Sage 300 Web Screens SDK

Drilldown Implementation

August 2022

The MIT License (MIT)

Copyright © 2022 The Sage Group plc or its licensors. All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Contents

[1. Introduction 4](#_Toc29896849)

[2. Web Screen Implementation 5](#_Toc29896850)

[2.1 Script File Implementation 5](#_Toc29896851)

[2.1.1 Database Setup Screen 5](#_Toc29896852)

[2.1.2 Sage Drilldown Script 5](#_Toc29896853)

[2.1.3 Partner Drilldown Script 7](#_Toc29896854)

[2.1.4 Summary 8](#_Toc29896855)

[2.2 Dynamic Entry Implementation 8](#_Toc29896856)

[2.2.1 Inserting into the Database 8](#_Toc29896857)

[2.2.2 Summary 9](#_Toc29896858)

[2.3 Drilldown Tokens 9](#_Toc29896859)

[2.3.1 Tokens 10](#_Toc29896860)

1. Introduction

With proper configuration (INI settings, data entry field population, etc.), the Sage 300 application provides a drilldown feature where, for example, an entry in GL may be able to drilldown to the source for that entry.

This functionality is available in the desktop today with no additional work other than the proper configurations as just mentioned.

The proper drilldown configuration and settings will not be discussed in this document

However, prior to Sage 300 2020.1, this capability was not available in the web screens for the following reasons:

* The web screen link (URL) for the drilldown was stored in an internal Resx file and thus partners were not able to add to this resource for their own URL
* The tokens used in the URL were evaluated and replaced by an internal routine not accessible to partners

Thus, partners were not able to specify this critical URL to be able to display a drilldown web screen.

This document is intended to provide information on the ability for partners to participate in the drilldown feature from the web screens.

1. Web Screen Implementation

As noted in the Introduction, the ability to display a drilldown in the web screens was limited to Sage 300 developers simply because the implementation did not consider third-party developers.

This changed in Sage 300 2020.1 to implement a single strategy for both internal and third-party developers.

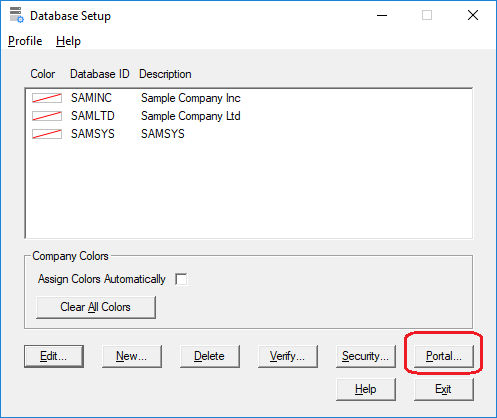
This section will explain the two options for providing the URL.

* 1. Script File Implementation

The Sage 300 web screens leverage the *Portal* or *Landlord* database for storing items such as user, tenant, workflow, inquiry and other items. This database is where we will store the URL for the drilldowns.

* + 1. Database Setup Screen

The *Database Setup Screen* in the desktop has a “Portal” button which discovers and consumes SQL scripts for populating tables in the database:



* + 1. Sage Drilldown Script

Sage has placed its drilldown script in the *../runtime/Database/Landlord/Scripts* folder so that when “Portal” button is selected, it will populate the drilldown table in the database with the values required for Sage 300 drilldowns.

The following is the content of the Sage 300 script (as of 2020.1):

|  |
| --- |
| -- Copyright (c) 1994-2019 Sage Software, Inc. All rights reserved.  -- Insert\_Drilldown\_Data.sql  SET ANSI\_NULLS ON  SET QUOTED\_IDENTIFIER ON  BEGIN  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('AP2100', '/AP/InvoiceEntry?batchNumber={BATCH}&entryNumber={ENTRY}&actionType={MODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('AP3100', '/AP/PaymentEntry?batchNo={BATCH}&entryNo={ENTRY}&FromScreen={MODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('AP4100', '/AP/AdjustmentEntry?batchNo={BATCH}&entryNo={ENTRY}&actionType={MODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('AR2100', '/AR/InvoiceEntry?batchNumber={BATCH}&entryNumber={ENTRY}&actionType={MODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('AR3100', '/AR/ReceiptEntry?batchNumber={BATCH}&entryNumber={ENTRY}&actionType={MODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('AR4100', '/AR/AdjustmentEntry?batchNumber={BATCH}&entryNumber={ENTRY}&actionType={MODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('AR5020', '/AR/RefundEntry?batchNumber={BATCH}&entryNumber={ENTRY}&actionType={MODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('BK2050', '/CS/BankTransfer?postSequence={POSTSEQ}&keySequence={KEYSEQ}&sequenceType={TYPE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('BK2300', '/CS/BankEntry?sequenceNumber={SEQUENCENO}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('BK6000', '/CS/ReconcileStatement?postSequence={POSTSEQ}&bankSequence={BANKSEQ}&sequenceType={TYPE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('IC3120', '/IC/Receipt?id={KEY}&disableAll={INQUIRYMODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('IC3210', '/IC/Shipment?id={KEY}&integratedScreen={INQUIRYMODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('IC3310', '/IC/Adjustment?id={KEY}&integratedScreen={INQUIRYMODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('IC3410', '/IC/Transfer?id={KEY}&isEditable={INQUIRYMODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('IC3530', '/IC/InternalUsage?id={KEY}&disableAll={INQUIRYMODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('OE1600', '/OE/CreditDebitNoteEntry?drillDownkey={KEY}&disableAll={INQUIRYMODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('OE1900', '/OE/InvoiceEntry?id={KEY}&disableAll={INQUIRYMODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('OE2200', '/OE/ShipmentEntry?drillDownKey={KEY}&isEditable={NOTINQUIRYMODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('PO1310', '/PO/ReceiptEntry?sequenceKey={KEY}&isNavigated={INQUIRYMODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('PO1320', '/PO/ReturnEntry?sequenceKey={KEY}&disableAll={INQUIRYMODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('PO1400', '/PO/InvoiceEntry?invoiceSequence={KEY}&isDisable={INQUIRYMODE}')  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('PO1500', '/PO/CreditDebitNoteEntry?sequenceKey={KEY}&disable={INQUIRYMODE}')  END  GO |

* + 1. Partner Drilldown Script

A partner will place its drilldown script in the *../runtime/Database/Landlord/Scripts* folder so that when “Portal” button is selected, it will populate the drilldown table in the database with the values required for the partner drilldowns.

The following is an example of a partner script (with a fictitious *XX* module and *InvoiceEntry* controller):

|  |
| --- |
| -- Copyright (c) 2019 Valued Partner. All rights reserved.  -- ValuedPartnerDrilldownData.sql  SET ANSI\_NULLS ON  SET QUOTED\_IDENTIFIER ON  BEGIN  INSERT INTO [dbo].[Drilldown](Id, Url) VALUES ('XX2100', '/XX/InvoiceEntry?batchNumber={BATCH}&entryNumber={ENTRY}&actionType={MODE}')  END  GO |

In the example above:

* The name can be whatever is specified by the partner, but it is recommended to include the partner name in the file as there may be multiple partners participating at a customer site
* The file must end in “sql”
* The *Id* property is the Roto Id specified in the action for drilldown
* The *Url* property is the URL to be invoked or the destination for the drilldown action
  + Url must start with the slash character
  + Url specifies the area (*XX*)
  + Url specifies the controller name (*InvoiceEntry*)
  + Url specifies any query string parameters required for the method (action)
  + In the examples above, methods/actions are not specified in any of the examples as no method/action specified will default to *Index*. Therefore, if a method/action other than *Index* is required, it will be specified after the controller name.
  + The tokens are finite and described later in this document

* + 1. Summary
* Create the script with the values required by the partner’s drilldown
* Partner is responsible for delivering the script to the customer site
* Customer or partner to run the *Database Setup* program, Select the *Portal* button to import the partner script into the database
* Partner drilldown is now possible
  1. Dynamic Entry Implementation

The Script File Implementation section described how a partner can use a script file delivered to the customer for partner participation in the drilldown feature.

Sage 300 believes this to be a valid enough delivery mechanism, but also realizes that partner requirements are at times different for external developers.

Since the drilldown information is stored in a database and the drilldown features reads this data, it is also possible for a partner module to directly insert their drilldown data into the database as opposed to allowing the script to insert it.

This would come in handy if the third-party developer either does not wish to deliver the script or if their drilldown URLs are dynamic in nature (i.e. based upon dynamic data, a URL may have different query string parameters, etc.).

We believe that this implementation will be rarely used, but nonetheless it is available.

* + 1. Inserting into the Database

The drilldown table in the database has been exposed via the web screen’s landlord service for allowing the required *CRUD* methods.

Using the example above for the partner script, the below snippet illustrates how this could be accomplished via a partner’s module code:

|  |
| --- |
| var service = Utilities.Resolve<ILandlordService>(context);  service.AddDrilldown(new Common.Models.Landlord.Drilldown()  {  Id = "XX2100",  Url = @"/XX/InvoiceEntry?batchNumber={BATCH}&entryNumber={ENTRY}&actionType={MODE}"  }); |

* + 1. Summary
* Instantiate the landlord service (if in the controller or service) or the landlord repository (if in the business repository)
* Invoke the *AddDrilldown* method with a populated *Drilldown* object
* Partner drilldown is now possible
* *GetDrilldown*, *UpdateDrilldown* and *DeleteDrilldown* methods are also available
  1. Drilldown Tokens

There are a finite set of tokens used by the drilldown framework and will be explained in this section.

Previously, the token replacement used the *String.Format()* function to replace the tokens, but that required knowing what tokens were used and in what order. For example:

|  |
| --- |
| string url = string.Format(@"/AP/InvoiceEntry?batchNumber={0}&entryNumber={1}&actionType={2}",  \_parser.GetBatchNumber(),  \_parser.GetBatchEntryNumber(),  \_parser.GetMode()); |

As you can see, not all URLs invoke the same endpoint with the same parameters and the same values. Therefore, the URLs use named tokens, such as:

|  |
| --- |
| /AP/InvoiceEntry?batchNumber={BATCH}&entryNumber={ENTRY}&actionType={MODE} |

Additionally, a partner module’s drilldown link may not be parsed like Sage’s drilldown link and therefore the parsing will need to be performed in the partner’s endpoint where the drilldown link is passed and once parsing is complete another endpoint can be invoked with the parameters supplied for contextual display of the screen, such as:

|  |
| --- |
| /XX/InvoiceEntry/Drilldown?id={DRILLDWNLK}} |

And, now any URL can have their tokens replaced with a single line:

|  |
| --- |
| // Replace tokens in URL  string url = urlFormat.Replace("{BATCH}", \_parser.GetBatchNumber())  .Replace("{ENTRY}", \_parser.GetBatchEntryNumber())  .Replace("{MODE}", \_parser.GetMode())  .Replace("{SEQUENCENO}", \_parser.GetSequenceNumber())  .Replace("{KEY}", \_parser.GetKey())  .Replace("{INQUIRYMODE}", \_parser.GetInquiryMode().ToString())  .Replace("{NOTINQUIRYMODE}", (!\_parser.GetInquiryMode()).ToString())  .Replace("{POSTSEQ}", \_parser.GetPostingSequence())  .Replace("{KEYSEQ}", \_parser.GetSequenceKey())  .Replace("{TYPE}", \_parser.GetTypeParameter())  .Replace("{BANKSEQ}", \_parser.GetBankSequence())  .Replace("{DRILLDWNLK}", \_parser.GetUnparsedParametersString()); |

* + 1. Tokens

This section will define the various tokens.

Since the tokens are finite and are mapped to the drilldown values in the drilldown framework, the use of any tokens not in this list will not result in a token replacement.

* {BATCH}
  + BATCH drilldown parameter value or empty if null
* {ENTRY}
  + ENTRY drilldown parameter value or empty if null
* {MODE}
  + “Inquiry”
* {SEQUENCENO}
  + SEQUENCENO drilldown parameter value or empty if null
* {KEY}
  + KEY drilldown parameter value or empty if null
* {INQUIRYMODE}
  + If INQUIRYMODE drilldown parameter is 1 then true otherwise false
* {NOTINQUIRYMODE}
  + Opposite of {INQUIRYMODE}
* {POSTSEQ}
  + POSTSEQ drilldown parameter value or empty if null
* {KEYSEQ}
  + KEYSEQ drilldown parameter value or empty if null
* {TYPE}
  + TYPE drilldown parameter value or empty if null
* {BANKSEQ}
  + BANKSEQ drilldown parameter value or empty if null
* {DRILLDWNLK}
  + The DRILLDWNLK value from the business view