

# **Programmatically Hiding and Displaying Panels Based on the Value of a Drop Down**



**Author: Kaz**  
**Director of Sales Engineering, Sugar CRM**

**March 21, 2008 Version 1.0**

Sugar Version: 5.0

# Cookbook - Programmatically Hiding and Displaying Panels Based on the Value of a Drop Down

## Overview

In many cases it is desirable to hide or show certain fields on a form based on the value of field. For example, if the account is a “partner” the form should show *partnerish* fields. The easiest way to do this is to create panels with information specific to each record type and then hide or show the panels based on the record type.

## Step 1 – Rearrange the Edit and Detail Views

Start with the **edit** view first. From Studio, open the account edit view and create 2 panels.

Name the first panel “CUSTOMER INFO” and the second panel “PARTNER INFO”. The naming of these is very important because they will be used in the JavaScript code to hide these panels. Drag some fields into both panels (See example below)

**Layout Preview**

The screenshot displays a 'Layout Preview' of a form with four distinct panels, each containing several input fields:

- ACCOUNT INFORMATION**: Contains fields for Name, Phone Office, Website, Fax, Ticker Symbol, Alternate Phone, Type, Team, and Assigned to.
- CUSTOMER INFO**: Contains fields for Member of, Employees, Industry, SIC Code, Ownership, and Rating.
- PARTNER INFO**: Contains fields for Annual Revenue and a placeholder '(filler)'. This panel is currently hidden.
- ADDRESS INFORMATION**: Contains fields for Billing Street and Shipping Street.

Repeat the process for the detail view **ensuring the panel names are the same.**

## Layout Preview

Name	Phone Office
Website	Fax
Ticker Symbol	Other Phone
Type	(filler)
Team	Date Modified
Assigned to	Date Entered
Billing Address	Shipping Address
Description	
Email	

### CUSTOMER

Member of	Employees
Ownership	Rating
Industry	SIC Code

### PARTNER

Annual Revenue	(filler)
----------------	----------

## Creating a Custom Edit Handler

Create a file called /modules/Accounts/views/view.edit.php. This file is a custom edit view handler for the accounts module. It will use all of the default behavior, but it will add some JavaScript code to add an event handler for the *onchange* event of the account type. The text s in red is what you would change for your specific field and panels.

Add the following code:

```
<?php
if(!defined('sugarEntry') || !sugarEntry) die('Not A Valid Entry Point');

/*****
 * Description: This file is used to override the default Meta-data EditView behavior
 * to provide customization specific to the Accounts module.
 * Portions created by SugarCRM are Copyright (C) SugarCRM, Inc.
 * All Rights Reserved.
 * Contributor(s): _____
 *****/

require_once('include/MVC/View/views/view.edit.php');
class AccountsViewEdit extends ViewEdit {
    function AccountsViewEdit(){
        parent::ViewEdit();
    }
    function display() {
        //Code to hide or display the panels when first editing the record
        $prePop = 'document.getElementById(\'account_type\').onchange();';

        $js=<<<EOQ
        <script>
        document.getElementById("account_type").onchange = function() {
            value = document.getElementsByName('account_type')[0].value;
            if(value=="Partner") {
                document.getElementById("PARTNER INFO").style.display="block";
                document.getElementById("CUSTOMER INFO").style.display="none";
            } else if (value == "Customer") {
                document.getElementById("CUSTOMER INFO").style.display="block";
                document.getElementById("PARTNER INFO") style.display="none";
            } else {
                document.getElementById("CUSTOMER INFO").style.display="none";
                document.getElementById("PARTNER INFO ").style.display="none";
            }
        };
        $prePop;
        </script>

        EOQ;
        parent::display();
        echo $js;
    }
}
?>
```

## Explaining the Code

```
//Code to hide or display the panels when first editing the record
$prePop = 'document.getElementById(\'account_type\').onchange();';
```

When the form first draws (as either a new record or editing an existing record) the correct panels should be hidden or displayed. The code above will call the onchange event for the account type so the form is initially rendered properly.

```
<script>
document.getElementById("account_type").onchange = function() {
value = document.getElementsByName('account_type')[0].value;
if(value=="Partner") {
    document.getElementById("PARTNER INFO").style.display="block";
    document.getElementById("CUSTOMER INFO").style.display="none";
} else if (value == "Customer") {
    document.getElementById("CUSTOMER INFO").style.display="block";
    document.getElementById("PARTNER INFO").style.display="none";
} else {
    document.getElementById("CUSTOMER INFO").style.display="none";
    document.getElementById("PARTNER INFO").style.display="none";
}
};
$prePop;
</script>
```

This code is pretty simple to follow. For the field `account_type` we are simply adding an onchange event. For those of you that are not experts in JavaScript, the `onChange` event does not have to be declared when the element is created. You can simply have code that declares the function later. Kind of nice.

Based on the value of the dropdown, we are simply hiding or showing tabs. `"PARTNER INFO"` and `"CUSTOMER INFO"` are the name of the panels that were created in Studio.

```
$this->ss->assign('HIDE_SHOW_PANELS_SCRIPT', $js);
parent::display();
echo $js;
```

This code puts the javascript after the panel is created.

## Creating a Custom Detail View Handler

Create a file called /modules/Accounts/views/view.detail.php. This file is a custom detail view handler for the accounts module. It is very similar to the code above, so lots of explanations are not required.

Add the following code:

```
<?php
if(!defined('sugarEntry') || !sugarEntry) die('Not A Valid Entry Point');
/*****

* Description: This file is used to override the default Meta-data Detail View behavior
* to provide customization specific to the Accounts module.
* Portions created by SugarCRM are Copyright (C) SugarCRM, Inc.
* All Rights Reserved.
* Contributor(s): _____

*****/

require_once('include/MVC/View/views/view.detail.php');
class AccountsViewDetail extends ViewDetail {
    function AccountsViewDetail(){
        parent::ViewDetail();
    }
    function display() {
        global $app_list_strings;
        $json = getJSONobj();
        $value = $json->encode($this->bean->account_type);
        $js=<<<EOQ
            <script>

                value = $value;
                if(value=="Partner") {
                    document.getElementById("PARTNER INFO").style.display="block";
                    document.getElementById("CUSTOMER INFO").style.display="none";
                } else if (value == "Customer") {
                    document.getElementById("CUSTOMER INFO").style.display="block";
                    document.getElementById("PARTNER INFO").style.display="none";
                } else {
                    document.getElementById("CUSTOMER INFO").style.display="none";
                    document.getElementById("PARTNER INFO").style.display="none";
                }
            </script>
        EOQ;
        parent::display();
        echo $js;
    }
}
?>
```

## Fixing a Minor Bug in the Detail View Container

In Edit Views, there is a div tag wrapped around panels. The div id is the name of the panel. In the detail view, the naming of the div tags is different. All the divs are named panel\_1. This is a known bug in the application. To fix the problem, change the file

include/DetailView/DetailView.tpl.

Around line 36 change the following line:

```
<div id='panel_{{ $panel_id }}'>
```

to

```
<div id='{{ $label }}'>
```

**This change is not upgrade safe. You are changing core code. This change will need to be reapplied when upgrading!!!!**