Adobe CQ Help /
Programmatically Accessing Adobe CQ Content using the JCR API

Article summary

Summary	Discusses how to programmatically manipulate the Adobe CQ JCR using the JCR API. This article manipulates the JCR by using an external Java console application.
Digital Marketing Solution(s)	Adobe Experience Manager (Adobe CQ)
Audience	Developer (intermediate)
Required Skills	Java
Tested On	Adobe CQ 5.5, Adobe CQ 5.6

Introduction

You can programmatically modify nodes and properties located within the Adobe CQ repository, which is part of the Adobe Marketing Cloud. To access the CQ repository, you use the Java Content Repository (JCR) API. You can use the Java JCR API to perform create, replace, update, and delete (CRUD) operations on content located within the Adobe CQ repository. For more information about the Java JCR API, see http://jackrabbit.apache.org/jcr-api.html.

This development article creates a Java class that modifies nodes and properties within Adobe CQ. The Java class connects to a local instance of Adobe CQ and creates nodes and stores data values to node properties. You store data by manipulating node properties.

Note: This development article modifies the Adobe CQ JCR from an external Java application. In contrast, you can modify the JCR from within an OSGi bundle using the JCR API. For details, see Persisting CQ data in the Java Content Repository.

Note: To learn how to query the Adobe CQ JCR using the JCR Query API, see Querying Adobe Experience Manager Data using the JCR API.

Add the JCR JAR file

To the top

To use the JCR API, add the version of the jackrabbit-standalone JAR file that works with your instance of Adobe Experience Manager. For example, you are using AEM 5.5, you can use jackrabbit-standalone-2.4.3.jar. If you are using AEM 5.5, and you use a newer jackrabbit-standalone JAR file, such as jackrabbit-standalone-2.6.5.jar, then an exception is thrown. You can obtain the jackrabbit-standalone JAR file from http://jackrabbit.apache.org/jcr-api.html.

Create a Repository instance

To the top

Although there are different ways to connect to a repository and establish a connection, this development article uses a static method that belongs to the `org.apache.jackrabbit.commons.JcrUtils` class. The name of the method is `getRepository`. This method takes a string parameter that represents the URL of the Adobe CQ server. For example `http://localhost:4503/crx/server`.

The `getRepository` method returns a `Repository` instance, as shown in the following code example.

```
//Create a connection to the Adobe Day CQ repository running on local
host
Repository repository =
JcrUtils.getRepository("http://localhost:4503/crx/server");
```

Create a Session instance

To the top

The `Repository` instance represents the CRX repository. You use the `Repository` instance to establish a session with the repository. To create a session, invoke the `Repository` instance's `login` method and pass a `javax.jcr.SimpleCredentials` object. The `login` method returns a `javax.jcr.Session` instance.

You create a `SimpleCredentials` object by using its constructor and passing the following string values:

- The user name
- The corresponding password

When passing the second parameter, call the String object's `toCharArray` method. The following code shows how to call the login method that returns a `javax.jcr.Session` instance.

```
//Create a Session instance
javax.jcr.Session session = repository.login( new
SimpleCredentials("admin", "admin".toCharArray()));
```

Create a Node instance

To the top

Use a `Session` instance to create a `javax.jcr.Node` instance. A `Node` instance lets you perform node operations. For example, you can create a new node. To create a node that represents the root node, invoke the `Session` instance's `getRootNode` method, as shown in the following line of code.

```
//Create a Node
Node root = session.getRootNode();
```

Once you create a `Node` instance, you can perform tasks such as creating another node and adding a value to it. For example, the following code creates two nodes and adds a value to the second node.

```
1 // Store content
2 Node day = adobe.addNode("day");
3 day.setProperty("message", "Adobe CQ is part of the Adobe Digital Marketing Suite
```

Retrieve Node Values

To the top

To retrieve a node and its value, invoke the `Node` instance's `getNode` method and pass a string value that represents the fully-qualified path to the node. Consider the node structure created in the previous code example. To retrieve the day node, specify adobe/day, as shown in the following code:

```
1 // Retrieve content
2 Node node = root.getNode("adobe/day");
3 System.out.println(node.getPath());
4 System.out.println(node.getProperty("message").getString());
```

Create nodes in the Adobe CQ Repository

To the top

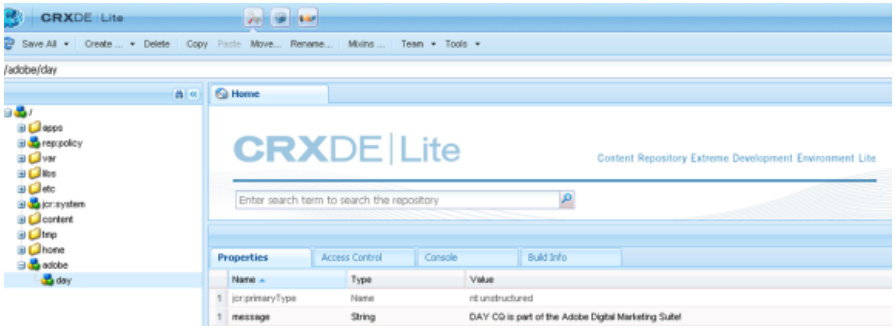
The following Java code example represents a Java class that connects to Adobe CQ, creates a `Session` instance, and adds new nodes. A node is assigned a data value and then the value of the node and its path is written out to the console.

```
1  /*
2   * This Java Quick Start uses the jackrabbit-standalone-2.4.0.jar
3   * file. See the previous section for the location of this JAR file
4   */
5
6  import javax.jcr.Repository;
7  import javax.jcr.Session;
8  import javax.jcr.SimpleCredentials;
9  import javax.jcr.Node;
10
11 import org.apache.jackrabbit.commons.JcrUtils;
12 import org.apache.jackrabbit.core.TransientRepository;
13
14
15 public class GetRepository {
16
17
18 public static void main(String[] args) throws Exception {
19
20 try {
```

```
21
22      //Create a connection to the CQ repository running on local host
23      Repository repository = JcrUtils.getRepository("http://localhost:4503/crx/se
24
25     //Create a Session
26     javax.jcr.Session session = repository.login( new SimpleCredentials("admin",
27
28    //Create a node that represents the root node
29    Node root = session.getRootNode();
30
31    // Store content
32    Node adobe = root.addNode("adobe");
33    Node day = adobe.addNode("day");
34    day.setProperty("message", "DAY CQ is part of the Adobe Digital Marketing Suit
35
36
37    // Retrieve content
38    Node node = root.getNode("adobe/day");
39    System.out.println(node.getPath());
40    System.out.println(node.getProperty("message").getString());
41
42    // Save the session changes and log out
43    session.save();
44    session.logout();
45    }
46   catch(Exception e){
47    e.printStackTrace();
48    }
49   }
50  }
```

View the new nodes in CRXDE Lite

After you run the full code example and create the nodes, you can view the new nodes in the CRXDE Lite, as shown in the following illustration.



CRXDE Lite displaying the new node