Digital Nurture 3.0

ServiceNow

Week 4

Module 4 Report

Topic : ServiceNow Scripting Fundamentals and Functions

Name: P Vainavi

Superset ID: 5097830

College: Dayananda Sagar Academy of Technology and Management

USN: 1DT21IS100

Date:16/09/2024

ServiceNow Scripting Tutorials | Scripting in ServiceNow | ServiceNow Scripting Full Course

Introduction to ServiceNow Scripting:

- Definition: ServiceNow scripting involves writing and executing code on the ServiceNow platform to extend its capabilities, automate tasks, and customize functionalities. It leverages JavaScript for both client-side and server-side scripting.
- Key Areas:
  - Client-Side Scripting: Runs in the user's browser to manage form behavior, field validation, and dynamic interactions.
  - Server-Side Scripting: Runs on the ServiceNow server to handle data operations, business logic, and integrations.

ServiceNow Scripting Tutorials:

1. Client Scripts:
   - Purpose: Manage the behavior of forms and fields in real-time on the client side. They enhance user experience by providing immediate feedback and dynamically updating the form.
   - Types:
     - onLoad: Executes when a form is loaded, used for initial setup or displaying data.

```
function onLoad() {
  g_form.setValue('short_description', 'Default description');
}
```

- onChange: Executes when a field value changes, used to trigger actions based on user input.

```
function onChange(control, oldValue, newValue, isLoading) {
  if (isLoading || newValue == '') return;
  g_form.setValue('priority', 'High');
}
```

  - onSubmit: Executes before a form is submitted, used for validation or additional processing.

```
function onSubmit() {
  if (g_form.getValue('short_description') == '') {
    g_form.addErrorMessage('Short description is required');
    return false;
  }
  return true;
}
```

- onCellEdit: Executes when a cell in a list is edited, used for custom actions on list edits.

2. Business Rules:
  - Purpose: Automate server-side operations in response to database actions such as insertions, updates, or deletions. They ensure business logic is consistently applied.
  - Types:
    - Before: Executes before the record is saved to the database, used for validation or modification.

```
(function executeRule(current, previous /*null when async*/) {
  if (current.priority == 'Critical') {
    current.urgency = 1;
  }
})(current, previous);
```

- After: Executes after the record is saved, used for notifications or further processing.

```
(function executeRule(current, previous /*null when async*/) {
  gs.eventQueue('incident.updated', current, current.sys_id, gs.getUserID());
})(current, previous);
```

- Async: Executes asynchronously to handle long-running tasks or background processing.

```
(function executeRule(current, previous /*null when async*/) {
  gs.info('Async business rule executed');
})(current, previous);
```

3. Script Includes:

- Purpose: Define reusable server-side JavaScript that can be called from other scripts or modules. They promote code reusability and modularization.

```
var UserValidator = Class.create();
UserValidator.prototype = {
  initialize: function() {},
  validateEmail: function(email) {
    var regex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
    return regex.test(email);
  },
  type: 'UserValidator'
};
```

- Usage:

- Call from Business Rules or Script Includes:

```
var validator = new UserValidator();
var isValid = validator.validateEmail('test@example.com');
```

4. UI Actions:

- Purpose: Customize UI elements such as buttons, links, and context menus to perform custom actions.
- Types:
  - Button: Adds custom buttons to forms or lists to trigger specific actions.

```
function customButtonAction() {
  gs.addInfoMessage('Custom button clicked!');
}
```

- Link: Adds links to forms or lists for additional functionality.
  - Context Menu: Adds custom options to context menus for records.

5. Scheduled Jobs:

- Purpose: Automate background tasks to run at specified intervals or times.
- Example:

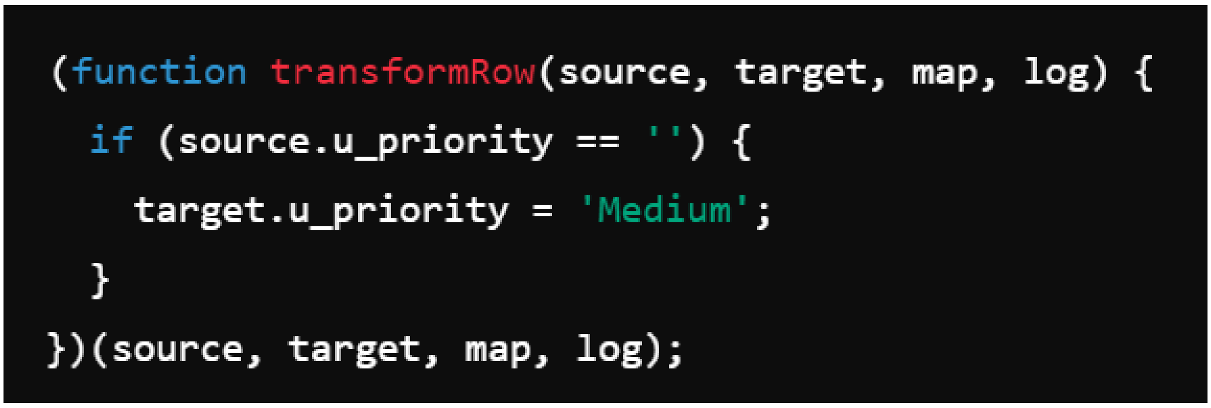
```
(function runScheduledJob() {
  var gr = new GlideRecord('incident');
  gr.addQuery('state', 'Closed');
  gr.addQuery('close_date', '<', gs.daysAgo(30));
  gr.deleteMultiple();
})();
```

- Use Cases:
  - Data Cleanup: Regularly clean up outdated or unnecessary records.
  - Report Generation: Automatically generate and distribute reports.

6. Transform Maps:

  - Purpose: Map data from import sets to ServiceNow tables, transforming and validating data during import.

  - Example:



```
(function transformRow(source, target, map, log) {
  if (source.u_priority == '') {
    target.u_priority = 'Medium';
  }
})(source, target, map, log);
```

- Usage:

    - Import Data: Import and transform data from external sources into ServiceNow tables.

ServiceNow Tutorial for Beginners

Understanding How ServiceNow Functions:

Platform Basics:
- Architecture:
  - Cloud-Based: ServiceNow is a cloud-native platform, ensuring scalability and accessibility.
  - Multi-Tenancy: A single instance of ServiceNow can serve multiple customers, with each customer’s data being isolated.
- Core Modules:
  - Incident Management: Manages incidents from creation through resolution, ensuring service continuity.
  - Problem Management: Addresses the root causes of incidents to prevent future occurrences.
  - Change Management: Manages changes to IT systems to minimize risk and disruption.

User Interface:
- Navigation: Users interact with ServiceNow through a web-based interface. Key elements include:
  - Application Navigator: Provides access to various applications and modules.
  - Lists and Forms: Displays and manages records in tables.
  - Dashboards: Visualize key metrics and performance indicators.

How to Properly Configure and Personalize the Platform:

Configuration:
- System Properties:
  - Purpose: Control various platform settings, such as email configurations and user preferences.
  - Access: Navigate to System Properties to view and modify settings.
- Application Settings:
  - Purpose: Customize application behavior, such as setting default values and enabling features.
  - Examples: Configure application-specific options like SLA definitions or approval rules.

Personalization:

- Forms and Fields:
  - Customizing Forms: Add, remove, or modify fields to capture relevant information.
    - Example: Add a custom field to the incident form to capture additional details.
  - Section Management: Organize fields into sections for better user experience.
- Themes and Branding:
  - Custom Branding: Apply company logos, color schemes, and other branding elements to the ServiceNow interface.
    - Access: Navigate to System Properties > UI Properties to modify branding settings.

Incident Module:

- Functionality:
  - Incident Creation: Incidents can be created manually by users or automatically through integrations.
  - Incident Management: Includes processes for assignment, escalation, resolution, and closure.
- Features:
  - Service Desk Integration: Allows users to report incidents via a service desk or portal.
  - SLAs: Define service level agreements to ensure timely resolution.

Problem Module:

- Functionality:
  - Problem Management: Focuses on identifying and managing the root causes of incidents.
  - Known Errors: Tracks known errors and workarounds to improve service reliability.
- Features:
  - Problem Records: Manage problems and link them to related incidents.
  - Root Cause Analysis: Tools for analyzing and resolving underlying issues.

Change Module:

- Functionality:
  - Change Management: Oversees the process of making changes to the IT environment.
  - Change Requests: Create, review, and approve change requests to ensure smooth transitions.

- Features:
  - Change Approvals: Automated approval workflows to manage change requests.
  - Change Implementation: Tools for planning, scheduling, and executing changes.

List and Forms:

Lists:

- Overview:
  - Data Display: Lists display records from ServiceNow tables in a tabular format, allowing for sorting, filtering, and searching.
- Customization:
  - List Layouts: Customize which columns are displayed and their order.
  - Filters: Create and apply filters to view specific subsets of data.

Forms:

- Overview:
  - Data Entry: Forms are used for creating and editing records. They include fields for capturing data and sections for organization.
- Customization:
  - Form Layouts: Modify the arrangement of fields and sections to suit business needs.
  - UI Policies: Control form behavior based on user interactions or data values.