CLIENT SCRIPT

A client-side script that runs JavaScript on the client (web browser) when client-based events occur.

Client scripts are used to customize the behavior of a form or other UI element.

They can be triggered by a user action such as clicking a button or by an event such as the loading of a page.

Some use cases for client scripts include:

- Placing the cursor in a form field on form load
- Generating alerts, confirmations, and messages
- Populating a form field in response to another field's value
- Highlighting a form field
- Validating form data
- Modifying choice list options
- Making a field read only, hidden, or mandatory
- Displaying informational messages
- Generating pop ups on button click or on page load

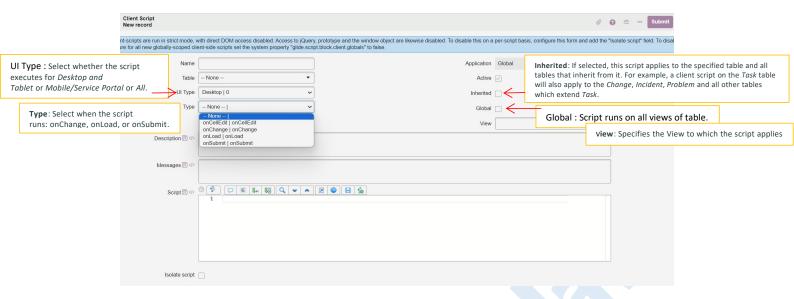
Note *** Client scripts are executed by the browser. It is important to make sure any DOM calls are as technology agnostic as possible.

Where client scripts run

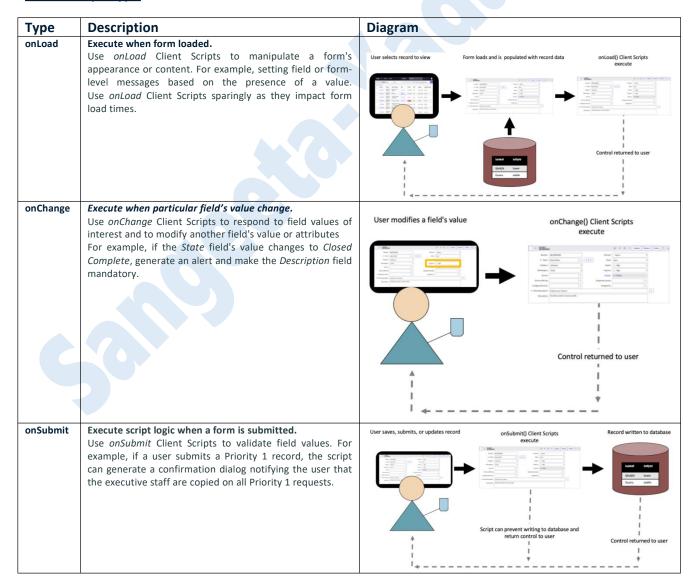
With the exception of onCellEdit() client scripts, client scripts only apply to forms and search pages. If you create a client script to control field values on a form, you must use one of these other methods to control field values when on a list.

- Create an access control to restrict who can edit field values.
- Create a business rule to validate content.
- Create a data policy to validate content.
- Create an onCellEdit() client script to validate content.
- Disable list editing for the table.

Note: Client scripts are not supported on ServiceNow mobile applications.



Client Script Type



THE SCRIPT FIELD

onLoad Script Template

onSubmit Script Template

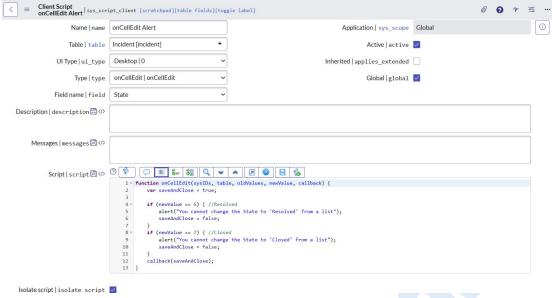
onChange Script Template

- control: the DHTML (Dynamic Hyper Text Markup Language) widget whose value changed.
- oldValue : value of the field when the form loaded and prior to the change.
- newValue : value of the field after the change.
- isLoading: boolean value indicating whether the change is occurring as part of a form load. Value is true if change is due to a form load. When forms load, all the field values on the form change as the record is loaded into the form.
- isTemplate: boolean value indicating whether the change occurred due to population of the field by a template. Value is true if change is due to population from a template.

onCellEdit Script Template

- sysIDs: an array of the sys ids for all items parameters
- table : the table of the items being edited.
- oldValues : the old value for the cells being edited.
- newValues : the new value for the cells being edited.
- callback: a callback that continues the execution of any other related cell edit scripts. If true is passed as a parameter, the other scripts are executed or the change is committed if there are no more scripts. If false is passed as a parameter, any further scripts are not executed and the change is not committed.
- Note: onCellEdit() client script cann't apply to dashboard list widgets.

Example of onCellEdit() Client Script:



Calling in Client Script

```
Client Side Script
Server Side Script
Business Rule
                                                                                 Using: g_scratchpad
(function executeRule(current, previous /*null when async*/) {
                                                                                 function onLoad() {
  var gr = new GlideRecord("problem");
                                                                                  //Type appropriate comment here, and begin script below
  if (gr.get(current.parent)) {
                                                                                  if (g_scratchpad.parent) {
    g_scratchpad.parent = current.parent;
                                                                                    g_form.setValue("short_description", g_scratchpad.short_description);
    g_scratchpad.short_description = gr.short_description;
                                                                                     g_form.setValue("assignment_group", g_scratchpad.assignment_group);
    g_scratchpad.assigned_to = gr.assigned_to;
                                                                                     g_form.setValue("assigned_to", g_scratchpad.assigned_to);
    g_scratchpad.assignment_group = gr.assignment_group;
                                                                                }
})(current, previous);
Script Include
                                                                                 Using: GlideAjax (*** this is catalog client script)
var listCollectorUseCase = Class.create();
                                                                                 function on Change (control, old Value, new Value, is Loading) {
listCollectorUseCase.prototype = Object.extendsObject(AbstractAjaxProcessor,
                                                                                  if (isLoading | | newValue == ") {
                                                                                     return:
                                                                                  }
  listCollectorUser: function() {
                                                                                           g_form.addInfoMessage(g_form.getValue("trainees_users"));
    var traineesUser = this.getParameter('Trainees'); //Parameter passed at
                                                                                  //Type appropriate comment here, and begin script below
                                                                                  var rec = new GlideAjax('listCollectorUseCase');
Client Side 'Trainees'
    var traineeUsr = traineesUser.split(',');
                                                                                  rec.addParam('sysparm name', 'listCollectorUser');
    var traineesRecord = [];
                                                                                  rec.addParam('Trainees', g_form.getValue("trainees_users"));
                                                                                   rec.getXMLAnswer(response);
    for (var i = 0; i < traineeUsr.length; i++) {
      var userRec = new GlideRecord('sys_user');
                                                                                   function response(result) {
      userRec.addActiveQuery();
                                                                                     alert('Check Result ' + result);
      userRec.addQuery('sys_id', traineeUsr[i])
                                                                                     g_form.setValue('trainees_email_id', result);
      userRec.query();
      while (userRec.next()) {
                                                                                }
         traineesRecord.push(userRec.email);
    return traineesRecord.toString();
  type: 'listCollectorUseCase '
});
```

