Extension Development

This chapter lists important integration events and examples to help you develop an extension for Freeze for MSDynamics 365 Business Central Important: Adaptations may only be carried out by appropriately trained consultants/developers. Furthermore, these are always outside the standard support provided by DEXPRO.

- Attachment Download

Attachment Download

Overview

The DXP Freeze Result Management codeunit provides two primary methods for downloading archived attachments as ZIP files:

1. `DownloadAttachmentsAsZipWithPagination` - Downloads attachments from search query results with pagination support
2. `DownloadAttachmentsAsZipFromRecord` - Downloads attachments from specific Business Central records

Both methods organize attachments into structured ZIP archives with comprehensive metadata for audit and tracking purposes.

Method 1: DownloadAttachmentsAsZipWithPa

Purpose

Downloads all attachments from a search query result set, processing results page by page to handle large datasets efficiently. Each record's attachments are organized into individual ZIP files within a main ZIP archive.

Overloads Available

1. Basic Usage

```
procedure DownloadAttachmentsAsZipWithPagination(SearchQuery: Text; var ZipArchive: Codeunit "Data Compression"; var AttachmentCount: Integer): Boolean
```

2. With Filters

```
procedure DownloadAttachmentsAsZipWithPagination(SearchQuery: Text; var ZipArchive: Codeunit "Data Compression"; var AttachmentCount: Integer; FilenameFilter: Text; FileExtensionFilter: Text): Boolean
```

3. Full Control

```
procedure DownloadAttachmentsAsZipWithPagination(SearchQuery: Text; var ZipArchive: Codeunit "Data Compression"; var AttachmentCount: Integer; FilenameFilter: Text; FileExtensionFilter: Text; StoreApiLink: Text; RecordsPerPage: Integer; SuppressDialog: Boolean): Boolean
```

4. Pre-populated Records (Advanced)

```
procedure DownloadAttachmentsAsZipWithPagination(var TempFrzResultQueryHeader: Record "DXP FRZ Query Result Header" temporary; var TempFrzResultRecordHeader: Record "DXP FRZ Record Result Header" temporary; var TempFrzResultRecordField: Record "DXP FRZ Result Record-Field" temporary; var TempFrzAttachmentResult: Record "DXP FRZ Attachment Result" temporary; SearchQuery: Text; var ZipArchive: Codeunit "Data Compression"; var AttachmentCount: Integer; FilenameFilter: Text; FileExtensionFilter: Text; SuppressDialog: Boolean): Boolean
```

Parameters

Parameter	Type	Description
`SearchQuery`	Text	Freeze search query string. If empty in pre-populated overload, uses existing query or re-executes search
`ZipArchive`	Codeunit “Data Compression”	ZIP archive object that will contain the downloaded files
`AttachmentCount`	Integer (var)	Returns the total number of attachments downloaded
`FilenameFilter`	Text	Filter for attachment filenames (e.g., ‘*.pdf*’, ‘*invoice*’)
`FileExtensionFilter`	Text	Filter for file extensions (e.g., ‘pdf’, ‘docx’)
`StoreApiLink`	Text	Optional specific store API link
`RecordsPerPage`	Integer	Number of records per page (default: 100)

Parameter	Type	Description
`SuppressDialog`	Boolean	Whether to suppress progress dialog

Return Value

- `Boolean`: `true` if attachments were found and downloaded; `false` otherwise

Example Usage

Basic Download

```
procedure DownloadSearchResults()
var
    ResultMgt: Codeunit "DXP FRZ Result Mgt.";
    ZipArchive: Codeunit "Data Compression";
    FileMgt: Codeunit "File Management";
    TempBlob: Codeunit "Temp Blob";
    AttachmentCount: Integer;
    InStr: InStream;
    OutStr: OutStream;
    SearchQuery: Text;
begin
    SearchQuery := 'invoice AND 2024';

    if ResultMgt.DownloadAttachmentsAsZipWithPagination(SearchQuery, ZipArchive,
AttachmentCount) then begin
        // Save ZIP to file
        TempBlob.CreateOutStream(OutStr);
        ZipArchive.SaveZipArchive(OutStr);
        TempBlob.CreateInStream(InStr);

        FileMgt.DownloadFromStreamHandler(InStr, '', '', '', 'SearchResults.zip');
        Message('Downloaded %1 attachments successfully.', AttachmentCount);
    end else
        Message('No attachments found for the search query.');
end;
```

With Filters

```
procedure DownloadPDFInvoices()
var
    ResultMgt: Codeunit "DXP FRZ Result Mgt.";
    ZipArchive: Codeunit "Data Compression";
    AttachmentCount: Integer;
    SearchQuery: Text;
begin
    SearchQuery := 'type:invoice';

    if ResultMgt.DownloadAttachmentsAsZipWithPagination(
        SearchQuery,
```

```
        ZipArchive,
        AttachmentCount,
        '*.pdf', // Only PDF files
        'pdf'    // File extension filter
    ) then begin
        // Process the ZIP archive
        ProcessDownloadedFiles(ZipArchive, AttachmentCount);
    end;
end;
```

Using Pre-populated Records

```
procedure DownloadFromExistingResults()
var
    ResultMgt: Codeunit "DXP FRZ Result Mgt.";
    TempFrzResultQueryHeader: Record "DXP FRZ Query Result Header" temporary;
    TempFrzResultRecordHeader: Record "DXP FRZ Record Result Header" temporary;
    TempFrzResultRecordField: Record "DXP FRZ Result Record-Field" temporary;
    TempFrzAttachmentResult: Record "DXP FRZ Attachment Result" temporary;
    ZipArchive: Codeunit "Data Compression";
    AttachmentCount: Integer;
begin
    // Assume these records are already populated from a previous search
    PopulateSearchResults(TempFrzResultQueryHeader, TempFrzResultRecordHeader,
TempFrzResultRecordField, TempFrzAttachmentResult);

    // Download using existing results without re-executing search
    if ResultMgt.DownloadAttachmentsAsZipWithPagination(
        TempFrzResultQueryHeader,
        TempFrzResultRecordHeader,
        TempFrzResultRecordField,
        TempFrzAttachmentResult,
        'invoice search', // SearchQuery - if empty, will re-execute search
        ZipArchive,
        AttachmentCount,
        '',     // No filename filter
        '',     // No extension filter
        true    // Suppress dialog
    ) then begin
```

```
        ProcessDownloadedFiles(ZipArchive, AttachmentCount);
    end;
end;
```

ZIP Structure (Pagination)

```
SearchResults.zip
├── export-metadata.json
├── Invoice_001_V1_20241201_1430.zip
│   ├── {GUID}_invoice.pdf
│   └── {GUID}_supporting_doc.docx
├── PurchaseOrder_002_V2_20241202_0900.zip
│   └── {GUID}_po_document.pdf
└── Contract_003_V1_20241203_1200.zip
    ├── {GUID}_contract.pdf
    └── {GUID}_amendment.pdf
```

Method 2: DownloadAttachmentsAsZipFromRe

Purpose

Downloads attachments from specific Business Central records. Each selected record’s attachments are organized into individual ZIP files within a main ZIP archive.

Overloads Available

1. Basic Usage

```
procedure DownloadAttachmentsAsZipFromRecord(var SelectedRecord: RecordRef; var ZipArchive:
Codeunit "Data Compression"): Boolean
```

2. With Filters

```
procedure DownloadAttachmentsAsZipFromRecord(var SelectedRecord: RecordRef; var ZipArchive: Codeunit "Data Compression"; FilenameFilter: Text; FileExtensionFilter: Text): Boolean
```

Parameters

Parameter	Type	Description
`SelectedRecord`	RecordRef (var)	RecordRef containing the selected Business Central records
`ZipArchive`	Codeunit “Data Compression”	ZIP archive object that will contain the downloaded files
`FilenameFilter`	Text	Filter for attachment filenames
`FileExtensionFilter`	Text	Filter for file extensions

Return Value

- `Boolean`: `true` if attachments were found and downloaded; `false` otherwise

Example Usage

Download from Sales Invoices

```
procedure DownloadInvoiceAttachments()
var
    SalesInvoiceHeader: Record "Sales Invoice Header";
    ResultMgt: Codeunit "DXP FRZ Result Mgt.";
    ZipArchive: Codeunit "Data Compression";
    RecordRef: RecordRef;
    HasAttachments: Boolean;
begin
    // Select specific invoices
    SalesInvoiceHeader.SetRange("Posting Date", DMY2Date(1, 1, 2024), DMY2Date(31, 12, 2024));
```

```
    SalesInvoiceHeader.SetFilter("Sell-to Customer No.", '10000|20000');

    if SalesInvoiceHeader.FindSet() then begin
        RecordRef.GetTable(SalesInvoiceHeader);

        HasAttachments := ResultMgt.DownloadAttachmentsAsZipFromRecord(RecordRef,
ZipArchive);

        if HasAttachments then
            SaveZipFile(ZipArchive, 'InvoiceAttachments.zip')
        else
            Message('No attachments found for the selected invoices.');
    end;
end;
```

Download with Filters from Page

```
// In a page extension
action(DownloadAttachmentsFiltered)
{
    Caption = 'Download Filtered Attachments';
    Image = ExportFile;

    trigger OnAction()
    var
        ResultMgt: Codeunit "DXP FRZ Result Mgt.";
        ZipArchive: Codeunit "Data Compression";
        RecordRef: RecordRef;
        FilenameFilter: Text;
        FileExtensionFilter: Text;
    begin
        // Show filter dialog
        if ShowFilterDialog(FilenameFilter, FileExtensionFilter) then begin
            CurrPage.SetSelectionFilter(Rec);
            RecordRef.GetTable(Rec);

            if ResultMgt.DownloadAttachmentsAsZipFromRecord(
                RecordRef,
                ZipArchive,
```

```
                FilenameFilter,
                FileExtensionFilter
            ) then
                DownloadZipFile(ZipArchive, 'FilteredAttachments.zip');
        end;
    end;
}
```

ZIP Structure (Records)

```
RecordAttachments.zip
├── export-metadata.json
├── Sales_Invoice_Header_Company_SI-001.zip
│   ├── {GUID}_invoice.pdf
│   └── {GUID}_terms.pdf
├── Sales_Invoice_Header_Company_SI-002.zip
│   └── {GUID}_invoice.pdf
└── Sales_Invoice_Header_Company_SI-003.zip
    ├── {GUID}_invoice.pdf
    ├── {GUID}_delivery_note.pdf
    └── {GUID}_receipt.jpg
```

Metadata Structure

Both methods generate comprehensive metadata in `export-metadata.json`:

Pagination Export Metadata

```
{
  "exportInfo": {
    "exportTimestamp": "2024-12-19T10:13:52.248Z",
    "exportedBy": "USER001",
    "searchQuery": "type:invoice AND year:2024",
    "totalRecordsFound": 150,
```

```
"totalPages": 15,
    "exportType": "paginated-search",
    "description": "Freeze Search Query Export"
  },
  "appliedFilters": {
    "filenameFilter": "*.pdf",
    "fileExtensionFilter": "pdf"
  },
  "statistics": {
    "pagesProcessed": 15,
    "totalRecordsProcessed": 150,
    "recordsWithAttachments": 120,
    "recordsWithoutAttachments": 30,
    "totalAttachments": 245,
    "exportCompletedAt": "2024-12-19T10:15:33.021Z"
  },
  "records": [
    {
      "recordId": "{GUID}",
      "title": "Invoice INV-2024-001",
      "version": 1,
      "archivedAt": "2024-12-01T09:30:00Z",
      "archivedBy": "SYSTEM",
      "type": "Sales Invoice",
      "masterId": "{GUID}",
      "attachmentCount": 3,
      "hasAttachments": true,
      "zipFile": "Invoice_INV-2024-001_V1_20241201_0930.zip"
    }
  ]
}
```


Record Export Metadata


```
{
  "exportTimestamp": "2024-12-19T14:30:00Z",
  "exportedBy": "USER001",
```

```
"totalRecordsProcessed": 25,
  "description": "DXP Freeze Attachments Export",
  "sourceTable": {
    "tableNumber": 112,
    "tableName": "Sales Invoice Header",
    "tableCaption": "Posted Sales Invoice"
  },
  "appliedFilters": {
    "filenameFilter": "*.pdf",
    "fileExtensionFilter": "pdf"
  },
  "statistics": {
    "totalAttachments": 45,
    "recordsWithAttachments": 20,
    "recordsWithoutAttachments": 5,
    "totalZipFiles": 20
  },
  "records": [
    {
      "recordId": "Sales Invoice Header: Company, SI-001",
      "systemId": "{GUID}",
      "primaryKey": {
        "fields": [
          {
            "fieldName": "No.",
            "fieldValue": "SI-001",
            "fieldType": "Code"
          }
        ]
      },
      "hasAttachments": true,
      "attachmentCount": 2,
      "zipFile": "Sales_Invoice_Header_Company_SI-001.zip"
    }
  ]
}
```

Performance Considerations

Pagination Method

- **Large Result Sets**: Automatically handles pagination to process large datasets efficiently
- **Memory Management**: Processes one page at a time, clearing memory between pages
- **Progress Tracking**: Shows real-time progress for long-running operations
- **Recommended For**: Search queries that may return hundreds or thousands of records

Record Method

- **Selected Records**: Processes only the records you specifically select
- **Direct Processing**: No pagination overhead for smaller datasets
- **Batch Processing**: Efficient for processing specific record sets
- **Recommended For**: Targeted downloads from specific Business Central records

Error Handling

Both methods include comprehensive error handling:

Common Scenarios

- **No Results Found**: Returns `false` when no records or attachments are found
- **Permission Issues**: Automatically excludes records the user cannot access
- **API Failures**: Gracefully handles API communication errors
- **Empty Filters**: Handles empty or invalid filter parameters

Best Practices

```
// Always check return value
if not ResultMgt.DownloadAttachmentsAsZipWithPagination(SearchQuery, ZipArchive,
```

```
AttachmentCount) then begin
    Message('No attachments found or download failed.');
    exit;
end;

// Validate attachment count
if AttachmentCount = 0 then begin
    Message('Search completed but no attachments matched the criteria.');
    exit;
end;

// Handle large downloads
if AttachmentCount > 1000 then
    if not Confirm('This will download %1 attachments. Continue?', false, AttachmentCount)
then
        exit;
```

Integration Events

Both methods support integration events for customization:

Available Events

- `OnBeforeDownloadAttachmentsAsZip`: Modify behavior before download starts
- `OnBeforeProcessAttachmentForZip`: Skip or modify individual attachments
- `OnAfterGetAttachmentBase64`: Modify attachment content after retrieval
- `OnAfterAddAttachmentToZip`: Perform actions after adding to ZIP
- `OnNoAttachmentsFound`: Handle no attachments scenario

Example Integration

```
[EventSubscriber(ObjectType::Codeunit, Codeunit::"DXP FRZ Result Mgt.",
'OnBeforeProcessAttachmentForZip', '', false, false)]
local procedure OnBeforeProcessAttachmentForZip(var TempFrzAttachmentResult: Record "DXP FRZ
Attachment Result" temporary; var IsHandled: Boolean)
```

```
begin
    // Skip attachments larger than 10MB
    if TempFrzAttachmentResult.Filesize > 10485760 then
        IsHandled := true;
end;
```

File Naming Conventions

Automatic Sanitization

All filenames are automatically sanitized using the `SanitizeFileName` method:

- Invalid characters (`< > : " / \ | ? *`) are replaced with underscores
- Spaces are replaced with underscores
- Maximum filename lengths are enforced

Unique Naming

- **Individual Files**: Include attachment GUID prefix to ensure uniqueness
- **ZIP Files**: Include record information and timestamps
- **No Conflicts**: Guaranteed unique names within each ZIP archive