Wizdocs Forms Designer: Formula Tip sheet

A compilation of basic and advanced formulas to operate the Wizdocs Forms Designer

Updated August 2017



Wizdocs Forms Designer commonly used Formulas

# New Form Related List

|  |  |  |
| --- | --- | --- |
|  | Description | Example |
| Code for related new item list | Used to insert new record in a related item list and tie that item back to the master list. | $( document ).ready(function() {  //HIDE FIELDS ASSIGNED THE HIDER CSS CLASS BY DEFAULT  $('.hider').hide();  //ALWAYS SET TITLE TO VIEW ITEM SINCE IT IS NOT FILLED BY THE USER  fd.field('Title').control().value("View Item");  //SET VARIABLES FOR PARENT FORM TRIGGER  var parentID = fd.getSourceID();  //IF LOGIC FOR PARENT FORM TRIGGER  if (parentID) {    fd.field('CompanyLink').control().value(parentID);      }  //CLOSE DOCUMENT LOAD  }); |
| Quick edit related items new form | Used to enable quick edit for related list embedded in a master form. | //AUTOMATIC SET ON PAGE LOAD  fd.populateFieldsInGrid($('.related-items-company-locations'), {  CompanyLink: '{CurrentItem}',  Title: 'View Item',  }); |
| Conditional hide on checkbox | Used to show and hide classes based on checkbox. | $("input[id\*='Sales\_x0020\_and\_x0020\_distribut']").change(function(){  if($("input[id\*='Sales\_x0020\_and\_x0020\_distribut']").is(':checked'))  {  $('.saleshide').show();  }  else  {  $('.saleshide').hide();  fd.field('Sales\_x0020\_and\_x0020\_distribut').control().value('');  alert('Contents have been removed.');  }  });  if($("input[id\*='Sales\_x0020\_and\_x0020\_distribut']").is(':checked'))  {  $('.saleshide').show();  }  else  {  $('.saleshide').hide();  } |
| Conditional hide based on dropdown box | Used to show and hide classes based on dropdown selection box. | $("select[title='Acceleration']").change(function(){  var aceltarget = $("select[title='Acceleration']").val();    if(aceltarget == "None")  {  $('.accelamount').hide();    }  else  {  $('.accelamount').show();  }  });    if($("select[title='Acceleration']").val() == "None")  {  $('.accelamount').hide();  }  else  {  $('.accelamount').show();  } |
| NewForm refer to EditForm code | Used when you have a form that contains related lists. In the parent form, you have to create and save a new form so an "ID" number can be generated which will link the child record to the parent record. This code will create a new form record then refer the user to the edit form. | $('.hide').hide();  fd.onsubmit(function() {  var uri = fd.setUrlParam(decodeURIComponent(window.location.href), 'FDRedirectWithID', 'fd\_Major Issues Template\_Edit.aspx?ID=');  fd.sourceFormParam(uri);  return true;  }); |
| Accordion Keep Collapsed | By default, the Wizdocs Forms Designer will track which accordion tabs the user has open so that it may automatically open such tabs when the user returns. This behavior applies to even new forms.  Use the following code to remove the cookie and keep the accordion tabs expanded. | $(window).unload(function(){  var i = 0;  while($.removeCookie('fd\_accordion-'+i)) {  i++;  }  }); |
| Accordion Advance to Next on Button Push | This will open the next accordion in the sequence. Ensure that you are using "Single" or "SingleCollapsable" accordion.  Enter the code in the "OnClick" variable of the inserted button | var currentAccordion = $('#fd\_accordion-0').accordion( "option", "active");  $('#fd\_accordion-0').accordion( "option", "active", currentAccordion+1); |
| Accordion Previous on Button Push | This will open the next accordion in the sequence. Ensure that you are using "Single" or "SingleCollapsable" accordion.  Enter the code in the "OnClick" variable of the inserted button | var currentAccordion = $('#fd\_accordion-0').accordion( "option", "active");  $('#fd\_accordion-0').accordion( "option", "active", currentAccordion-1); |
| Tab Advance to Next on Button Push | This will open the next tab in the sequence.  Enter the code in the "OnClick" variable of the inserted button | var currentTab = fd.tabControl(0).tabs('option', 'active');  fd.tabControl(0).tabs('option', 'active', currentTab + 1); |
| Tab Previous on Button Push | This will open the previous tab in the sequence.  Enter the code in the "OnClick" variable of the inserted button | var currentTab = fd.tabControl(0).tabs('option', 'active');  fd.tabControl(0).tabs('option', 'active', currentTab - 1); |
| Save form (and do not close) | Used to add a save button to the form that will save its contents and then reload it. | 1. Create a button on the form. 2. In the "OnClick" event, accessible from the settings menu when you click the newly added button, enter the following code   fd.sourceFormParam(decodeURIComponent(window.location.href));return true; |
| XSLT for "Add New Item" within a related list | Used to add a related list but only show the "Add new item" option. | <View>  <ViewFields></ViewFields>  <ViewData />  <Query />  <Aggregations Value="Off" />  <RowLimit>1</RowLimit>  <Mobile MobileItemLimit="3" MobileSimpleViewField="LinkTitle" />  <XslLink Default="TRUE">main.xsl</XslLink>  <JSLink>clienttemplates.js</JSLink>  <Toolbar Type="Standard" />  <ParameterBindings>  <ParameterBinding Name="NoAnnouncements" Location="Resource(wss,noXinviewofY\_LIST)" />  <ParameterBinding Name="NoAnnouncementsHowTo" Location="Resource(wss,noXinviewofY\_DEFAULT)" />  </ParameterBindings>  </View> |

# Field Form Values (Getting and Setting)

|  |  |  |
| --- | --- | --- |
| **FIELD TYPE** | **GET / SET** | **SCRIPT** |
| Single Line of Text | Get | fd.field('Title').value(); |
| Set | fd.field('Title').value('Default title'); |
| OnChange | fd.field('Title').change(function(){}); |
| Multiline Plain Text Field | Get | fd.field('MultipleLinePlain').value(); |
| Set | fd.field('MultipleLinePlain')  .value('MultipleLinePlain'); |
| OnChange | fd.field('MultilinePlain').change(  function(){alert('Field has been changed!')}  ); |
| Multiline RTF Field | Get | fd.field('MultilineRichText').value(); |
| Set | fd.field('MultilineRichText')  .value('MultilineRichText'); |
| OnChange | fd.field('MultilineRichText').change(  function(){alert('Field has been changed!')}  ); |
| Multiline Enhanced Rich Text Field | Get | fd.field('MultilineEnhanced').value(); |
| Set | fd.field('MultilineEnhanced')  .value('MultilineEnhanced'); |
| OnChange | fd.field('MultilineEnhanced').change(  function(){alert('Field has been changed!')}  ); |
| Choice Single | Get | fd.field('ChoiceSingle').value(); |
| Set | fd.field('ChoiceSingle').value('B'); |
| OnChange | fd.field('Choice').change(function(){  alert('Field changed!');  }); |
| Choice Multiple | Get | var checkboxIndex = 2;  fd.field('MultiChoice').control().\_el()  .find('input[type="checkbox"]').eq(checkboxIndex)  .is(':checked'); |
| Set | var checkboxIndex = 2;  fd.field('MultiChoice').control().\_el()  .find('input[type="checkbox"]').eq(checkboxIndex)  .prop('checked', true); |
| OnChange | fd.field('MultiChoice').change(function(){  alert('Field changed!');  }); |
| Number | Get | fd.field('Number').value(); |
| Set | fd.field('Number').value('13'); |
| OnChange | fd.field('Number').change(function(){  alert('Field changed!');  }); |
| Currency | Get | fd.field('Currency').value(); |
| Set | fd.field('Currency').value('23'); |
| OnChange | fd.field('Currency').change(function(){  alert('Field changed!');  }); |
| Date | Get | fd.field('Date').value(); |
| Get Date-object | field('Date').control('getDate') // returns Date-object |
| Set | fd.field('Date').value('4/21/2012'); // by String  fd.field('DateTime')  .value(new Date()); // by Date-object |
| OnChange | fd.field('Date').change(function(){  alert('Field changed!');  }); |
| DateTime | Get | fd.field('DateTime').value()[0]; // date  fd.field('DateTime').value()[1]; // hours  fd.field('DateTime').value()[2]; // minutes |
| Get Date-object | field('DateTime').control('getDate') // returns Date-object |
| Set | fd.field('DateTime')  .value(['4/21/2012', '11 PM', '35']); // by Array  fd.field('DateTime')  .value(new Date()); // by Date-object |
| OnChange | fd.field('DateTime').change(function(){  alert('Field changed!');  }); |
| Hyperlink / Image | Get | fd.field('Hyperlink').value()[0]; // link  fd.field('Hyperlink').value()[1]; // descr |
| Set | fd.field('Hyperlink')  .value(['http://someurl', 'description']); |
| OnChange | fd.field('Hyperlink').change(function() {  alert('Changed!');  }); |
| Single Lookup Value | Get Text | fd.field('Lookup').control('getSelectedText'); |
| Get ID | fd.field('Lookup').value(); |
| Set Text | fd.field('Lookup').value('Text'); |
| Set ID | var ID = 4;  fd.field('Lookup').value(ID); |
| OnChange | fd.field('Lookup')  .change(function(){}); |
| Multi Lookup Values | Get All | var selectedItems = fd.field('LookupMultiple')  .control().\_el().find("select:eq(1) option");  var s = '';  for (var i = 0; i < selectedItems.length; i++) {  s += selectedItems.eq(i).text() + ';';  }  alert(s); |
| Get First | fd.field('MultiLookup').control().\_el()  .find("select:eq(1) option").eq(0).text() |
| Get Second | fd.field('MultiLookup').control().\_el()  .find("select:eq(1) option").eq(1).text() |
| Get ID | // first selected:  fd.field('MultiLookup').control().\_el()  .find('select:eq(1) option').eq(0).val();  // second selected:  fd.field('MultiLookup').control().\_el()  .find('select:eq(1) option').eq(1).val(); |
| Set | var ID = 3;  // select element by ID:  fd.field('MultiLookup').control().\_el()  .find('select:eq(0)').val(ID);  // add button click:  fd.field('MultiLookup').control().\_el()  .find('button:eq(0)').click(); |
| OnChange | fd.field('MultiLookup').change(function(){  alert('Field changed!');  }); |
| Boolean | Get | fd.field('Boolean').value(); |
| Set | fd.field('Boolean').value('1'); |
| OnChange | fd.field('Boolean').change(function(){  alert('Field changed!');  }); |
| Client People Picker | Get All Logins | fd.field('User').control('ready', function() {  var selectedUsers = fd.field('User').value();  for (var i = 0; i < selectedUsers.length; i++) {  alert('Login: ' + selectedUsers[i].Key);  }  }); |
| Set | // assign value by a display name  fd.field('User').value('Tim Watts');  // by a login  fd.field('User').value('DEV\\ann');  // or by an e-mail:  fd.field('User').value('AGibson@contoso.com'); |
| OnChange | fd.field('User').change(function(){  alert('Field changed!');  }); |
| Server People Picker with multiple choice | Get All Logins | var logins = '';  $.each(fd.field('People').value()  .dictionaryEntries,  function() {  logins += this.AccountName + '; ';  }); |
| Set | fd.field('People').value('DOMAIN\\login'); |
| OnChange | fd.field('People').change(function(){  alert('Field changed!');  }); |
| Server People Picker with single choice | Get | fd.field('PersonGroupSingle')  .value().dictionaryEntries[0].AccountName |
| Set | fd.field('Person').value('DOMAIN\\login'); |
| OnChange | fd.field('Person').change(function(){  alert('Field changed!');  }); |
| External Data | Get | Returns field value of External List item:  fd.field('ExternalData').value()  .dictionaryEntries[0]['FieldOfExternalList']  ‘ExternalData’ is an internal name of the lookup column.  ‘FieldNameOfExternalList’ is an internal name of the external list column.  If you include additional columns to the target list in lookup settings you will be able to get them dynamically as well:  fd.field('ExternalData').value()  .dictionaryEntries[0]['AdditionalFieldOfExternalList']  ‘AdditionalFieldOfExternalList’ is an internal name of the external list column which has been added as additional field in lookup settings. |
| Set | fd.field('ExternalData')  .value({key: '\_\_bg40003300', text: 'Item 1'});  **key** is BDC Identity of the selected item. **text** is a display name of the selected item. |
| OnChange | fd.field('ExternalData').change(function(){  alert('Field changed!');  }); |