SERVICENOW(WEEK-4)

🡪**SCRIPTING IN ServiceNow**:

Scripting on the ServiceNow platform involves using JavaScript to customize and extend the functionalities of the ServiceNow environment. This can be done in several ways, They are:

\*\*\***Client Scripts**: Execute on the user's browser to manage the behaviour of forms and fields in real-time, such as form validation or dynamic field updates.

[Some of the types included in Client Scripts are],

 **onLoad()**: Executes when a form loads. It is useful for initializing form elements or setting default values.

 **onChange()**: Triggered when a field value changes. It is ideal for dynamically adjusting other fields or form sections.

 **onSubmit()**: Runs when a form is submitted. It is commonly used for validation or to perform actions before saving the record.

 **onCellEdit()**: Executes when a cell in a list is edited. It is useful for list-level interactions.

[Small example for client side scripting,

function onLoad()

{

if (g\_form.getValue('category') === 'Hardware') { g\_form.setValue('sub category', 'Laptop');}

}

\*\*\***Server Scripts**: Run on the server-side to handle data processing, business logic, and interactions with the database. (Examples: Business Rules, Script Includes etc)

[Some of the types included in Server Scripts are],

 **Business Rules**: Automate tasks and enforce business logic on record operations (create, update, delete). They can be configured to run before or after a record operation.

[There are two categories in Business Rules: Before Business rule and After Business rule]

 **Script Includes**: Contain reusable JavaScript functions and are invoked by other server-side scripts. They help to keep the code maintainable.

 **Scheduled Jobs**: Run scripts at specified intervals for tasks like data clean-up or periodic notifications.

 **GlideAjax**: Allows server-side scripts to communicate with client-side scripts. It is useful for fetching server-side data without reloading the form.

[Example: Automatically assigning a task to a specific user based on the task's category]

\*\*\***UI Actions**: Customize buttons, links or context menu items on forms and lists.

[Some of the types included in UI Actions are],

 **Form Buttons**: Added to forms for specific actions like saving or executing custom scripts.

 **List Buttons**: Added to list views for bulk actions or custom interactions.

 **Context Menu Items**: Available in right-click menus on list items or forms.

[example: Adding a custom button on an incident form to trigger a workflow]

\*\*\***Workflow Scripts**: Integrate and customize workflows within the ServiceNow orchestration engine.

[There are two types included in **Workflow** Scripts, They are],

 **Workflow Activities**: Custom scripts within a workflow to perform specific actions or decisions based on workflow state.

 **Scripted Workflows**: Use JavaScript to define custom logic within workflows, enabling advanced automation and integration scenarios.

[Example: Creating a workflow that automates the approval process for service requests.]

\*\*\* **Scheduled Jobs**: Execute scripts at scheduled intervals for tasks like data clean up or periodic updates.

[There are two types included in **Scheduled Jobs, They** are],

 **Scheduled Script Executions**: Run server-side scripts on a predefined schedule (eg, daily, weekly etc).

 **Cron Jobs**: Manage and schedule tasks that require periodic execution such as data aggregation or system maintenance.

[Example: Periodically checking and updating records, generating reports etc]

\*\*\***Data Policies:** Enforce data consistency and integrity on forms.

[There are two types included in **Scheduled Jobs, They** are],

 **Mandatory Data Policies**: Ensure certain fields are filled before a record can be saved.

 **Read-Only Data Policies**: Control the visibility and editability of fields based on conditions.

{There are other Scripting techniques like Transform Maps, Practices etc}

[The last part of 4th week hands-on is already done in both Week-1 and Week-2]