

# Embedded Mash-Ups

## Getting Started Samples





# Table of Contents

Useful S-Controls.....	3
Useful Custom Buttons and Links.....	8
S-Control Terminology.....	12
<b>Index.....</b>	<b>14</b>



## Useful S-Controls

Available in: **Group, Professional, Enterprise, Unlimited, and Developer** Editions

Custom buttons and links are available in: **All** Editions

S-controls are available in: **Group, Professional, Enterprise, Unlimited, and Developer** Editions

Overriding standard buttons and tab home pages is available in: **Enterprise, Unlimited, and Developer** Editions

Use the following samples to get started using s-controls:

- [S-Controls for Detail Pages](#) on page 3
- [S-Controls that Override Standard Buttons and Tab Home Pages](#) on page 4
- [S-Controls that Include Snippets](#) on page 7

## S-Controls for Detail Pages

### Yahoo Map

Use the Yahoo Map API and the billing address merge fields to display a map for an account. Use the following code in an HTML s-control and add it to your account detail page layout:

```
<html>
<head>
<script type="text/javascript"
src="http://api.maps.yahoo.com/ajaxymap?v=3.0&appid=YahooDemo">
</script>
<style type="text/css">
#mapContainer {
height: 200px;
width: 100%;
}
</style>
</head>
<body>
<div id="mapContainer"></div>
<script type="text/javascript">
// Create a map object
var map = new YMap(document.getElementById('mapContainer'));
// Display the map centered on given address
map.drawZoomAndCenter("{!Account.BillingStreet}, \
    {!Account.BillingCity}, \
    {!Account.BillingState}, \
    {!Account.BillingPostalCode}", 3);
// Set marker at that address
map.addMarker("{!Account.BillingStreet}, \
    {!Account.BillingCity}, \
    {!Account.BillingState}, \
    {!Account.BillingPostalCode}", 3);
</script>
</body>
</html>
```

## S-Controls that Override Standard Buttons and Tab Home Pages

### Add Product Override

You may have your own code that you prefer to use for adding products to opportunities instead of the standard page. Use the s-control sample below to pass data values using merge fields from a record detail page into a custom s-control that overrides the **Add Product** button on the Products related list of an opportunity. This type of override illustrates how related list buttons can contain merge fields from the master object as well as the detail. For example, the code below contains opportunity merge fields, which is on the master side of a master-detail relationship with opportunity products.

```
<html>
<head>
<script type="text/javascript"
src="/soap/ajax/13.0/connection.js">
</script>
</head>
<body>
<b>Opportunity Info:</b>
<br>
Opportunity ID: {!Opportunity.Id}
<br>
Opportunity Name: {!Opportunity.Name}
<br>
Opportunity Record Type: {!Opportunity.RecordType}
<br>
</body>
</html>
```

To implement this functionality, create an HTML s-control with the content above inserting your code in the space provided. Then, override the add product action from the opportunity products object using the s-control. This example assumes you have record types on opportunities.



**Note:** This example does not include the code to add products. The content in the body section simply illustrates how to use opportunity merge fields from the opportunity products related list. Replace the body section with your code.

### Conditional Override for Editing Leads

You can override a standard action conditionally, redirecting to a standard action or custom s-control depending on certain conditions. For example, you may want to use a separate s-control to edit leads when they have been open longer than 30 days. Using the following example, create an s-control to evaluate if a lead has been open longer than 30 days and, if so, run your custom s-control to edit leads. Otherwise, use the standard lead edit action.

```
<script type="text/javascript">

//determine if the lead has been open longer than 30 days
if ({!IF(ISPICKVAL( Lead.Status , "Open"), ROUND(NOW()- Lead.CreatedDate , 0), 0)} > 30)
{
//more than 30 days - display a custom scontrol page
window.location.href="{!URLFOR($SControl.EditLeadsOpenLongerThan30)}";
}
else
{
//30 days or less - display the standard edit page
window.parent.location.href="{!URLFOR($Action.Lead.Edit, Lead.Id,
[retURL=URLFOR($Action.Lead.View, Lead.Id)], true)}";
}

</script>
```

To implement this in your organization, create the s-control that you want to use to edit leads that have been open longer than 30 days. Name this s-control `EditLeadsOpenLongerThan30`. Next, create an s-control using the example code above to determine if a lead has been open longer than 30 days, and, if so, override the edit action on leads using the `EditLeadsOpenLongerThan30` s-control.

Note the differences between the first and second "if" statements in the example code above. The first one is a JavaScript "if" statement that evaluates on the browser. The second is the Salesforce IF function that evaluates on the server and returns a single value—the number of days the lead has been open, or zero if the lead is not open.



**Tip:** Use the URLFOR function in this example to build Salesforce URLs rather than specifying individual URLs to ensure they are supported across releases.

To display a standard Salesforce page without invoking the override, set the `no override` argument in the URLFOR function to "true."

Also, use the "retURL" parameter in your URLFOR function to return the user to the detail page after saving.

### Edit Contact Override

You may have your own code that you prefer to use for editing contacts. Use the s-control sample below to pass data values using merge fields from a record detail page into a custom s-control that overrides a standard detail page button.

```
<html>
<head>
<script type="text/javascript" src="/soap/ajax/13.0/connection.js">
</script>
</head>
<body>
<b>Contact Info:</b>
<br>
Contact ID: {!Contact.Id}
<br>
Contact Name: {!Contact.FirstName} {!Contact.LastName}
<br>
</body>
</html>
```

To implement this functionality, create an HTML s-control with the content above inserting your code in the body section. Then, override the edit contact action using the s-control. This overrides the edit contact action everywhere it is available: the **Edit** button on a contact detail page, the **Edit** link on list views, and the **Edit** link on any related lists.



**Note:** This example does not include the code to edit contacts. The code within the body section only illustrates how to use contact merge fields to display information about the contact. Replace the body section with your code.

### Interrupt Override for New Accounts

Overriding standard buttons makes them unavailable in your entire Salesforce organization. However, you can override a standard action and redirect to that action from your s-control without getting into an infinite loop. For example, you can override the **New** button on accounts, perform your own custom process, and resume with the standard new account action without getting into an infinite loop. To do this, use the `no override` argument in the URLFOR function.

```
<script type="text/javascript">

alert("Hi, I am demonstrating how to interrupt New Account with an override. Click OK to continue.");

window.parent.location.href="{! URLFOR($Action.Account.New, null, null, true)}";
```

```
</script>
```

To implement this s-control, create an HTML s-control with the content above. Then, override the new account action using the s-control.



**Note:** The new action does not require an ID, which is why the second argument in the URLFOR function is set to "null." This example does not require any inputs, which is why the third argument in the URLFOR function is set to "null." The fourth argument in the URLFOR function is set to "true" to ignore the override, avoiding an infinite loop.

## Close Case Page Override

Validation rules are a useful way of validating data on a record before saving it. However, you may want to validate the data associated with a record. Use an s-control like the one below to ensure that a case is associated with at least one solution before users are allowed to close it:

```
<html>
<head>
<script src="/soap/ajax/13.0/connection.js">
</script>
<script>
function init()
{
var queryResult = sforce.connection.query("Select count() from CaseSolution Where CaseId =
' {!Case.Id} '");
var size = queryResult.size;
if (size > 0) {
//go to the standard case close page without invoking the override
window.parent.location.href = "{!URLFOR($Action.Case.CloseCase, Case.Id,
[retURL=URLFOR($Action.Case.View, Case.Id)], true)}"
}
else
{
alert("Case must contain at least one solution before it can be closed.");
//go to the standard case detail page
this.parent.location.href = "{!URLFOR($Action.Case.View, Case.Id)}";
}
}
</script>
</head>
<body onload="init()">
<p>&nbsp;</p>
</body>
</html>
```

To implement this functionality, create an HTML s-control with the content above. Then, override the close case action using the s-control. The s-control displays a message to users when they click **Close Case** on a case with no associated solutions. If the case is associated with at least one solution, the standard close case page displays as usual. This override does not apply to records updated using the API.



**Tip:** Use the URLFOR function in this example to build Salesforce URLs rather than specifying individual URLs to ensure they are supported across releases.

To display a standard Salesforce page without invoking the override, set the *no override* argument in the URLFOR function to "true."

Also, use the "retURL" parameter in your URLFOR function to return the user to the detail page after saving.

For more information on using this function, see "URLFOR" in the Salesforce online help.

## Conditional Accounts Tab Home Page Override

You can override a tab home page conditionally, redirecting the original tab home page to an s-control depending on certain conditions. For example, you may want to display an s-control, instead of the standard Accounts tab home page, to users with a specific profile. Using the following sample code, create an s-control to display job applicant information to users with the Recruiter profile when they click the Accounts tab; for all other users, display the standard Accounts tab home page.

To implement this, first create an s-control called "ApplicantHomePage" that contains the content to display to recruiters. Next create an s-control of type HTML using the following code to implement the conditional override logic:

```
<script type="text/javascript">
//determine the user profile name
var recruiter = {!IF($Profile.Name = "Recruiter", true, false)};

//when the profile is recruiter - display a custom s-control page
if (recruiter) {
    window.parent.location.href="{! urlFor($SControl.ApplicantHomePage)}";
} else {
//when the profile is not recruiter - display the standard Accounts tab page
    window.parent.location.href="{! urlFor( $Action.Account.Tab ,
$ObjectType.Account,null,true)}";
}
</script>
```

Finally, override the Accounts tab to use the HTML s-control shown here. This example assumes that a profile named "Recruiter" exists in your organization.



**Note:** \$Profile merge fields are only available in Enterprise, Unlimited, and Developer Editions.

## S-Controls that Include Snippets

### Including Snippets

Include snippets in your custom s-controls to reuse common code. The following example references a snippet that provides a header for a page that displays in a web tab. The page will have the title "My Title." Use the \$SControl global variable to reference a snippet. To implement this, create two snippets called "Resize\_Iframe\_head" and "Resize\_Iframe\_onload" and create an HTML s-control called "Resize\_Iframe\_sample" that includes the following code:

```
<html>
<body>
{! INCLUDE($SControl.Header_Snippet,
[title = "My Title", theme = "modern"])}
</body>
</html>
```

For more information on using this function, see "INCLUDE" in the Salesforce online help.

### Resizing Iframes

Dynamically resize the height of the iframe for s-controls on detail pages to make it big enough to fit the full content of the custom s-control. This sample requires two snippets:

- Resize\_Iframe\_head
- Resize\_Iframe\_onload

Include these snippets in an HTML s-control called "Resize\_Iframe\_sample." Put the first in the <head> section of the HTML and the second inside the <body> tag as part of the onload event.

To implement this, create the two snippets first. To create the Resize\_Iframe\_head snippet, use the following code:

```
<script type="text/javascript">

function resizeIframe() {
var me = window.name;
if (me) {
var iframes = parent.document.getElementsByName(me);
if (iframes && iframes.length == 1) {
var height = document.body.scrollHeight;
iframes[0].style.height = height + "px";
}
}
}

</script>
```

To create the Resize\_Iframe\_onload snippet, use the following code. The snippet only includes the function call to the code in the Resize\_iframe\_head snippet so that you have a single place to make a change in case the name of this function call changes. This creates a dependency between the code in Resize\_iframe\_head and Resize\_iframe\_onload rather than a dependency between the code in Resize\_iframe\_head and everywhere that Resize\_iframe\_head is included.

```
setTimeout(resizeIframe, 100)
```

Finally, include both snippets in an HTML s-control as shown in the example below. Embed this custom s-control on any detail page and you will see the resizing in action.

```
<html>
<head>

{!INCLUDE($SControl.Resize_Iframe_head)}

</head>

<body onload="{!INCLUDE($SControl.Resize_Iframe_onload)}" >
This is the content of the document.
<br><br><br><br><br><br><br><br><br><br><br><br><br>
...and this demonstrates what happens with many lines of text, with iframe resizing.
</body>

</html>
```

## Useful Custom Buttons and Links

User Permissions Needed	
To create or change custom buttons or links:	"Customize Application"
Custom buttons and links are available in: <b>All Editions</b>	
S-controls are available in: <b>Group, Professional, Enterprise, Unlimited, and Developer Editions</b>	

Use the following samples to get started using custom buttons and links:

- [Simple Custom Buttons](#)
- [Custom List Buttons](#)

## Simple Custom Buttons

### Displaying Alerts

You may want to display a simple alert window with text. In this example, the custom button displays a popup dialog with a welcome message containing the user's first name, using the `!$User.FirstName` merge field.

1. Define a custom button with the following attributes:

- `Display Type` is "Detail Page Button"
- `Behavior` is "Execute JavaScript"
- `Content Source` is "OnClick JavaScript"
- Use the following sample code:

```
alert ("Hello {!$User.FirstName}");
```

2. Add the button to the appropriate page layout.

### Getting Record IDs

You may want to define a custom button that opens a popup window to display a list of record IDs for the records a user has selected in a list. This is useful when testing to determine if you are getting the record IDs correctly before processing them with a more complex process.

1. Define a custom list button with the following attributes:

- `Display Type` is "List Button"



**Note:** The **Select Display Checkboxes (for Multi-Record Selection)** option automatically selected ensures that users can select any number of records in the list before clicking the button.

- `Behavior` is "Execute JavaScript"
- `Content Source` is "OnClick JavaScript"
- Use the following sample code:

```
idArray = {!GETRECORDIDS($ObjectType.Contact)};
alert("The Ids you have selected are: "+idArray);
```



**Note:** This example is for contacts. Change the object type for a different type of record.

2. Add the button to the appropriate related list on a page layout or list view layout.

## Custom List Buttons

### Mass Delete

Your busy users may want to delete more than one record in a list view or related list with a single click. Create the following custom list button and add it your activity related lists and list views:

1. Define a custom list button for events with the following attributes:

- Display Type is "List Button"



**Note:** The **Select Display Checkboxes (for Multi-Record Selection)** option automatically selected ensures that users can select any number of records in the list before clicking the button.

- Behavior is "Execute JavaScript"
- Content Source is "OnClick JavaScript"
- Use the following sample code:

```
{!REQUIRESRIPT("/soap/ajax/9.0/connection.js")}

var records = {!GETRECORDIDS( $ObjectType.Event )};
var taskRecords = {!GETRECORDIDS( $ObjectType.Task )};
records = records.concat(taskRecords);

if (records[0] == null) {
  alert("Please select at least one record.") }
else {

  var errors = [];
  var result = sforce.connection.deleteIds(records);
  if (result && result.length){
    var numFailed = 0;
    var numSucceeded = 0;
    for (var i = 0; i < result.length; i++){
      var res = result[i];
      if (res && res.success == 'true'){
        numSucceeded++;
      } else {
        var es = res.getArray("errors");
        if (es.length > 0) {
          errors.push(es[0].message);
        }
        numFailed++;
      }
    }
    if (numFailed > 0){
      alert("Failed: " + numFailed + "\nSucceeded: " + numSucceeded + " \n Due to: " +
        errors.join("\n"));
    } else {
      alert("Number of records deleted: " + numSucceeded);
    }
  }
  window.location.reload();
}
```

2. Add the custom list button to your activity list views.
3. Add the custom list button to any page layout that contains an activity related list. The custom button deletes any selected task or event in the list.

You can install this custom button and others like it by going to [www.salesforce.com/appexchange/](http://www.salesforce.com/appexchange/) and browsing or searching for the Mass Delete app.

### Passing Record IDs to an External System

If you use Salesforce record IDs as unique identifiers for integrating with an external system, you can configure a list button to pass those record IDs using a URL. In the example below, a list button calls a custom s-control to determine the record IDs of the records selected in a list and passes them in a URL query parameter to an external Web page called "www.yourwebsitehere.com."

1. Create a custom HTML s-control that uses the GETRECORDIDS function to retrieve a list of selected records. Use the following content for the s-control:

```
<script type="text/javascript">
idArray = {!GETRECORDIDS ($ObjectType.Account)};
window.location.href="http://www.yourwebsitehere.com?array="+idArray;
</script>
```



**Note:** Replace "www.yourwebsitehere.com" with your own URL.

2. Define a custom list button for accounts with the following attributes:

- Display Type is "List Button"



**Note:** The **Select Display Checkboxes (for Multi-Record Selection)** option automatically selected ensures that users can select any number of records in the list before clicking the button.

- Behavior is "Display in existing window with sidebar"
- Content Source is "Custom S-Control"
- Select the s-control you created in the first step.

3. Add the custom list button to the appropriate page layout or list view layout.

### Reopening Cases

Related lists are valuable because they allow users to perform an action on several records at once. Add a custom list button to your cases related lists so that users can reopen several cases on an opportunity at once.

1. Define a custom list button for cases with the following attributes:

- Display Type is "List Button"



**Note:** The **Select Display Checkboxes (for Multi-Record Selection)** option automatically selected ensures that users can select any number of records in the list before clicking the button.

- Behavior is "Execute JavaScript"
- Content Source is "OnClick JavaScript"
- Use the following sample code:

```
{!REQUIRESRIPT ("/soap/ajax/13.0/connection.js")}
var records = {!GETRECORDIDS ($ObjectType.Case)};
```

```

var newRecords = [];

if (records[0] == null)
{
    alert("Please select at least one row")
}
else
{
    for (var n=0; n<records.length; n++) {
        var c = new sforce.SObject("Case");
        c.id = records[n];
        c.Status = "New";
        newRecords.push(c);
    }

    result = sforce.connection.update(newRecords);

    window.location.reload();
}

```



**Note:** This example references the *AJAX Toolkit*, which is available if API access is enabled for your organization. For more information about the *AJAX Toolkit*, see <https://wiki.apexdevnet.com/index.php/API>.

2. Add the custom list button to your opportunity page layouts by editing the Cases related list.



**Tip:** Use this list button on any page layout that contains the cases related list, such as account or contact page layouts.



**Note:** Notice the check for `records[0] == null`, which displays a message to users when they do not select at least one record in the list.

## S-Control Terminology

Available in: **Group, Professional, Enterprise, Unlimited, and Developer** Editions

The following terminology is used for custom s-controls in Salesforce:

### Advanced Functions

Advanced functions are formula functions designed for use in custom buttons, links, and s-controls. For example, the **INCLUDE** advanced function returns the content from an s-control snippet. For more details on advanced functions, see "Operators and Functions" in the Salesforce online help.

### Global Variables

Global variables are special merge fields that you can use to reference data in your organization. For more information, see "Understanding Global Variables" in the Salesforce online help.

### S-Controls

S-controls are components that you set up and define in Salesforce to store your custom code. Use s-controls to create and display your custom data forms within Salesforce using components like custom links, Web tabs, or custom buttons. For example, you can define a custom s-control containing HTML or JavaScript and address merge fields to display a map of a contact's address.

**Snippet**

Snippets are s-controls that are designed to be included in other s-controls. Similar to a helper method that is used by other methods in a piece of code, a snippet allows you to maintain a single copy of HTML or JavaScript that you can reuse in multiple s-controls. For common uses of snippets, see [Useful S-Controls](#) on page 7.

# Index

## M

Mash-ups  
examples [3](#)

## S

S-controls  
examples [3](#)  
terminology [12](#)  
useful samples [3](#)

