ServiceNow Training Course - Detailed Curriculum

MODULE 9: Business Rules

Overview:
Business Rules in ServiceNow are server-side scripts that trigger when records are inserted, updated, deleted, or queried. They are often used to automate processes and enforce business logic.

Types of Business Rules:

1. **Before Business Rule**:
    - Runs before a record is saved to the database.
    - **Use Case**: Validating a form before submission or modifying data before it's committed.
        - Example: Checking if the user's email matches a valid pattern before saving the record.
        - **Syntax**: current.email = emailPattern();
2. **After Business Rule**:
    - Runs after the record is saved.
    - **Use Case**: Used for tasks like updating related records or sending notifications.
        - Example: Send an email notification to a user once their incident is closed.
        - **Syntax**: gs.eventQueue('incident.closed', current, current.sys_id);
3. **Async Business Rule**:
    - Runs asynchronously, allowing non-blocking operations.
    - **Use Case**: Used for time-consuming tasks like logging background activities without affecting the user's experience.
        - Example: Logging the activity of a closed incident to a separate table in the background.
        - **Syntax**: gs.info('Incident closed: ' + current.sys_id);
4. **Display Business Rule**:
    - Runs when a record is displayed on the form.
    - **Use Case**: Modify data before it's shown to the user, such as making fields visible or hidden based on user roles.
        - Example: Display or hide fields on the incident form depending on the user's role.
        - **Syntax**: current.setDisplayValue('priority', 'High');

Use Case:

- Use a **Before Business Rule** to ensure that a user's email follows a specific pattern.
- Use an **After Business Rule** to send an email notification when an incident is closed.

Practice Tasks:

1. Create a **Business Rule** to enforce that a priority field in an Incident record cannot be blank when it is saved.
2. Use a **Display Business Rule** to show/hide fields based on the incident type.

MODULE 10: Server-Side APIs (Glide-System API)

Overview:

ServiceNow provides built-in server-side APIs like **GlideRecord**, **GlideDateTime**, and more for efficient database interactions and time manipulations.

Key APIs:

1. **GlideRecord**:
   - Used for querying and manipulating records in ServiceNow tables.
   - **Use Case**: Fetching records from a table, updating data, or creating new records.
     - Example: Query all incidents that are “Open”.
     - **Syntax**:

```
var gr = new GlideRecord('incident');
gr.addQuery('state', 'Open');
gr.query();
while (gr.next()) {
    gs.info(gr.number);
}
```

2. **GlideDateTime**:
   - Handles date and time manipulations.
   - **Use Case**: Calculate the time difference between incident creation and resolution.
     - Example: Adding days to a date or calculating the difference in time.
     - **Syntax**:

```
var dt = new GlideDateTime();
dt.addDays(3);
gs.info(dt);
```

3. **Encoded Query**:
   - A string representation of a GlideRecord query.
   - **Use Case**: Filter records efficiently using encoded query format.
     - Example: Query incidents with a specific encoded condition.

Syntax:

```
var gr = new GlideRecord('incident');
gr.addEncodedQuery('state=1^active=true');
gr.query();
```

Use Case:

- Use **GlideRecord** to fetch all unresolved incidents and send a notification to the assigned user.
- Use **GlideDateTime** to calculate the time spent on incidents.

Practice Tasks:

1. Write a **GlideRecord** script to fetch all unresolved incidents.
2. Implement **GlideDateTime** to calculate the time difference between when an incident was created and resolved.

MODULE 11: Glide Ajax and Notifications

Overview:

Glide Ajax is used for calling server-side code from client scripts. **Notifications** in ServiceNow are automated alerts triggered by system events.

Key Concepts:

1. **Glide Ajax**:
    - Calls server-side scripts from client-side scripts to retrieve or manipulate data.
        - Example: Sending a request to the server and receiving the result.
        - **Syntax**:

```
var ga = new GlideAjax('GetIncidentStatus');
ga.addParam('sys_id', current.sys_id);
ga.getXMLAnswer(function(response) {
    var status = response.responseXML.documentElement.getAttribute('answer');
    alert(status);
});
```

2. **Notifications**:
    - ServiceNow allows you to create email, SMS, and system notifications triggered by events.
        - Example: Sending an email when an incident state changes to “Closed”.
        - **Use Case**: Trigger notifications on events like state changes, approvals, etc.

Use Case:

- Use **Glide Ajax** to retrieve the status of an incident and display it dynamically on the form.
- Set up an email notification to alert a user when their high-priority incident is assigned to them.

Practice Tasks:

1. Create a **Glide Ajax** script to fetch incident details for display on the user form.
2. Set up an **email notification** to inform users when their incident’s state is changed.

MODULE 12: Access Control Lists (ACLs)

Overview:
ACLs control access to ServiceNow records and fields. They define who can read, write, or create records in a table.

Key ACL Types:

1. **Read ACL**:
    - Grants access to view the field or record.
    - **Use Case**: Restrict access to view records based on user roles.
2. **Write ACL**:
    - Grants access to modify records or fields.
    - **Use Case**: Restrict editing fields like priority based on specific roles.
3. **Create ACL**:
    - Allows users to create new records for a specific table.
    - **Use Case**: Prevent unauthorized users from creating records in sensitive tables.

Use Case:

- **Read ACL**: Ensure only the creator of an incident can modify it.
- **Write ACL**: Restrict updating the priority field to users with the “Admin” role.

Practice Tasks:

1. Create an ACL to restrict access to a custom table.
2. Set up a scheduled job to run nightly and close incidents older than 90 days.

MODULE 13: REST and SOAP APIs

Overview:
ServiceNow supports both **REST** and **SOAP** APIs for integrating with external systems.

REST API:

- **Use Case**: Allow external systems to query incident records.
- **Example**: Creating a POST request to create an incident.

SOAP API:

- **Use Case**: Update user information in ServiceNow from external systems.
- **Example**: SOAP web service to update employee data.

Practice Tasks:

1. Create a **POST request** to create an incident via the REST API.
2. Set up **SOAP** web services for integrating with an external HR application.

MODULE 14: Table API Using HTTP Methods

Overview:
The **Table API** allows interaction with ServiceNow tables using **HTTP methods** (POST, GET, PUT, DELETE).

1. **POST**:
    - Create new records.
        - **Example**: Creating a new incident record.
        - **Request**: POST /api/now/table/incident
2. **GET**:
    - Retrieve records.
        - **Example**: Fetch all active incidents.
        - **Request**: GET /api/now/table/incident
3. **PUT**:
    - Update existing records.
        - **Example**: Update the state of an incident.
        - **Request**: PUT /api/now/table/incident/{sys_id}
4. **DELETE**:
    - Delete records.
        - **Example**: Delete a specific incident.
        - **Request**: DELETE /api/now/table/incident/{sys_id}

Use Case:

- Use **POST** to create a new incident record from an external system.

- Use **GET** to retrieve all active incidents and display them on a dashboard.

MODULE 15: Scripted REST API

Overview:
Scripted REST APIs allow you to create custom RESTful endpoints in ServiceNow.

Use Case:

- Build a **GET endpoint** to fetch high-priority incidents from an external system.
- **Example**: A scripted REST API that returns specific incident records.

Practice Tasks:

1. Create a **Scripted REST API** with a GET method for retrieving incidents.
2. Test the endpoint using **Postman**.

MODULE 16: Inbound and Outbound Email

Overview:
ServiceNow can handle inbound emails (creating records) and outbound emails (sending notifications).

Inbound Email:

- **Use Case**: Parse an inbound email and create an incident record.

Outbound Email:

- **Use Case**: Send email notifications when incidents are closed.

MODULE 17: Conclusion

🔁 **Summary: Recap of All Key Concepts Covered**

This module serves as a comprehensive review of all the concepts you've learned in the course. Below is a summarized recap of each module:

🔷 **Module 1–3: Introduction & Basics**

- **Understanding what ServiceNow is and how it is used in ITSM and business automation.**
- **Navigation, forms, lists, filters, and ServiceNow UI basics.**

🔷 **Module 4: Tables and Fields**

- **Understanding table hierarchy (Base Tables, Extended Tables).**
- **Creating custom tables and fields, and field types (string, choice, reference).**

🔷 **Module 5: Forms and Lists**

- **Customizing form layouts and list views.**
- **Adding related lists, sections, and personalizing UI for better usability.**

🔷 **Module 6: Client-Side Scripting**

- **Working with Client Scripts (onLoad, onChange, onSubmit).**
- **DOM manipulation using GlideForm (g_form) and GlideUser (g_user) APIs.**

🔷 **Module 7: UI Policies and UI Actions**

- **Creating dynamic behavior without coding using UI Policies.**
- **Adding buttons and context menu options using UI Actions.**

🔷 **Module 8: Data Policies and Data Dictionary**

- **Enforcing data integrity and setting mandatory fields using Data Policies.**
- **Understanding Dictionary Entries and Dictionary Overrides.**

🔷 **Module 9: Business Rules**

- **Server-side scripting for automating logic at database-level.**
- **Using Before, After, Display, and Async Business Rules for various use cases.**

🔷 **Module 10: Server-Side APIs (Glide APIs)**

- **Using GlideRecord for querying/updating records.**
- **GlideDateTime for date operations, addEncodedQuery for complex filters.**

🔷 **Module 11: Glide Ajax & Notifications**

- **Calling server-side logic from client-side using Glide Ajax.**
- **Triggering Notifications and Events for automated alerts.**

🔷 **Module 12: Access Control Lists (ACLs)**

- **Controlling access to records and fields using Create, Read, Write ACLs.**
- **Use of conditions, scripts, and roles in ACLs.**

🔷 **Module 13–14: REST & SOAP APIs and Table API**

- **Consuming and exposing APIs using HTTP methods (GET, POST, PUT, DELETE).**
- **Using Postman for testing integration with ServiceNow's Table APIs.**

🔷 **Module 15: Scripted REST APIs**

- **Creating custom REST endpoints.**

- **Handling external requests using Scripted REST APIs with custom logic.**

🔷 **Module 16: Inbound/Outbound Email and MID Server**

- **Configuring Inbound Email Actions to auto-create records.**
- **Using Outbound Notifications and the role of MID Servers in integrations and discovery.**

🧠 **Mock Interviews: Practice and Preparation**

Mock interviews simulate real-world interview scenarios to help solidify your knowledge and improve confidence. Here's how you should prepare:

🔍 **Interview Format Simulation**

1. **Technical Round (Scripting & Platform Knowledge)**
   Example questions:
   - **Explain the difference between Client Script and Business Rule.**
   - **How do you debug a GlideRecord script?**
   - **Write a script to fetch all incidents assigned to the logged-in user.**
2. **Scenario-Based Questions**
   Example:
   - **A user reports that an incident isn't visible to them after creation. How would you troubleshoot?**
   - **Describe how you would create a custom approval workflow using Business Rules and Notifications.**
3. **REST API / Integration Round**
   - **How do you secure a REST API in ServiceNow?**
   - **Write a POST API call that creates an incident with a short description and priority.**
4. **Access Control & Security Questions**
   - **How would you restrict access to a specific record in a table?**
   - **What is the difference between a Role and an ACL?**

📝 **Sample Mock Interview Questions by Module**

Module	Sample Question
Business Rules	What is the difference between Async and After Business Rules?
GlideRecord	How do you fetch active incidents created in the last 7 days?
Notifications	How do you send an email only when the priority is critical?
REST API	What HTTP method would you use to update a record, and why?
ACLs	How do you prevent a user from editing the 'priority' field unless they are in the ITIL role?

💡 **Tips for Interview Success**

- **Understand real use cases, not just definitions.**
- **Practice scripting — especially GlideRecord, GlideForm, and GlideAjax.**
- **Use the ServiceNow Developer Instance for hands-on testing.**
- **Review your custom apps or projects and be ready to discuss them.**

Resources

✅ **MODULE 9: Business Rules**

- **Business Rules Overview:**
  https://www.servicenow.com/docs/bundle/yokohama-api-reference/page/script/business-rules/concept/c_BusinessRules.html

---

✅ **MODULE 10: Server-Side APIs (Glide-System API)**

- **GlideRecord API:**
  https://www.servicenow.com/docs/bundle/yokohama-api-reference/page/app-store/dev_portal/API_reference/GlideRecord/concept/c_GlideRecordAPI.html
- **GlideDateTime API:**
  https://www.servicenow.com/docs/bundle/yokohama-api-reference/page/app-store/dev_portal/API_reference/GlideDateTime/concept/c_GlideDateTimeAPI.html

---

✅ **MODULE 11: Glide Ajax and Notifications**

- **GlideAjax API:**
  https://www.servicenow.com/docs/bundle/vancouver-api-reference/page/app-store/dev_portal/API_reference/GlideAjax/concept/c_GlideAjaxAPI.html
- **Notifications Overview:**
  https://www.servicenow.com/docs/bundle/yokohama-platform-administration/page/administer/notification/reference/notifications.html

---

✅ **MODULE 12: Access Control Lists (ACLs)**

- **Access Control Rules:**
  https://www.servicenow.com/docs/bundle/vancouver-platform-security/page/administer/contextual-security/concept/access-control-rules.html
- **Understanding ACLs:**
  https://www.servicenow.com/community/itsm-articles/access-control-lists-in-servicenow/ta-p/2304210

✅ **MODULE 13: REST and SOAP APIs**

- **REST API Reference:**
  https://www.servicenow.com/docs/bundle/yokohama-api-reference/page/integrate/inbound-rest/concept/c_RESTAPI.html
- **SOAP Web Service Overview:**
  https://www.servicenow.com/docs/bundle/xanadu-api-reference/page/integrate/inbound-soap/concept/c_SOAPWebService.html

---

✅ **MODULE 14: Table API Using HTTP Methods**

- **Table API Documentation:**
  https://www.servicenow.com/docs/bundle/yokohama-api-reference/page/integrate/inbound-rest/concept/c_TableAPI.html

---

✅ **MODULE 15: Scripted REST API**

- **Scripted REST APIs Overview:**
  https://www.servicenow.com/docs/bundle/yokohama-api-reference/page/integrate/custom-web-services/concept/c_CustomWebServices.html

---

✅ **MODULE 16: Inbound and Outbound Email**

- **Inbound Email Actions:**
  https://www.servicenow.com/docs/bundle/xanadu-platform-administration/page/administer/notification/concept/c_InboundEmailActions.html
- **Outbound Mail Configuration:**
  https://www.servicenow.com/docs/bundle/xanadu-platform-administration/page/administer/reference-pages/reference/r_OutboundMailConfiguration.html

---

✅ **MODULE 17: Conclusion & Practice**

- **ServiceNow Developer Portal:**
  https://developer.servicenow.com
- **Create a Personal Developer Instance:**
  https://developer.servicenow.com/dev.do#!/guides/vancouver/now-platform/pdi-guide/create-a-personal-developer-instance