Billig Shipping Script

The Billig shipping script, named 'Export_Billig' in the Airtox File, is for sending shipping orders to Billig (BA) for processing.
It runs as a MW script and uses CURL to send shipping orders over tcp/http. It is triggered by selecting Sales Invoices and pressing the “Export Billig” button.

Technical documentation

Key features:

- **MW Sales Invoice Sending:** Sends selected sales invoices to the Billig API.
- **Checks Sales Invoices not already processed:** Uses Transaction field user_4. Puts a timestamp if already sent. err:err text if error.
- **Validates Delivery Addresses:** Checks the delivery address meets criteria. More info below
- **Validates Country 2 Char representation:** Billig API uses 2 Char country digits for addresses. Uses JSON file to verify.

Missing features:

- Unable to resend if send failure. (MW script is probably not the place to do this. It is busy enough as it is)
- No local record of sent orders (with MW SQLite, this could be done. Again, script is pretty busy already)
- If when adding a Sales Line to Sales Header fails.... no resend available as per above. Just gets left off order (has only happened a few times)

BILLIG API Information:

API is locked down by IPs. The available IPs are Martin's Office IP, Airtox's Server IP, and possibly TMF or Sussol IPs.

Test Site Enpoints

- Sales Header endpoint: http://bc-test.billig-arbejdstoj.dk:18248/BC18_TEST_INTERNALIP_WINDOWUSER/ODataV4/Company('Airtox')/WS_3PL_Sales_Header
- Sales Line endpoint: http://bc-test.billig-arbejdstoj.dk:18248/BC18_TEST_INTERNALIP_WINDOWUSER/ODataV4/Company('Airtox')/WS_3PL_Sales_Line
- Header duplication check:

http://bc-test.billig-arbejdstoj.dk:18248/BC18_TEST_INTERNALIP_WINDOWUSER/ODataV4/Company('Airtox')/WS_3PL_Sales_Line

Live Site Enpoints

- Sales Header endpoint: http://bc.billig-arbejdstoj.dk:18648/BC18_WS/ODataV4/Company('Airtox')/WS_3PL_Sales_Header
- Sales Line endpoint: http://bc.billig-arbejdstoj.dk:18648/BC18_WS/ODataV4/Company('Airtox')/WS_3PL_Sales_Line
- Header duplication check: http://bc.billig-arbejdstoj.dk:18648/BC18_WS/ODataV4/Company('Airtox')/WS_3PL_Sales_Shipment_Lines?%24filter=Order_No%20eq%20

Auth info is hard coded in script. Test Auth info is commented out.

The API is what can be deduced as a Microsoft Dynamics 365 API. The API is very basic and not very ideal, but it has been rock solid for 3 years despite its shortcomings. Some of those short commings are:

- Cannot immediately check for duplicated orders. This is because orders are not available for retrieval until Billig manually processes them! 🥴
- Uses HTTP and not HTTPS
- The test site is never updated with data and rules from the “live” site, which makes it impossible to test anything properly. If any new shipping codes are added, just have to add code to live script, and hope you do it correctly.
- Uses antiquated NTLM for authentication. Luckily CURL has a method for this with option. Thanks Craig for finding this 🙂

```
Curl_SetOpt(ch, CURLOPT_HTTPAUTH, 8)
```

How API Works

- Before adding a Sales Header, check if order already exists using the Header Check duplication endpoint
- A Sales Header is created using the Sales Header endpoint
- Individual lines are added to Sales Header using the Lines Header endpoint

Sales Header Fields (text from script)

Document_No = “AI” + transaction.OurRef
External_Docuement_No = “Inv-” + Transaction.OurRef + “/Ord-” + Transaction.TheirRef
shippingCodeLines = See below how shippingCodeLines are calculated

```
let pdata = "{\n  "
```

```
let pdata = pdata + "\"Document_Type\": \"Ordre\",\n  "
let pdata = pdata + "\"Document_No\": \"" + docNumber + "\",\n  "
let pdata = pdata + "\"Customer_No\": \"" + billigCompanyNoPROP + "\",\n  "
let pdata = pdata + "\"Shipment_Date\": \"" + shipDate + "\",\n  "
let pdata = pdata + "\"Ship_to_Name\": \"" + shipToName + "\",\n  "
let pdata = pdata + "\"Ship_to_Address\": \"" + addr1 + "\",\n  "
let pdata = pdata + "\"Ship_to_Address_2\" : \"" + addr2 + "\",\n  "
let pdata = pdata + "\"Ship_to_City\": \"" + shipToCity + "\",\n  "
let pdata = pdata + "\"Ship_to_Post_Code\": \"" + shipToPostcode + "\",\n  "
let pdata = pdata + "\"Ship_to_Country_Region_Code\": \"" + shipToCountry +
"\",\n  "
let pdata = pdata + shippingCodeLines
let pdata = pdata + "\"External_Document_No\": \""+ extDocNumber +"\""
let pdata = pdata + "\n}
```

Sales Line Fields (text from script)

Document_No = “AI” + transaction.OurRef Line_No = sequential integer number. 1, 2, 3, etc…

```
let pdata = "{\n  "
let pdata = pdata + "\"Document_Type\": \"Ordre\",\n  "
let pdata = pdata + "\"Document_No\": \"" + docNumber + "\",\n"
let pdata = pdata + "\"Line_No\": \"" + lineNum  + "\",\n"
let pdata = pdata + "\"Type\": \"Vare\",\n"
let pdata = pdata + "\"Item_No\": \"" + itemCode + "\",\n"
let pdata = pdata + "\"Quantity\" : "  + quantity
let pdata = pdata + "\n}"
```

Shipping Code Lines

A few shipping codes are sent in the Sales Header. It consists of 2 fields.

- “Shipping_Agent_Code” : “predetermined shipping code created by Billig”
- “Shipping_Agent_Service_Code” : “predetermined shipping service code created by Billig

The logic depends (as of today, 17th Feb 2026):

- If internal order is for customer “BILLIG”. These are interal orders sent to Billig warehouse.
  - “Shipping_Agent_Code” = “AFHENT_AIR”
  - “Shipping_Agent_Service_Code” = “AFHENT_AIR”
- If destination country “DK” (Denmark)
  - “Shipping_Agent_Code” = “GLS”
  - “Shipping_Agent_Service_Code” = “GLS-ERH”
- If destination country “FO”, or “GL”, or “IS”
  - “Shipping_Agent_Code” = “UPS-AIRTOX”
  - “Shipping_Agent_Service_Code” = “UPS_Saver”
- If destination country is none of the above, and not “DK”
  - “Shipping_Agent_Code” = “UPS-AIRTOX”
  - “Shipping_Agent_Service_Code” = “UPS-AIRTOX”

Entry point:

```
on Before:F_TRANSLIST(windowRef)
  InstallToolBarIcon(windowRef, "Export_Billig")
end
```

Script Logic and Steps

- User highlights Sales Invoices to send and presses “Export Billig” button (recommended to only select a max of 10 as unable to give UI feedback of running process)
- Check each Transaction is of type “DI@” (debit invoice/Sales Invoice). Reject operation if not Sales Invoice
- Check Transaction.user_4 field is empty AND != “Err:NoResponse”. Reject if has timestamp or if equal to anything but “Err:NoResponse”
- Check there are Transactions selected. Reject if none selected
- Attempt to send transactions with following validations for each transaction
    - Validate again if user_4 is empty or anything but “Err:NoResponse” (redundant really as already did this!)
    - Check if Web Order. Check transaction.flag = “WEB”. **This feature is not being used as Airtox not doing web sales**
    - Parse transaction delivery addresses and validate
        - Address must be 4, 5, or 6 lines
        - If only 4 lines, third line must have City and Postcode on same line, in that order
        - First line is always the Recipient Name
        - Last line is always the country name, or country 2 code letters, nothing more. i.e “Denmark” or “DK”, “Denmark 8444” would fail.
        - Second to last line must always be JUST postocde, or have postcode as last value on line. i.e “Coppenhagen 8000”
        - Parse for 2 Char country code if full country name is given
    - Check all Transaction lines are from the “BA” location
    - Check order doesn't already exist with Billig
    - Create Sales Header
    - Add lines to Sales Header

From:
https://www.docs.sussol.net/ - **Sussol Docs**

Permanent link:
https://www.docs.sussol.net/doku.php/moneyworks:billig_shipping_script?rev=1771436999

Last update: **2026/02/18 17:49**