

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
New record

nt-scripts are run in strict mode, with direct DOM access disabled. Access to jQuery, prototype and the window object are likewise disabled. To disable this on a per-script basis, configure this form and add the "Isolate script" field. To disable for all new globally-scoped client-side scripts set the system property "glide.script.block.client.globals" to false.

Name:

Table:

UI Type:

Type:

- None --
- onCellEdit | onCellEdit
- onChange | onChange
- onLoad | onLoad
- onSubmit | onSubmit

Description:

Messages:

Script:

Isolate script: ☐

Application:

Active: ☒

Inherited: ☐

Global: ☐

View:

Submit

UI Type : Select whether the script executes for *Desktop and Tablet* or *Mobile/Service Portal* or *All*.

Type: Select when the script runs: *onChange*, *onLoad*, or *onSubmit*.

Inherited: If selected, this script applies to the specified table and all tables that inherit from it. For example, a client script on the *Task* table will also apply to the *Change*, *Incident*, *Problem* and all other tables which extend *Task*.

Global : Script runs on all views of table.















view: Specifies the View to which the script applies

Client Script Type

Type	Description	Diagram
onLoad	<p>Execute when form loaded.</p> <p>Use <i>onLoad</i> Client Scripts to manipulate a form's appearance or content. For example, setting field or form-level messages based on the presence of a value. Use <i>onLoad</i> Client Scripts sparingly as they impact form load times.</p>	<p>User selects record to view</p> <p>Form loads and is populated with record data</p> <p>onLoad() Client Scripts execute</p> <p>Control returned to user</p>
onChange	<p>Execute when particular field's value change.</p> <p>Use <i>onChange</i> Client Scripts to respond to field values of interest and to modify another field's value or attributes. For example, if the <i>State</i> field's value changes to <i>Closed Complete</i>, generate an alert and make the <i>Description</i> field mandatory.</p>	<p>User modifies a field's value</p> <p>onChange() Client Scripts execute</p> <p>Control returned to user</p>
onSubmit	<p>Execute script logic when a form is submitted.</p> <p>Use <i>onSubmit</i> Client Scripts to validate field values. For example, if a user submits a <i>Priority 1</i> record, the script can generate a confirmation dialog notifying the user that the executive staff are copied on all <i>Priority 1</i> requests.</p>	<p>User saves, submits, or updates record</p> <p>onSubmit() Client Scripts execute</p> <p>Script can prevent writing to database and return control to user</p> <p>Record written to database</p> <p>Control returned to user</p>















THE SCRIPT FIELD

onLoad Script Template

```
Script | script              
```















```
1 function onLoad() {  
2     //Type appropriate comment here, and begin script below  
3  
4 }
```

onSubmit Script Template

```
Script | script              
```

```
1 function onSubmit() {  
2     //Type appropriate comment here, and begin script below  
3  
4 }
```















onChange Script Template

```
Script | script              
```

```
1 function onChange(control, oldValue, newValue, isLoading, isTemplate) {  
2     if (isLoading || newValue === '') {  
3         return;  
4     }  
5  
6     //Type appropriate comment here, and begin script below  
7  
8 }
```

- 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

```
Script | script              
```

```
1 function onCellEdit(sysIDs, table, oldValues, newValue, callback) {  
2     var saveAndClose = true;  
3     //Type appropriate comment here, and begin script below  
4  
5     callback(saveAndClose);  
6 }
```

- 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 can't apply to dashboard list widgets.

Example of onCellEdit() Client Script :

Client Script

onCellEdit Alert

sys_script_client [scratchpad][table fields][toggle label]

Name | name

onCellEdit Alert

Application | sys_scope

Global

Table | table

Incident [incident]

Active | active

☒

UI Type | ui_type

Desktop | 0

Inherited | applies_extended

☐

Type | type

onCellEdit | onCellEdit

Global | global

☒

Field name | field

State

Description | description

Messages | messages

Script | script

```

1 * function onCellEdit(sysIDs, table, oldValues, newValue, callback) {
2   var saveAndClose = true;
3
4   if (newValue == 6) { //Resolved
5     alert("You cannot change the State to 'Resolved' from a list");
6     saveAndClose = false;
7   }
8   if (newValue == 7) { //Closed
9     alert("You cannot change the State to 'Closed' from a list");
10    saveAndClose = false;
11  }
12  callback(saveAndClose);
13 }

```

Isolate script | isolate script

☒

Calling in Client Script

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

onSubmit Client Script

```
function onSubmit() {  
    var totalCost = 10;  
    updateGrandTotal(totalCost);  
}
```

onLoad Client Script

```
function updateGrandTotal(totalCost) {  
    g_form.setValue('grand_total', totalCost);  
}  
  
function onLoad() {  
  
}
```

Sangeeta-Yadav