Discover how easy it is to leverage Cognos on Bluemix by using the Embeddable Reporting service. In this tutorial, get step-by-step instructions on how to create, develop, and deploy an app with the Embeddable Reporting service on Bluemix from scratch.

Have you heard of IBM Cognos®, the world-class data analysis and reporting platform? The answer is probably yes. Do you have the ability to leverage such a powerful platform on IBM Bluemix? The answer is absolutely yes—by using the Bluemix Embeddable Reporting service. In this tutorial, get step-by-step instructions on creating, developing, and deploying an app employing the Embeddable Reporting service on Bluemix as well as how to design reports using Cognos Report Studio with Bluemix.

What you'll need to build your application

- A Bluemix account
- Familiarity with the Java™ programming language
• A DevOps Services account
• Familiarity with IBM Cognos (Optional)
• Familiarity with the cf cloud foundry command-line tool (Optional)

Run the app
Get the code

With the Embeddable Reporting service, you can easily build IBM Cognos reports on Bluemix. To build the app, you need:

• The Embeddable Reporting service.
• A compatible runtime—all Bluemix runtime environments are supported. We use the Java Liberty runtime in this example.
• A compatible NoSQL Database service—required to hold the metadata for the Embeddable Reporting service. Cloudant NoSQL DB, MongoLabs, and MongoDB are supported. In this example, Cloudant NoSQL DB is employed.
• An instance of the SQL DB service—the repository that holds the data to report.

With a few configuration steps and source code lines, you can easily integrate reporting features into apps.

Step 1. Create an app with the Embeddable Reporting service on Bluemix

Create an app on the dashboard

1. Log in to Bluemix with your Bluemix account.
2. Open the Catalog menu.
3. From the Runtimes section, click Liberty for Java.
4. In the App field, specify the name of your app; ours is set to cognos008.
5. Click Create. Wait for your application to provision.

Add the Cloudant NoSQL DB service

Create the Cloudant NoSQL DB instance needed to support the reporting services.

1. Click the app created in the dashboard.
2. Click Add a Service.
3. Choose Cloudant NoSQL DB under Data Management.
4. Click Create.
5. Click Restage to restart the application, if prompted.

Note: You can use the MongoDB or MongoLab service instead of the Cloudant NoSQL DB service, if you want.

Add the SQL database service

Now you are going to add the database instance that holds the data on top of which the reports are built.

1. Click the app created in the dashboard.
2. Click **Add a Service**.
3. Choose **SQL Database** under Data Management.
4. Click **Create**. Click **Restage**, if prompted.

### Add the Embeddable Reporting service
1. Click the app created in the dashboard.
2. Click **Add a Service**.
3. Choose **Embeddable Reporting** under the Business Analytics section.
4. Click **Create**.
5. Click **Restage**, if prompted. An app similar to the following is created.

![Image of Embeddable Reporting app](image)

### Gather the VCAP_SERVICES environment variables
Now take note of the environment variables that you can use in your code to see the services you have created.

1. Navigate to your application overview page.
2. Click **Environment Variables** on the left.
   
   Here you will find relevant environment variables, such as the Cloudant DB URL and the SQL Database jdbcurl as well as user name and password credentials. Copy these variables into a text file for later use.

### Step 2. Build the app
Now look at the source code to see the details about what you have done.
Initialize the database

Under the src/com/ibm/sqldb/sample directory, find the SQLDBSample.java file, which is the main class to initialize the database using the Java API. Accessing the SQLDatabase service in Bluemix is similar to the classic Java Database programming scenario.

The following source code snippet shows how to fetch the relevant info from VCAP_SERVICES—such as database IP, port, user name, and password—that will be used in the future for creating JDBC database connections.

```java
// VCAP_SERVICES is a system environment variable
// Parse it to obtain the DB2 connection info
String VCAP_SERVICES = System.getenv("VCAP_SERVICES");
if (VCAP_SERVICES != null) {
    // parse the VCAP JSON structure
    BasicDBObject obj = (BasicDBObject) JSON.parse(VCAP_SERVICES);
    String thekey = null;
    Set<String> keys = obj.keySet();
    writer.println("Searching through VCAP keys");
    // Look for the VCAP key that holds the SQLDB information
    for (String eachkey : keys) {
        writer.println("Key is: " + eachkey);
        // Just in case the service name gets changed to lower case in the future, use toUpperCase
        if (eachkey.toUpperCase().startsWith("SQLDB")) {
            thekey = eachkey;
        }
    }
    if (thekey == null) {
        writer.println("Cannot find any SQLDB service in the VCAP; exiting");
        return false;
    } else {
        BasicDBList list = (BasicDBList) obj.get(thekey);
        obj = (BasicDBObject) list.get("0");
        writer.println("Service found: " + obj.get("name");
        // parse all the credentials from the vcap env variable
        obj = (BasicDBObject) obj.get("credentials");
        databaseHost = (String) obj.get("host");
        databaseName = (String) obj.get("db");
        port = (int) obj.get("port");
        user = (String) obj.get("username");
        password = (String) obj.get("password");
        url = (String) obj.get("jdbcurl");
    }
}
```

After the database is created, a table is created and some SQL queries are executed to initialize the database. In this case, you will create a simple table with two columns: the city name and the quantity. You will insert some sample data and use the Embeddable Reporting service to analyze this dataset and draw the reports later.

```java
// create a table
try {
    // Create the CREATE TABLE SQL statement and execute it
    String sqlStatement = "CREATE TABLE " + tableName + " (CITY VARCHAR(20), NUMBERS INTEGER)"
    writer.println("Executing: " + sqlStatement);
    stmt.executeUpdate(sqlStatement);
} catch (SQLException e) {
    writer.println("Error creating table: " + e);
}
```
Access the Embeddable Reporting service
Under the src/com/ibm/ba/ers/sample directory, find the SampleServlet.java file, which is the main class to access the Embeddable Report service using Java code.

Collect the relevant info from VCAP_SERVICES before connecting to the Embeddable Reporting service. You can use either Cloudant, MongoLab, or MongoDB as the repository database for the Embeddable Reporting service.

Map<String, String> env = System.getenv();
String vcap = env.get("VCAP_SERVICES");
if (vcap == null) {
    System.out.println("No VCAP_SERVICES found");
    return;
}

try {
    JSONObject services = new JSONObject(vcap);
    @SuppressWarnings("unchecked")
    Set<String> serviceList = services.keySet();
    for (String service : serviceList) {
        JSONObject credentials = (JSONObject) ((JSONObject) ((JSONArray) services.get(service)).get(0)).get("credentials");
        if (name.indexOf("ERSERVICE") != -1) {
            reportingUri = (String) credentials.get("url");
            reportingUserId = (String) credentials.get("userid");
            reportingPassword = (String) credentials.get("password");
        } else if (name.indexOf("CLOUDANT") != -1) {
            bundleUri = (String) credentials.get("url") + "/ers";
        } else if (((String) credentials.get("url")).endsWith("cloudant")) {
            // cloudant uri doesn't have a db name, so add one
            bundleUri = (String) credentials.get("url") + "/ers";
        }
    }
} catch (JSONException e) {
    System.out.println("Error getting VCAP_SERVICES");
    System.out.println(e.getMessage());
}
else if ((name.indexOf("MONGOLAB") != -1)
    && (bundleUri == null)) {
    bundleUri = (String) credentials.get("uri");
} else if ((name.indexOf("MONGO") != -1)
    && (bundleUri == null)) {
    bundleUri = (String) credentials.get("url");
}

if (reportingUri == null) {
    System.err.println("No reporting service found");
    return;
}

if (bundleUri == null) {
    System.err.println("No bundle storage service found");
    return;
}

synchronized (this) {
    m_ersProxy = new ERSProxy(reportingUri, reportingUserId,
        reportingPassword, bundleUri);
    try {
        m_ersProxy.connect();
    } catch (IOException e) {
    }
    catch (JSONException e) {
    }
}

You must use a reverse proxy to get and delete requests when executing reports using the Embeddable Reporting service.

@override
protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
    String pathInfo = request.getPathInfo();
    if (pathInfo.compareTo("/") == 0) { 
        loadStaticPage("index.html", request, response);
    } else if (pathInfo.compareTo("/overview") == 0) {
        loadStaticPage("sample.html", request, response);
    } else if (pathInfo.startsWith("/ba/cre")) {
        if (m_ersProxy != null) { 
            m_ersProxy.doGet(request, response);
        } else {
            response.setStatus(404);
        }
    } else { 
        loadStaticPage(pathInfo, request, response);
    }
}

@override
protected void doDelete(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
    String pathInfo = request.getPathInfo();
    if (pathInfo.startsWith("/ba/cre")) {
        if (m_ersProxy != null) { 
            m_ersProxy.doDelete(request, response);
        } else {
            response.setStatus(404);
        }
    } else { 
        loadStaticPage(pathInfo, request, response);
    }
}
Deploy and run the app

1. Log in to DevOps Services with your IBM ID.
2. Click the Source Code. Click Fork Project. Enter your project name when requested, and then click Create to fork the repository.
3. Click Edit Code. On the menu section of the newly created project, click Create New to create a new application deployment to Bluemix.
4. Update the Application Name and Host field to <app_name>, the Bluemix app you created. Click **Deploy** to deploy the app. Wait a minute or so and check the deployment status.

5. Navigate to the app. The link is usually <app_name>.mybluemix.net. In this case, it is [http://cognos008.mybluemix.net](http://cognos008.mybluemix.net).

6. Click **Initialize the target database**. You might find that the relevant table is created and some data is inserted.

7. Back at the main page, click **View the overall report**. You will see a blank page. Don't worry. It's because we have not yet designed any reports on the Embeddable Reporting service. We discuss designing reports in the next section.

**Note:** If you prefer to build and deploy this project locally, you can build it using Eclipse and export it as a WAR archive file (for example, erssample.war). You can find a compiled sample in the erssample/MyData/war directory under the source code in case you do not want to compile it yourself. You can deploy the app using `cf` tools.

```
cf push <app_name> -p erssample.war
```

### Step 3. Design reports

To enjoy the powerful reporting features of the Embeddable Reporting service, you will need to design the reports first.

#### Open the Embeddable Reporting console

1. Navigate to your application overview page.
2. From the Overview menu, click **Embeddable Reporting**.
3. Click **Launch** to open the Embeddable Reporting console.
4. Click **Connect**. When prompted for the URI of your repository service, enter the CloudantNoSQLDB URL from the VCAP_SERVICES variables and click **Update**.

**Note:** If you are using MongoLabs or MongoDB as the repository database, enter the MongoDB URI from the VCAP_SERVICES variables when prompted.

#### Create a new application

1. Click the asterisk symbol.
2. Fill in the application name when prompted. In this case, the name is set to Cognos Demo Application. You can choose another name. Click **Continue**.

3. Fill in the info from the VCAP_SERVICES variables. The JDBC URI field is set to `sqldb.jdbcurl`, the User Name field is set to `sqldb <username>`, and the Password field is set to `sqldb <password>`. Click **Finish** to create the application.

![Create new application](image)

The application is created.

![Embeddable Reporting](image)

**View the data source**

A data source is automatically created after the previous step.

1. Open the Data Source Tab by clicking the folder symbol.
2. A data source called **datasource** has been created. Click the datasource row. Detailed information about it is displayed on the right.

3. You can create or update the data source on this tab if needed.

### Launch IBM Cognos Report Studio

1. Open the Report tab by clicking the folder symbol: 🗂️
2. Click the asterisk symbol. The IBM Cognos Report Studio page is launched.

### Design a sample report

With Cognos Report Studio, you are able to design different kinds of reports easily. In this case, we are going to design a very simple report with a pie diagram.

1. In the New window, choose **SQL Chart** and click **OK** to continue.
2. In the Data Source window, choose the data source that was created previously. Click **OK** to continue.

3. Enter the SQL query to fetch the data from the database that was initialized earlier for reporting. In this example, the query is set to **SELECT CITY, NUMBERS FROM LOCATION**. Click **OK**.
4. In the Insert Chart window, click **Pie, Donut** in the left pane. Choose **Pie**, and click **OK**.

![Insert Chart window](image)

5. Click the **Data Items** tab bar. Drag and drop **CITY** to **Series**. Then drag and drop **NUMBERS** to **Default measure** and **Categories (pies)**.

![Data Items tab](image)

6. In the chart, click **NUMBERS** under **Series**. In the Properties pane, choose **Total** for the Aggregate Function property. **Important**: This is a necessary step. Failure to perform it results in an error message saying **No Data Available** while the report is running.
7. Save it by clicking the diskette symbol. Enter the report name when prompted. In this case, the name is set to Cognos Demo Report. You can choose another name. Click Save.

8. Navigate to the app. The link is usually http://<app_name>.mybluemix.net. In this case, it is http://cognos008.mybluemix.net.

9. Click View the overall report to view the report generated.

10. Click Cognos Report Demo on the left pane to see the Cognos report.
    A pie diagram is used to analyze the city dataset created previously.
Design more diagrams: Append a column diagram

You are free to design more diagrams using Cognos Report Studio. Use the following steps to append a Column diagram to the report.

1. Right-click the pie diagram. Click **Copy**.
2. Right-click in the blank part of the page. Click **Paste** to get a copied pie diagram.
3. Right-click the copied pie diagram. Click **Convert Chart**.

Right-click the pie diagram. Click **Copy**.

2. Right-click in the blank part of the page. Click **Paste** to get a copied pie diagram.
3. Right-click the copied pie diagram. Click **Convert Chart**.

**Chart.**
4. In the Convert Chart window, click **Column** in the left pane. Choose **Column**, and click **OK**.

![Column chart conversion](image)

5. A Column diagram is appended to the report. Save it by clicking the diskette symbol.

6. Navigate to the app again. Click **View the overall report**. Click **Cognos Report Demo** on the left pane. A report with a pie diagram and a column diagram is displayed.

![Report with pie and column charts](image)

The Cognos Report Studio described here is based on IBM Cognos Business Intelligence Report Studio Version 10.2.2. For more information about how to create reports, see the user guide in the IBM Knowledge Center.

**Conclusion**

In this tutorial, we have provided step-by-step instructions on how to create, configure, and deploy an app leveraging the Embeddable Reporting service on Bluemix. With only a few configuration steps and a few lines of source code, you can see how easy it is to generate reports from the Embeddable Reporting service. Studying the code and embedding its features into other apps should be easy. That's the magic of Bluemix. Enjoy it.
The Embeddable Reporting service http://www.ibm.com/developerworks/topics/embeddable reporting service enables you to run IBM Cognos Business Intelligence reports within your Bluemix environment. The Cloudant NoSQL service http://www.ibm.com/developerworks/topics/cloudant nosql db service provides access to a fully managed NoSQL JSON data layer that's always on. The SQL Database service http://www.ibm.com/developerworks/topics/sql database service adds an on-demand relational database to your application.

© Copyright IBM Corporation 2015
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)