

Client-side scripts execute within a user's browser and are used to manage forms and form fields.

Examples of things client-side scripts can do include:

- Place the cursor in a form field on form load
- Generate alerts, confirmations, and messages
- Populate a form field in response to another field's value
- Highlight a form field
- Validate form data
- Modify choice list options
- Hide/Show fields or sections

The screenshot shows a web application interface for a form titled "NeedIt New record [User view]". The form contains several fields: "Number" with the value "NI002001", "Requested for" (marked with a red asterisk), "Short description" (highlighted with a yellow box), and "Description". A modal dialog box is displayed in the center, asking for confirmation: "All requests are subject to management review. Do you want to continue?". The dialog has "Cancel" and "OK" buttons. The "Short description" field is highlighted with a yellow box, and the dialog box is also highlighted with a yellow box. The "Requested for" field is marked with a red asterisk, and the "Short description" field is highlighted with a yellow box.

Client Script Types

A Client Script executes client-side script logic when forms are:

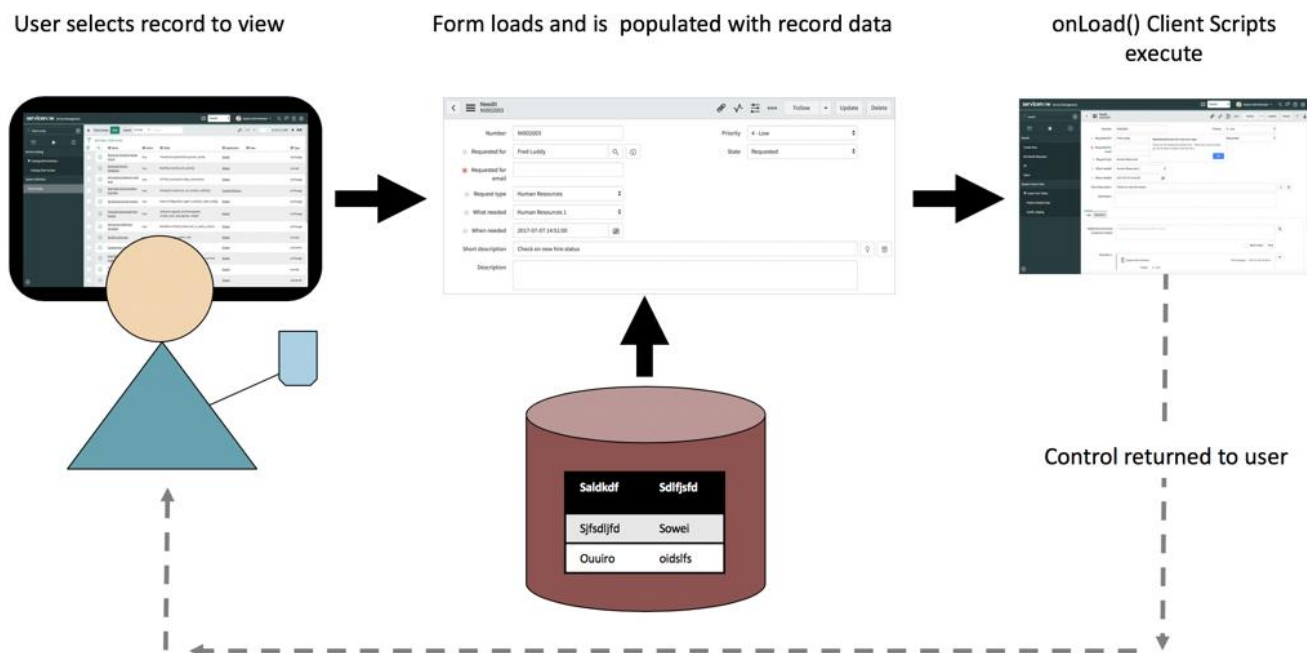
- Loaded
- Changed
- Submitted

onLoad

onLoad executes when forms are loaded. Use onLoad 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.

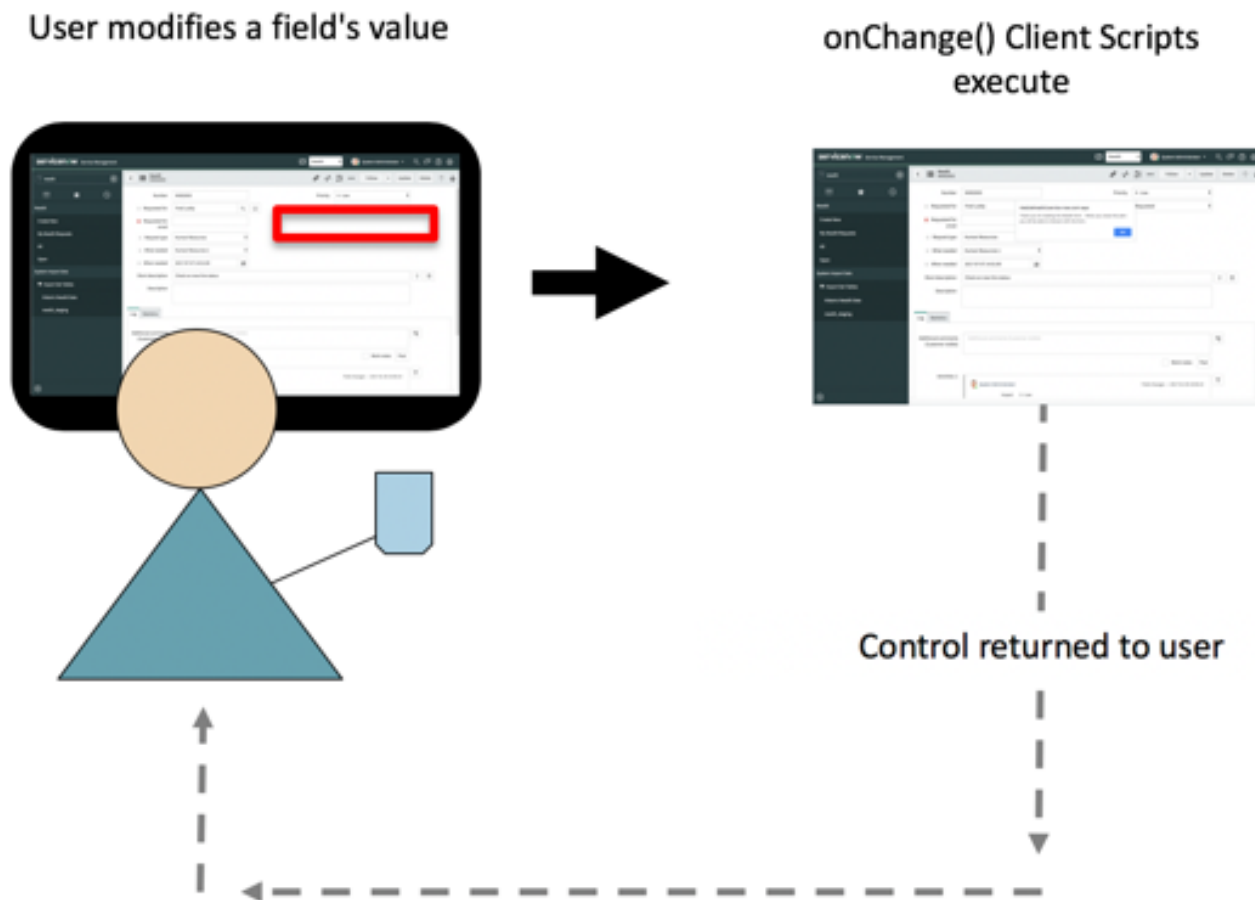
Use onLoad Client Scripts sparingly as they impact form load times.



onChange

onChange executes when a field's value changes. Use onChange Client Scripts to respond to field values of interest and to modify another field's value or attributes.

For example, if the State field's value changes to Closed Complete, generate an alert and make the Description field mandatory.



onSubmit

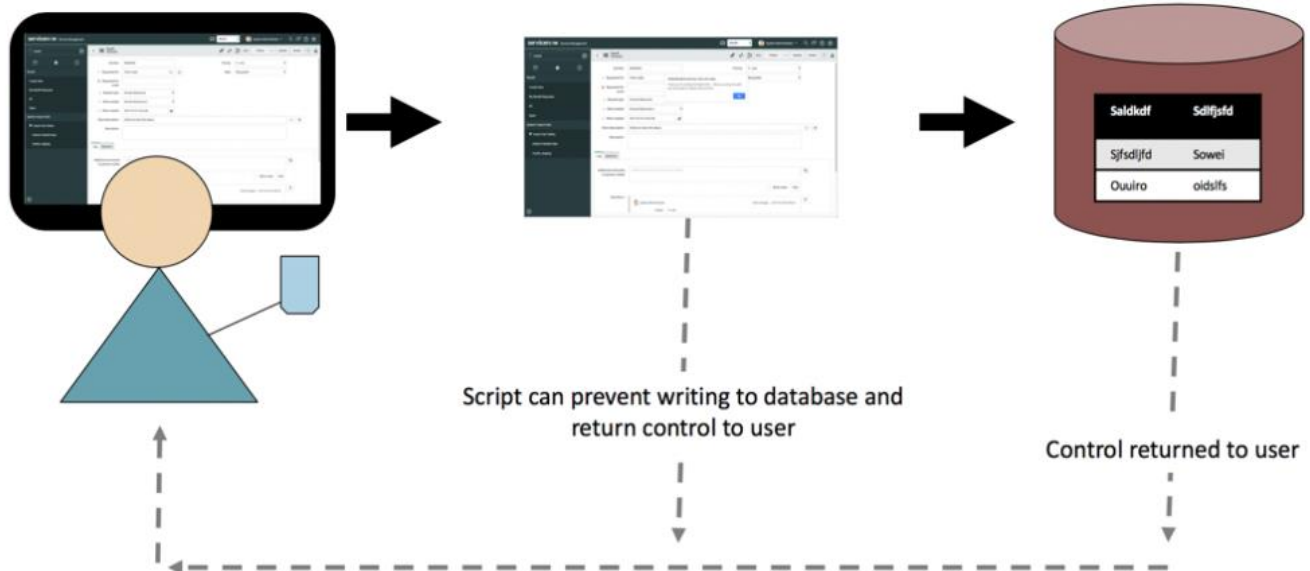
onSubmit executes when a form is submitted. Use onSubmit Client Scripts to validate field values.

For example, if a user submits a Priority 1 record, the script can generate a confirmation dialog notifying the user that the executive staff are copied on all Priority 1 requests.

User saves, submits, or updates record

onSubmit() Client Scripts
execute

Record written to database



Creating Client Scripts

The procedure for adding files to an application in Studio is the same regardless of file type:




- 1. Click the **Create Application File** link.
- 2. Choose the new file type, in this case, **Client Script**.
- 3. Configure the new file.

Configuring the Client Script

As with any script, the configuration tells the script when to execute. The Client Script configuration options are:

- **Name:** Name of Client Script. Use a standard naming scheme to identify custom scripts.
- **Table:** Table to which the script applies.
- **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*.
- **Field Name:** Used only if the script responds to a field value change (*onChange*); name of the field to which the script applies.
- **Active:** Controls whether the script is enabled. Inactive scripts do not execute.
- **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:** If *Global* is selected the script applies to all Views. If the *Global* field is not selected you must specify the View.
- **View:** Specifies the View to which the script applies. The *View* field is visible when *Global* is not selected. A script can only act on fields that are part of the selected form View. If the *View* field is blank the script applies to the *Default* view.

Client Script
New record



Name

NeedIt Set Requested for

Table

NeedIt [x_58872_needit_needit]

UI Type

Desktop

Type

onChange


Field name

Active

Description

Application

NeedIt



Active

☒

Inherited

☐

Global

☐

View

User

Submit

The Script Field

When the type value is set, a script template is automatically inserted into the *Script* field.

onLoad Script Template

This is the onLoad script template:

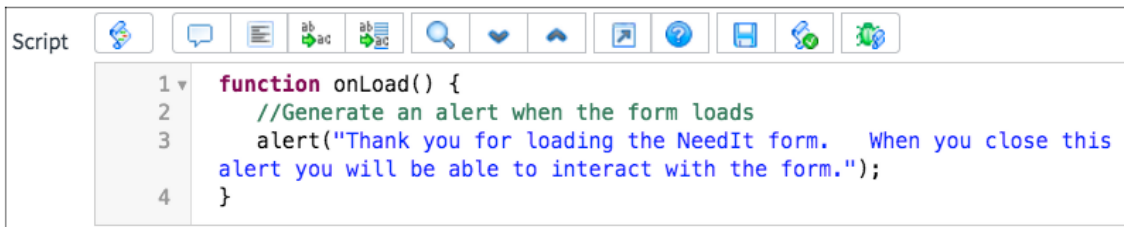


The screenshot shows a 'Script' field editor with a toolbar at the top containing icons for undo, redo, comment, list, code, search, heart, share, link, help, save, and test. The code area contains the following template:

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

The *onLoad* function has no arguments passed to it. As indicated by the comment, add your script logic in the *onLoad* function. It is a best practice to document your code so include comments to explain what the script does. Your future self will thank you for the clearly documented script.

This example script generates an alert when a user requests a form load for a record. The user cannot interact with a form until *onLoad* Client Scripts complete execution.



The screenshot shows the same 'Script' field editor with the following example script:

```
1 function onLoad() {  
2     //Generate an alert when the form loads  
3     alert("Thank you for loading the NeedIt form.  When you close this  
4     alert you will be able to interact with the form.");  
5 }
```

onSubmit Script Template


This is the onSubmit script template:

Script 

```
1  function onSubmit() {  
2      //Type appropriate comment here, and begin script below  
3  
4  
5  }
```

The *onSubmit* function has no arguments passed to it. As indicated by the comment, add your script logic in the *onSubmit* function.

This example script generates an alert when a user saves a *NeedIt* record. The record is not submitted to the server until the onSubmit Client Scripts complete execution and return true.

Script 

```
1  function onSubmit() {  
2      //Generate an alert to thank the user for submitting a record  
3      alert("Thank you for submitting a NeedIt request.");  
4  }
```


onChange Script Template

This is the onChange Script template:



```
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  }
```

The *onChange* function is automatically passed five parameters by ServiceNow. Although you do not need to do anything to pass the parameters, you can use them in your script.

- **control:** field the Client Script is configured for.
- **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.

When a user selects a record to load, the form and form layout are rendered first and then the fields are populated with values from the database. From the technical perspective, all field values are changed, from nothing to the record's values, when a form loads. The *if* statement in the template assumes that onChange Client scripts should not execute their script logic when a form loads. The onChange Client Script also stops execution if the *newValue* for a field is no value. Depending on your use case, you can modify or even remove the *if* statement. For example:

```
//Stop script execution if the field value change was caused by a Template
if(isLoading || newValue === '' || isTemplate) {
    return;
}
```

For this example, the onChange Client Script executes because of changes to the *Short description* field value:

Type	onChange
Field name	Short description

When a user changes the value of the *Short description* field *on the form* the onChange script logic executes. The example generates an alert stating that the value of the *Short description* field has changed from the value the field had when the form loaded to the new value on the form.

Script

```
1 function onChange(control, oldValue, newValue, isLoading, isTemplate) {  
2   if (isLoading || newValue === '') {  
3     return;  
4   }  
5  
6   //Generate an alert showing the oldValue and newValue  
7   alert("You have changed the Short description from " + oldValue + " to  
8   " + newValue + ".");  
}
```

The value of the *Short description* field has changed *only* on the form. Changes are not sent to the database until a user saves, updates, or submits the record.

It is important to know that the value of *oldValue* is set *when the form loads*. Regardless of how many times the value of the *Short description* field changes, *oldValue* remains the same until the form is re-loaded from the database.

- Form loads:
 - oldValue* = hello
 - newValue* has no value
- User changes the value in the *Short description* field to bye:
 - oldValue* = hello
 - newValue* = bye
- User changes the value in the *Short description* field to greetings:
 - oldValue* = hello
 - newValue* = greetings
- User saves the form and reloads the form page:
 - oldValue* = greetings
 - newValue* has no value