BPI Developer Advanced Training

BPI133ILT, Revision 1.0

Blue Planet Inventory

Challenge Lab

Security

Ciena cannot be responsible for unauthorized use of equipment and will not make allowance or credit for unauthorized use or access.

Contacting Ciena

Corporate headquarters	410-694-5700 or 800-921-1144	www.ciena.com
Customer technical support/warranty		
In North America	1-800-CIENA24 (243-6224) 410-865-4961	Email: CIENA24@ciena.com
In Europe, Middle East, and Africa	800-CIENA-24-7 (800-2436-2247) +44-207-012-5508	Email: CIENA24@ciena.com
In Asia-Pacific	800-CIENA-24-7 (800-2436-2247) +81-3-6367-3989 +91-124-4340-600	Email: CIENA24@ciena.com
In Caribbean and Latin America	800-CIENA-24-7 (800-2436-2247) 410-865-4944 (USA)	Email: CIENA24@ciena.com
Sales and General Information	410-694-5700	E-mail: sales@ciena.com
In North America	410-694-5700 or 800-207-3714	E-mail: sales@ciena.com
In Europe	+44-207-012-5500 (UK)	E-mail: sales@ciena.com
In Asia	+81-3-3248-4680 (Japan)	E-mail: sales@ciena.com
In India	+91-124-434-0500	E-mail: sales@ciena.com
In Latin America	011-5255-1719-0220 (Mexico City)	E-mail: sales@ciena.com
Training		E-mail: learning@ciena.com

For additional office locations and phone numbers, please visit the Ciena website at www.ciena.com

Change History

Blue Planet Release	Revision	Publication Date	Reason for Change
22.08.2	1.0	March 2023	Initial release.

Contents

Introduction

This document describes the concepts of the Challenge Labs for the Blue Planet Inventory Academy and contains the Challenge Lab tasks for Week 3 of the Academy.

After the ILT training is completed, you are provided with an additional day of access to the remote lab environment, where you should independently perform the Challenge Lab exercises. This will give you an opportunity for additional self-study and assessment of the knowledge gained during the training.

The Challenge Lab document and the exercises are similar to the Lab Guide, but only the final goal/outcome is presented to you, without step-by-step instructions.

For any questions or assistance, the instructor is available in the training chatroom for the duration of the Challenge Lab. You will be assisted as needed with troubleshooting issues that may arise during the Challenge Lab.

After the Challenge Lab is completed, a 1-hour review workshop will be delivered to the student group.

Task 1: BPI Customization Points

In this task, you modify the behavior of the DeviceNaming class that you created in Lab 1 Task 2.

Objectives

- From your Student PC desktop, open Visual Studio Code (VSC), and open your existing DeviceNaming.groovy file.
  - Modify the code so that if the populated device name is empty, the derivedName variable is prefixed with „_name_not_specified_ „ before returning.
    - For example, after the code modification, create another ASR 9001 in Lab-Loc1 building, leave Device Name empty and after clicking Apply, the name of the device should equal „_name_not_specified_ASR 9001_LAB-LOC1".
  - Onboard the changes to the server.
  - Verify that the changes are successful. Look out for syntax and other errors in the code which may prevent the bean to be set up properly.
  - Optionally
    - Use a logger to output the values of your variables to the log file.
    - Think of another use case scenario for the DeviceNaming class modification and try to implement it. Use Javadocs to get information on classes and methods for your use case.

Revision 1.0

March 2023

Page 6

Task 2: Advanced Graph DB Topics

In this task, you extend the graph model so that you can terminate additional logical connection types on Ciena OME 6500 devices.

Objectives

- Construct a Cypher query that will return all the **LogicalConnection** nodes that are *not* compatible with ports on line cards, which can be inserted into Ciena **OME 6500** slots. (Cannot be indirectly terminated on Ciena OME 6500 slots).

- Chose any LogicalConnection, which is currently not compatible with the Ciena OME 6500 cards and use the Metadata Modeller to add that compatibility.

- Use the BPI UI to create a connection of that new type between **DEVICE-A** and **DEVICE-B**.

- Construct a Cypher query that will return the end-to-end route of that newly created connection.

Task 3: UI Customizations

In this task, you add a new property field to the device summary page and create the logic to manage the value for that property.

Objectives

- Create a new component which will extend the Equipment Page functionality.
  - Add a new property named BPI ID which is read-only.
- Add the new component to the building context menu and to the Device Summary context menu option.
  - Note: Creating a new device in this task should be done only through the building context menu.
- Implement the Angular service to populate the "BPI ID" property with the drniID value from the database.
- Implement the REST API logic to populate the "BPI ID" property with the drniID value from the database.

Task 4: Advanced Guided Operations

In this task, you create a new workflow definition with a few modifications, and then use APIs to manage instances of that definition.

NOTE: The Groovy script, needed for completing this task, is provided in the lab.

Objectives

- Open Modeler from the BPI UI and set it up to use the definition from the lab exercise. Do not create the definition from scratch but find a way to get it to Modeler. Modify the definition so that:

  o **User task:** When running the task, the name of the menu option should be **Choose City and Building New Header**. Deploy the new definition and create a new instance to verify this.

  o **Service task**: In the groovy script source, add a line in the code to output a value of an object or variable of your choice to the log file. Onboard the new script and verify by tailing the catalina.out log file in the web-0 pod. Create a new workflow instance and confirm that the tasks can be executed correctly while verifying in the log file.

  o **API:**

    ▪ Use BPMN API to create a new instance of your new workflow.

    ▪ Test API calls of your choice on this workflow.

    ▪ List tasks in your new instance.

    ▪ Modify the parameters of the created instance.

    ▪ Suspend and reactivate your new instance.

Task 5: Rapid Path Search

In this task, you modify the behavior of the class that you created in the lab exercise. Specifically, you enable logging in the class and observe messages in the Neo4j log.

Objectives

- From the BP UI, delete DAL-HOU-ETH-Connection, so that it does not interfere with new searches.

- Enable logging in the ProjectBusinessRuleTechnology class that will enable you to inspect for each segment, the index of the segment, what is the start node, end node, along with all the properties of those nodes.
    - Use the **this.getLog()** method to write messages to the log.
    - The information you are looking for is available as part of the analysisInfo object.
    - When done, build and deploy the project.
    - **NOTE:** the Neo4j log file can be accessed from the BPI server shell by executing the **tail_neo4j_log.sh** script.

- In the getWeightCalculator method of the ProjectBusinessRuleTechnology class, enable logging of the returned weight values for a segment.
    - Observe what values are returned for different segments.

Task 6: Data Ingestion Framework

In this task, you create DROOLS rule for ingesting the card into the device in planned perspective, onboard the rule, and create Ingestion Data Source with that rule.

NOTE: You can view the existing Postman requests for reference.

Objectives

- Create and onboard rule to ingest the A9K-24X10GE-TR card information into the device ASR 9001. Name the rule as **Lab Ingestion Card Rule**.
    - o Hint: Inspect the TMF639 Ingestion Card Rule that is already in the database. Copy the code and modify it to match the requirements of this task.
- Create data source:
    - o Name: **Data_Source_001**
    - o Ingestion config:
        - ▪ Kafka topic: **data-ingestion-challenge-001**
        - ▪ Group id: **challenge_group**
    - o Retry config
        - ▪ Kafka topic: **data-ingestion-challenge-001-retry**
        - ▪ Group id: **challenge_group_retry**
        - ▪ Timeout(seconds): **10**
    - o Retry count: **4**
    - o Perspective: **2**
    - o Rules: ["Lab Ingestion Card Rule"]
    - o Operations: **manageInventory**
    - o ignoreCompatibilities: **true**
    - o createActivityCallout: **com.blueplanet.inventory.ingestionFramework.callouts.CreateActivityCallout.createActivity**
- Create an ASR 9001 device named LabDevice06 through BPI-UI.
- Generate an appropriate event message and test if the card information can be properly ingested into the LabDevice06 device.

Task 7: The Staging Ingestion Framework

In this task, you configure the Staging Ingestion Framework to support ingestion of the new equipment into the perspective 101.
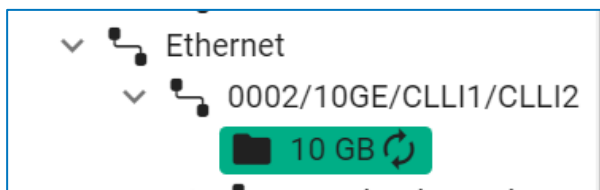
Objectives

- Configure new the Staging Ingestion Source where NETWORK_PERSPECTIVE is set to 101.
    - Modify the configuration files to apply the mapping configuration to this new source.
        - Hint: add the ARCHETYPEINSTANCE_ID of the source to the ARCHETYPEINSTANCE_ID_LIST column of the CSV config files using a pipe („|") symbol.
        - Files to modify:
            - ResourceObject.csv
            - ResourceObjectProperty.csv
            - Callout.csv
            - EquipmentConverter.csv
            - EquipmentInstance.csv
            - EquipmentMatching.csv
            - EquipmentProperty.csv
            - EquipmentRelation.csv
- Configure the mapping for:
    - Device ASR 9006
    - Card A9K-24X10GE-TR
- Configure positional matching for the card to be placed in one of the slots 0-3 of the device.
    - Hint: create the device in BPI-UI and add the card to it. Inspect the changes in the database to get an idea of the final mapping.
- Test the changes.
    - Hint: push the changes to the git repository and wait for the pipeline to finish the execution before checking the database.

Task 8a: Inventory Tree

In this task, you onboard a new XML configuration file, which extends the functionality of the Inventory Tree. You add an additional property to logical connections, which display the bandwidth of the connection in the tree.

Objectives

- In VSC, navigate to the cloned bpi area and find the XML file which configures the Inventory Tree. Create a new .xml file in the same directory. Name the file properly, so that after you onboard it, BPI will merge the contents of the new file with the existing configuration.

- Add an object to your XML file, which will override the configuration of the **<tree><treeLevel treeLevelName="LogicalConnectionsBandwidth">** element. Update the currently used CQL query so that it will return the logical connection bandwidth as the displayed name of the tree.

- Hint: recall that the bandwidth of the logical connection is defined in the Graph Database as the name of a certain Model node.

- Use the BPI SDK to onboard the new XML configuration file.

- The bandwidth information should now be displayed in the Inventory Tree, similar to the following figure:

Task 8b: Properties Inspector

In this task, you extend the functionality of the Properties Inspector. You display a new property for buildings and make it possible for users to edit the value directly from the Properties Inspector.

Objectives

- From the BPI cloned area, navigate to and open the XML file, which configures the Properties Inspector.

- Update the XML file, so that the Properties Inspector displays a new property, labeled **Building Type**. The new property should only be displayed when the Properties Inspector is displaying properties of a Building. The name of the property in the Graph Database is **buildingType**.

- Make sure that users can update the new property directly from the Properties Inspector.

- Onboard the new configuration and verify the functionality.