JDBC Datalist Action

Prevent SQL injection

When using Hash Variable that uses URL parameter or user-inputted value in the SQL query, ensure that these hash variable(s) are **escaped** in the query!

Make use of hash variable escape keywords, see Hash Variable - Escaping the Resultant Hash Variable.

Example of VULNERABLE query:

> *SELECT * FROM app_fd_sample_table WHERE c_value = '#requestParam.id#'*

To fix this, use **?sql** hash variable escape:

> *SELECT * FROM app_fd_sample_table WHERE c_value = '#requestParam.id**?sql**#'*

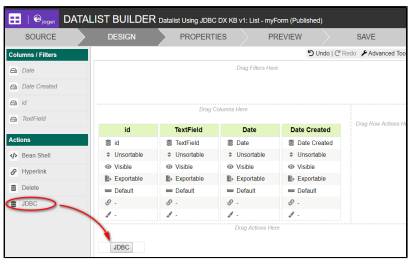
Introduction

JDBC Datalist Action allows you to perform SQL queries on one (a row action) or more records (a bulk action) in your datalist. You can specify which database to perform the SQL function, either the current Joget database (default datasource) or a custom datasource (external database).

JDBC Datalist Action can be used to delete records or perform an update on one or more records based on user selection in the datalist checkboxes.

JDBC Datalist Action Properties

Configure JDBC Datalist Action

Name	Description	Screens (Click to view)
Label	Datalist button label.	 Figure 1 : JDBC Action Menu
Confirmation Message	Confirmation message before performing action, for example "Are you sure?".	

Datasource	Target database to execute SQL statements on.
	
	Choices:-
	
	• Custom Datasource
	  ◦ JDBC Connection Parameters are needed for this choice.
	• Default Datasource
	  ◦ Points to the current database your copy of Joget currently connects to.	

Custom JDBC Driver	JDBC driver name. Example values: • com.mysql.jdbc.Driver (MySQL) • oracle.jdbc.driver.OracleDriver (Oracle) • com.microsoft.sqlserver.jdbc.SQLServerDriver (Microsoft SQL Server) Only applicable to "**Custom Datasource**" option.	

Custom JDBC URL	Database connection URL. Example: jdbc:mysql://localhost:3306/jwdb Only applicable to "**Custom Datasource**" option.
Custom JDBC Username	Database username. Example: root Only applicable to "**Custom Datasource**" option.
Custom JDBC Password	Specified database user's password. Only applicable to "**Custom Datasource**" option. **Test the connection parameters** Click on the "Test Connection" button at the bottom of the page to quickly test out your configurations.

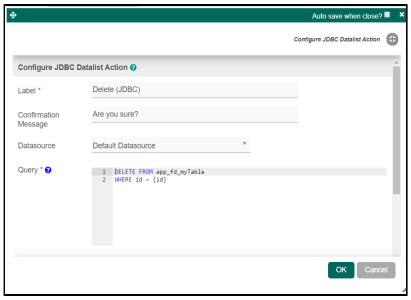


Figure 2 : JDBC Datalist Action Properties

Query

If a column name contains reserved keywords, do ensure it is encapsulated properly.

For example for MySQL, if the column identifier itself contains a dot symbol (.), it should be encapsulated like this:

```
SELECT `myAppName.myColumn` FROM app_fd_myTable;
```

Insert your SQL statement here. Use syntax like {id} in query to inject the selected row key. Use {uuid} to generate a unique id (or primary key). Examples:

Example

```
INSERT INTO
   app_fd_sample (id, c_clicked)
VALUES
   (
      {uuid}, {id}
   )
```

Example

```
UPDATE
   app_fd_sample
SET
   c_clicked = CONCAT(c_clicked, ',', {id})
WHERE
   id = {id}
```

Example

```
DELETE
FROM
   app_fd_myTable
WHERE
   id = {id}
```

Table & Column Naming

- For database tables created by Joget Forms, Joget adds a "c_" in front of table column names (or "t_" if your column name starts with a number) and "app_fd_" in front of database table names.
- If you use environment hash variables to store SQL query strings, in your hash variable, use "?noescape" to escape SQL query strings in JDBC binders to prevent the "<>" "not equal" operator from being converted, i.e. disables XSS prevention checking. Read here for more information..

How it works?

The special parameters {id} and {uuid} will be replaced with actual values through the use of PreparedStatement. As you can see from the example above, there is no need to encapsulate both of these special keywords with quotes.

Related JDBC Binders & Useful Links

- JDBC Options Binder
- JDBC Form Binder
- JDBC Datalist Database Binder
- Understanding JDBC Errors

Download Demo App

APP_datalist_using_jdbc_dx_kb.jwa