**🌐 ServiceNow Client-Side Scripting – Beginner Notes**

**🧠 What is Client-Side Scripting?**

Client-side scripting in ServiceNow runs in the **user's browser**, meaning the code is executed before the data hits the server. It's used to enhance user experience by validating, modifying, or reacting to data in real-time on forms and lists.

**🎯 Key Components of Client-Side Logic**

**🔹 1. UI Policy**

* **Purpose:** Dynamically change form behaviour (show/hide fields, make fields mandatory or read-only).
* **Runs on:** Form (client-side)
* **Use Cases:**
  + Hide the "Resolution Notes" field unless the incident is resolved.
  + Make "Short Description" mandatory if the category is "Software".

🛠️ *No scripting required* — logic is set via the UI.

**🔹 2. Data Policy**

* **Purpose:** Enforce field rules regardless of how the data is entered — via form, import, or APIs.
* **Runs on:** Server
* **Use Cases:**
  + Ensure the "State" field is always filled in, whether via the UI or an integration.

🛠️ *Data Policies enforce rules at the database level.*

**💻 Client Scripts (Key to Client-Side Logic)**

Client Scripts allow custom JavaScript to be executed **in the browser**, tied to form behaviour.

**🔸 Types of Client Scripts**

**✅ 1. onLoad**

**When:** Runs when the form is loaded.  
**Use Case:** Auto-fill a field when the form opens.

A close-up of a computer code

AI-generated content may be incorrect.

**✅ 2. onChange**

**When:** Runs when a specified field value changes.  
**Use Case:** Show or hide a field based on a selection.

A computer screen shot of a computer code

AI-generated content may be incorrect.

**✅ 3. onSubmit**

**When:** Executes just before form submission.  
**Use Case:** Prevent submission if a mandatory field isn’t filled.

A screen shot of a computer code

AI-generated content may be incorrect.

**✅ 4. onCellEdit**

**When:** Runs when a cell in a list is edited (List v3).  
**Use Case:** Alert the user or trigger a background process when a value is edited inline.

A close-up of a computer screen

AI-generated content may be incorrect.

**🧩 Client-Side APIs in ServiceNow**

**🔸 g\_form – GlideForm API**

Controls and interacts with form elements.

**Common Methods:**

* g\_form.getValue('field\_name') → Get field value
* g\_form.setValue('field\_name', 'value') → Set field value
* g\_form.setVisible('field\_name', false) → Show/hide fields
* g\_form.setMandatory('field\_name', true) → Make field mandatory
* g\_form.showFieldMsg('field\_name', 'message', 'info') → Show inline messages

**🔸 g\_user – GlideUser API**

Gives access to current logged-in user's info.

**Common Properties:**

* g\_user.name → Full name of the user
* g\_user.userID → Sys\_id of the user
* g\_user.hasRole('admin') → Check if user has a role

**📝 Summary for Beginners**

| **Feature** | **Type** | **Runs On** | **Script Required** | **Example Use** |
| --- | --- | --- | --- | --- |
| UI Policy | Client | Form | ❌ | Show/Hide fields |
| Data Policy | Server | Form/API | ❌ | Enforce required fields |
| Client Script | Client | Browser | ✅ | Auto-fill or validate data |
| g\_form | API | Client Side | ✅ | Modify form fields |
| g\_user | API | Client Side | ✅ | Get user info |

📌 **Pro Tips for Beginners:**

* Test all scripts in a **sub-production** environment.
* Use console.log() or gs.info() for debugging.
* Keep scripts **modular** and **well-commented**.
* Use UI Policies for simple logic — only use scripts when necessary.

**✅ Use Cases for Client-Side Scripting in ServiceNow**

**🔷 1. UI Policy Use Cases**

**📌 Use Case: Hide the "Resolution Notes" field unless the Incident State is "Resolved"**

* **Why?** To keep the form clean and only show relevant fields based on the current state.
* **How?** Create a UI Policy on the "state" field → Condition: State == Resolved → Action: Show "Resolution Notes".

**🔷 2. Data Policy Use Cases**

**📌 Use Case: Make "Assignment Group" mandatory for Incident, whether data comes via form or API**

* **Why?** Enforce data integrity even for integrations or imports.
* **How?** Create a Data Policy on the "assignment\_group" field and set it as mandatory.

**🔷 3. Client Script: onLoad**

**📌 Use Case: Auto-fill the caller field with the logged-in user**

* **Why?** Save user time and reduce data entry mistakes.
* **Script:**

A close-up of a computer code

AI-generated content may be incorrect.

**🔷 4. Client Script: onChange**

**📌 Use Case: Show a warning message if the category is set to "Hardware"**

* **Why?** Alert the user to follow a specific process.
* **Script:**

A white screen with black text

AI-generated content may be incorrect.

**🔷 5. Client Script: onSubmit**

**📌 Use Case: Prevent form submission if "Short Description" is empty**

* **Why?** Enforce critical fields before saving the record.
* **Script:**

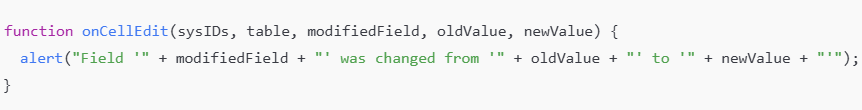
A screen shot of a computer code

AI-generated content may be incorrect.

**🔷 6. Client Script: onCellEdit**

**📌 Use Case: Log changes made directly in a list view (e.g., Priority changed)**

* **Why?** Track user interactions and notify or log inline edits.
* **Script:**



**🔷 7. g\_form API Use Case**

**📌 Use Case: Dynamically make "Impact" field mandatory when Priority is "1"**

* **Why?** Enforce critical information collection based on high priority.
* **Script (onChange of Priority):**

A computer code with text

AI-generated content may be incorrect.

**🔷 8. g\_user API Use Case**

**📌 Use Case: Hide an "Admin Only" section from non-admin users**

* **Why?** Secure sensitive form elements from regular users.
* **Script (onLoad):**

A computer code with text

AI-generated content may be incorrect.