTypeList;

// Logon to WebShop2
static Guid WebShop2_Logon()
{
    _WebShop2 = new wsWebshop2.Replicator();

    Guid? token = _WebShop2.Login("username", "password");

    if (token.HasValue)
    {
        Console.WriteLine("Login success");
        return token.Value;
    }
    else
    {
        Console.WriteLine("Login failed");
        return Guid.Empty;
    }
}

// Get OrderStatusList
static void WebShop2_GetOrderStatusTypeList(Guid token)
{
    _OrderStatusTypeList = _WebShop2.OrderStatusType_GetList(token);

    foreach (var ost in _OrderStatusTypeList)
    {
        Console.WriteLine("Orderstatustype: ID={0}, Desc.={1}", ost.Id, ost.Description);
    }
}

// Get OrderList and do some work.
static void WebShop2_GetOrderList(Guid token)
{
    // Find order status type for Invoiced and Refund
    wsWebshop2.WSOrderStatusType ostInvoiced = _OrderStatusTypeList.ToList().Where(os => os.Description ==
"Faktureret").FirstOrDefault();
    wsWebshop2.WSOrderStatusType ostRefund = _OrderStatusTypeList.ToList().Where(os => os.Description ==
"Tilbagebetaling").FirstOrDefault();

    // Get 10 orders with status 'Invoiced' starting with Order no 5.
    wsWebshop2.WSOrder[] orderList = _WebShop2.Order_GetListByOrderStatus(token, ostInvoiced.Id, 5, 10);

    foreach (wsWebshop2.WSOrder order in orderList)
    {
        Console.WriteLine("Order: ID={0}, Date={1}", order.Id, order.DateCreated);

        // Change order status to 'Refund'
        _WebShop2.Order_SetOrderStatus(token, order.Id, ostRefund.Id);
    }
}
```

VB.NET eksempel

```
' Variables
Private _WebShop2 As wsWebshop2.Replicator = wsWebshop2.Replicator
Private _OrderStatusTypeList As wsWebshop2.WSOrderStatusType ()

' Logon to WebShop2
Private Function WebShop2_Logon() As Guid
    Dim token As Nullable(Of Guid)

    _WebShop2 = New wsWebshop2.Replicator()

    token = _WebShop2.Login("username", "password")

    If token.HasValue Then
        Console.WriteLine("Login success")
        Return token
    Else
        Console.WriteLine("Login failed")
        Return Guid.Empty
    End If
End Function

' Get OrderStatusList
Private Sub WebShop2_GetOrderStatusTypeList(ByVal token As Guid)

    _OrderStatusTypeList = _WebShop2.OrderStatusType_GetList(token)

    For Each ost As wsWebshop2.WSOrderStatusType In _OrderStatusTypeList
        Console.WriteLine("Orderstatyetype: ID={0}, Desc.={1}", ost.Id, ost.Description)
    Next
End Sub

' Get OrderList and do some work.
Private Sub WebShop2_GetOrderList(ByVal token As Guid)

    ' Find order status type for Invoiced and Refund
    Dim ostInvoiced As wsWebshop2.WSOrderStatusType = _OrderStatusTypeList.ToList().Where(Function(os) os.Description = "Faktureret").FirstOrDefault()

    Dim ostRefund As wsWebshop2.WSOrderStatusType = _OrderStatusTypeList.ToList().Where(Function(os) os.Description = "Tilbagebetaling").FirstOrDefault()

    ' Get 10 orders with status 'Invoiced' starting with Order no 5.
    Dim orderList As wsWebshop2.WSOrder () = _WebShop2.Order_GetListByOrderStatus(token, ostInvoiced.Id, 5, 10)

    For Each o As wsWebshop2.WSOrder In orderList
        Console.WriteLine("Order: ID={0}, Date={1}", o.Id, o.DateCreated)

        ' Change order status to 'Refund'
        _WebShop2.Order_SetOrderStatus(token, o.Id, ostRefund.Id)
    Next
End Sub
```

Hvis man istedet for en Web Service tilføjer en service reference i Visual Studio 2008, skal service variabelen deklareres lidt anderledes, resten er uændret.

C#

```
static private wsWebShop2.ReplicatorSoapClient _Webshop2;
```

VB.NET

```
Private _WebShop2 As wsWebshop2.ReplicatorSoapClient
```