

Activity Customization Examples In Sage SalesLogix v8.0

Sage SalesLogix White Paper



Introduction to Sage SalesLogix Job Server

Documentation This documentation was developed by Sage SalesLogix User Assistance. For content revisions, questions, or comments, contact the
Comments Sage SalesLogix writers at saleslogix.techpubs@sage.com.

Copyright Copyright © 1997-2012, Sage Software, Inc. All rights reserved.

This product and related documentation are protected by copyright and are distributed under licenses restricting their use, copying, distribution, and decompilation. No part of this product or related documentation may be reproduced in any form by any means without prior written authorization of Sage and its licensors, if any.

Version Version 8.0
 120412

Trademarks Sage SalesLogix is a registered trademark of Sage Software, Inc.

Sage, the Sage logos, SalesLogix, and the Sage product and service names mentioned herein are registered trademarks or trademarks of Sage Software, Inc., or its affiliated entities. All other trademarks are the property of their respective owners.

Disclaimer Sage has thoroughly reviewed this paper. All statements, technical information, and recommendations in this paper and in any guides or related documents are believed reliable, but the accuracy and completeness thereof are not guaranteed or warranted, and they are not intended to be, nor should they be understood to be, representations or warranties concerning the products described. Sage assumes no responsibility or liability for errors or inaccuracies with respect to this publication or usage of information. Further, Sage reserves the right to make changes to the information described in this manual at any time without notice and without obligation to notify any person of such changes.

Table of Contents

Activity Customization Examples In Sage SalesLogix v8.0	1
Purpose	1
Getting started	1
base.master changes.....	1
Example - 001 Add columns to the existing list views.	3
Example - 002 Extend Activity Editor with new fields.	3
Example - 003 Extend Activity Editor with new Association and Lookup control	3
Example - 004 Extend Activity Editor with new Agenda Tab	4
Example - 005 Extend Activity List Tab with new fields.....	4
Putting it all together	4
Appendix A: Sage Activity Parts	5
Appendix B: BigDeal Activity Parts.....	6
Appendix C: Entity Model	7
Appendix D: Smart Part Changes	8

DRAFT

Activity Customization Examples In Sage SalesLogix v8.0

Purpose

For release 8.0, Sage rewrote the base Activity system to use JavaScript, Dojo, and SData. The examples provided in this document show how to customize the Activity system by simply inheriting and extending the base.

Getting started

To implement the examples in this document, you should apply the Web bundle provided with it:

To apply

1. Use Application Architect to apply the BigDeal Activities Customization Bundle to your project. The bundle makes the following changes for you:
 - a. Installs the new schema and entity model described in Appendix C.
 - b. Overwrites the “base.master” page to include the “BigDeal” and example name spaces to the dojo loader and to add the require statement to load the “BigDeal/main.js” file on each page.
 - c. Installs the JavaScript files as described in Appendix B to the support files of the SlxClient Portal.
 - d. Overwrites the smart part files described in Appendix D.
2. Build and deploy the web site.

base.master changes

The bundle makes two changes to the base master page:

1. In order for the dojo loader to know where to locate the “BigDeal” namespace and files that will be used, it adds the name space configuration to the dojo loader configuration as follows in the Dojo Library Script:

```
<%-- DoJo Library --%>
<script pin="pin" type="text/javascript">
.....

var dojoConfig = {
  parseOnLoad: true,
  async: true,
  isDebug: false,
  locale: '<# Global.Locale %>',
  paths: {
    'Sage': '../jscript/Sage',
    'BigDeal': '../jscript/BigDeal'
    //'BigDeal': '../jscript/BigDeal_EX001'
    //'BigDeal': '../jscript/BigDeal_EX002'
    //'BigDeal': '../jscript/BigDeal_EX003'
  }
}
```

```

        //'BigDeal': '../..../jscript/BigDeal_EX004'
        //'BigDeal': '../..../jscript/BigDeal_EX005'
    },
    deferredOnError: function (e) {
        if (dojo.config.isDebugEnabled) {
        }
    }
};
</script>

```

2. It adds the require statement to load the “BigDeal/main.js” which will allow for loading any global modules that will be needed:

```

<script type="text/javascript">
    require([
        "dojo/_base/html",
        "dijit/_base/manager",
        "dojo/parser",
        "dijit/Toolbar",
        "dijit/layout/ContentPane",
        [... Other Requires]
        "Sage/Utility/File/DragDropWatcher",
        "Sage/Utility/File/DefaultDropHandler",
        "Sage/TaskPane/ActivityTaskPaneActions",
        "BigDeal/main"
    ],
    [ ... Other Scripts]
</script>

```

Examples

Note that each example is mapped to the BigDeal name space, and by default, all of the examples are combined into one customization under the “BigDeal” folder. In order to load just one example of interest, comment out the mapping to the “BigDeal” default folder and uncomment out one of the examples as shown below:

For Example EX005:

```

paths: {
    'Sage': '../..../jscript/Sage',
    //'BigDeal': '../..../jscript/BigDeal'
    //'BigDeal': '../..../jscript/BigDeal_EX001'
    //'BigDeal': '../..../jscript/BigDeal_EX002'
    //'BigDeal': '../..../jscript/BigDeal_EX003'
    //'BigDeal': '../..../jscript/BigDeal_EX004'
    'BigDeal': '../..../jscript/BigDeal_EX005'
},

```

Example - 001 Add columns to the existing list views

This example shows how to add three columns (Source, UserField1, and ContractId) to the existing “My Activities” and “All Open” tabs on the Activity main list view by overriding and extending the following:

- Sage/MainView/ActivityManager
- Sage/MainView/ActivityMgr/ActivitiesListPanelConfig
- Sage/MainView/ActivityMgr/AllOpenListPanelConfig”

Review the following sample files:

- jscript/BigDeal_EX001/main.js
- jscript/BigDeal_EX001/ActivityManger.js
- jscript/BigDeal_EX001/ActivitiesListPanelConfig.js
- jscript/BigDeal_EX001/AllOpenListPanelConfig.js

Example - 002 Extend Activity Editor with new fields

This example shows how to add and bind two fields (Source and UserField1) to the Activity Editor by overriding and extending the following:

- Sage/MainView/ActivityManager
- Sage/MainView/ActivityMgr/ActivityEditor
- Sage/MainView/Services/ActivityService

Review the following sample files:

- jscript/BigDeal_EX001/main.js
- jscript/BigDeal_EX002/ActivityManger.js
- jscript/BigDeal_EX002/ActivityEditor.js
- jscript/BigDeal_EX002/ActivityService.js

Example - 003 Extend Activity Editor with new association and Lookup control

This example shows how to add and bind a Contract association with a lookup control to the Activity Editor by overriding and extending the following:

- Sage/MainView/ActivityManager
- Sage/MainView/ActivityMgr/ActivityEditor
- Sage/MainView/Services/ActivityService

Review the following sample files:

- jscript/BigDeal_EX003/main.js
- jscript/BigDeal_EX003/ActivityManger.js
- jscript/BigDeal_EX003/ActivityEditor.js
- jscript/BigDeal_EX003/ActivityService.js

Example - 004 Extend Activity Editor with new Agenda tab

This example shows how to add a 1 to Many Activity Agenda association in a new tab to the Activity Editor by overriding and extending the following:

- Sage/MainView/ActivityManager
- Sage/MainView/ActivityMgr/ActivityEditor
- Sage/MainView/Services/ActivityService

Review the following sample files:

- jscript/BigDeal_EX004/main.js
- jscript/BigDeal_EX004/ActivityManger.js
- jscript/BigDeal_EX004/ActivityEditor.js
- jscript/BigDeal_EX004/ActivityService.js
- jscript/BigDeal_EX004/ ActivityEditorAgendaItemsTab.js

Example - 005 Extend Activity List tab with new fields

This example shows how to add and bind two fields (Source and UserField1) to the Activity Editor by overriding and extending the following:

- Sage/UI/ActivityList
- SmartPart/Activity/ActivitList.acsx.cs

Review the following sample files:

- jscript/BigDeal_EX004/main.js
- jscript/BigDeal_EX005/ActivityTab.js

Putting it all together

To see all the customizations working together, see all the examples combined under the Big Deal folder. For a complete list of the files, see Appendix B.

Appendix A: Sage Activity Parts

Following is a list of the components of the Activity system before it is customized:

Name	Description
jscript/Sage/MainView/ActivityManager.js	
jscript/Sage/MainView/ActivityMgr/ActivityGroupContextService.js	
jscript/Sage/MainView/ActivityMgr/ActivityListPanelConfig.js	
jscript/Sage/MainView/ActivityMgr/AllOpenListPanelConfig.js	
jscript/Sage/MainView/ActivityMgr/PastDueListPanelConfig.js	
jscript/Sage/MainView/ActivityMgr/AlarmListPanelConfig.js	
jscript/Sage/MainView/ActivityMgr/EventListPanelConfig.js	
jscript/Sage/MainView/ActivityMgr/ConfirmationListPanelConfig.js	
jscript/Sage/MainView/ActivityMgr/ActivityEditor.js	
jscript/Sage/MainView/Services/Activity.js	
jscript/Sage/MainView/Utilitiy/Activity.js	
jscript/Sage/TaskPane/ActivityTaskConfigurationProvider.js	
jscript/Sage/TaskPane/ActivityManagerTasklet.js	
SmartParts/General/SDataListViewer.ascx	Note smart configured in AA to load Sage/MainView/ActivityManager.js
SmartParts/Activity/ActivityList.ascx.cs	Note smart is explicitly hard coded to load Sage/UI/ActivityList.js
SmartParts/TaskPane/ActivityManager/ActivityManagerTasks.ascx	Note smart is explicitly hard coded to load Sage/TaskPane/ActivityManagerTasklet.js and Sage/TaskPane/ActivityTaskConfigurationProvider.js

Appendix B: BigDeal Activity Parts

Following is the list of files that make up the complete Activity customization:

Name	Description
jscript/BigDeal/ActivityManager.js	
jscript/BigDeal/ActivityGroupContextService.js	
jscript/BigDeal/ActivityListPanelConfig.js	
jscript/BigDeal/AllOpenListPanelConfig.js	
jscript/BigDeal/ActivityEditor.js	
jscript/BigDeal/ActivityService.js	
jscript/BigDeal/Utiltiy.js	
jscript/BigDeal/ActivityTaskConfigurationProvider.js	
jscript/BigDeal/main.js	
jscript/BigDeal/ActivityEditorAgendaItemsTab.js	
jscript/BigDeal/ActivityTab.js	
SmartParts/General/SDataListViewer.ascx	Note smart configured in AA to load Sage/MainView/ActivityManager.js
SmartParts/Activity/ActivityList.ascx.cs	Note smart is explicitly hard coded to load Sage/UI/ActivityList.js
SmartParts/TaskPane/ActivityManager/ActivityManagerTasks.ascx	Note smart is explicitly hard coded to load Sage/TaskPane/ActivityManagerTasklet.js and Sage/TaskPane/ActivityTaskConfigurationProvider.js

Appendix C: Entity Model

Following are the new schema and entity model added through the bundle:

Activity (New Fields)

Property	Type	Description
Source	Text (64)	
ContractId	SalesLogixId	
ContractNumber	Text (64)	
ActivityAgendaItems	Relationship	

ActivityAgendaItem (New Table)

Property	Type	Description
ActivityAgendaItemId	SalesLogixId	
ActivityId	SalesLogixId	
ItemOrder	Integer	
AgendaItem	Text (64)	
Note	Text (512)	
Activity	Relationship	

Appendix D: Smart Part Changes

To customize the Activity Manager, customizers can configure the actmgrSDataListView smart part to use a custom SDataList View Provider. For the customizations in this example, the bundle does this for you; it changes “MainViewDefinition” from “Sage.MainView.ActivityManager” to “BigDeal.ActivityManager”. If you want to revert, click Configure Smart Part and set MainViewDefinition=Sage.MainView.ActivityManager.

The screenshot displays the configuration interface for the **actmgrSDataListView (Custom)** smart part. The **actmgrSDataListView (Custom)** configuration section is highlighted with a red circle around the **Configure Smart Part** button.

The **Smart Part Properties: SmartParts\General\SDataListView.ascx** window shows the following properties:

Property	Value
ClientIDMode	
Enable Theming	
Enable ViewState	
ViewStateMode	
Visible	
Misc	
(ID)	
DetailPaneDefaultHeight	
DetailPane Type	
DetailPane VisibleOnLoad	
DetailsPaneConfigFile	
Help TopicName	activitymgrlistview
ListContextMenu	ActivityManagerListContext Menu
MainViewDefinition	BigDeal.ActivityManager
MainViewDefinitionForwardSlash	
SummaryConfigFile	
TabContextMenu	