Spring Boot web application with JDBC – Delete Product

Overview

This tutorial will show how to delete existing products from the database. The process is similar to update but the form is not required. The steps are:

1. Select the product to be deleted from the product list
    a. Pass the product id as a parameter, e.g. http://localhost:8080/deleteProduct/?id=1
2. Delete the product by id from the database.
3. Re-load the products list.

1. Show delete links in the product list

The goal is to add an delete button for each link. Clicking the button should call and pass the product id to **`deleteProduct()`** in the controller. The result should look like this:

Id	Name	Description	Stock	Price		
1	Kettle	Steel Electric Kettle	97	€ 35.60	edit	delete
2	Fridge freezer	Fridge + freezer large	45	€ 799.00	edit	delete
3	Microsoft Surface Laptop 2	8GB ram, 512GB ssd	5	€ 1299.00	edit	delete

Add a new table column (**`<td>`** element) to the product row for the delete button and link:

```
        <td><a th:href="@{'/updateProduct/?id=' + ${product.ProductId}}" class="btn-sm btn-danger" role="button">edit</a></td>
        <td><a th:href="@{'/deleteProduct/?id=' + ${product.ProductId}}" onclick="return confirmDelete();" class="btn-sm btn-danger" role="button">delete</span></a></td>
    </tr>
</table> <!-- End table -->
```

Text version for copying:

```
<td><a th:href="@{'/deleteProduct/?id=' + ${product.ProductId}}" onclick="return confirmDelete();"
class="btn-sm btn-danger" role="button">delete</span></a></td>
```

Confirming delete

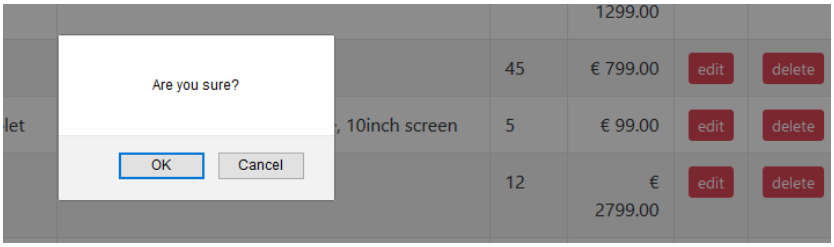
It would not be a good idea to delete without first confirming with the user – in case the button was clicked accidently. JavaScript can be used to confirm the action before calling the delete link. Note the **`onclick even handler`** in the link:

```
onclick="return confirmDelete();"
```

This calls the following JavaScript function which should be added to the end of the page body:

```
    </div> <!-- End container -->
    <script>
        // JavaScript function returns true if user clicks yes, otherwise, false
        // true - proceed (open link)
        // false - cancel
        function confirmDelete() {
            return confirm('Are you sure?');
        }
    </script>

    <!--/*/ <th:block th:include="fragments/footer :: footer"></th:block> /*/-->
</body>
</html>
```

Now whenever a delete button is clicked the user will be asked to confirm:



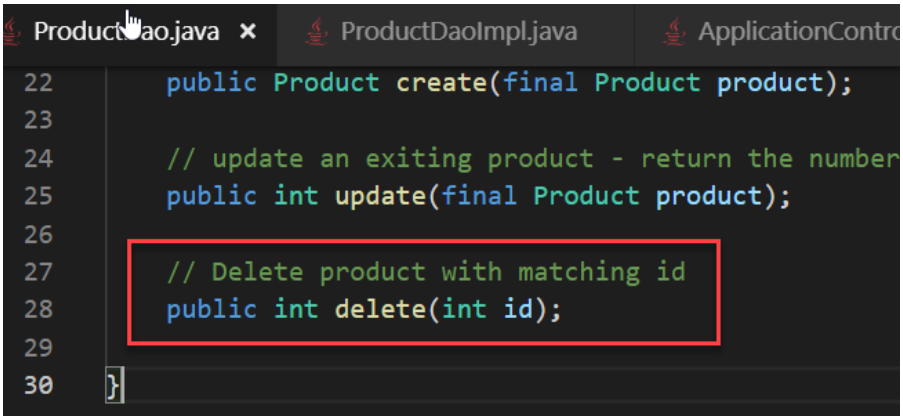
Ok will return true, allowing the link to be loaded, cancel returns false which stops /**`deleteProduct/?id=1`** from being called.

2. Product DAO update method

Add a method to **`ProductDao`** and its implementation to perform the update

First **`ProductDao`** (the interface)

The create method accepts an existing product object parameter and returns the number of rows affected by the update (usually 1).



The product implementation, **ProductDaoImpl.java**

The SQL statement required for delete. This is an SQL statement with value placeholders (**?**)

```
private final String DELETE_SQL_BY_ID = "DELETE FROM dbo.Product WHERE ProductId = ?";
```

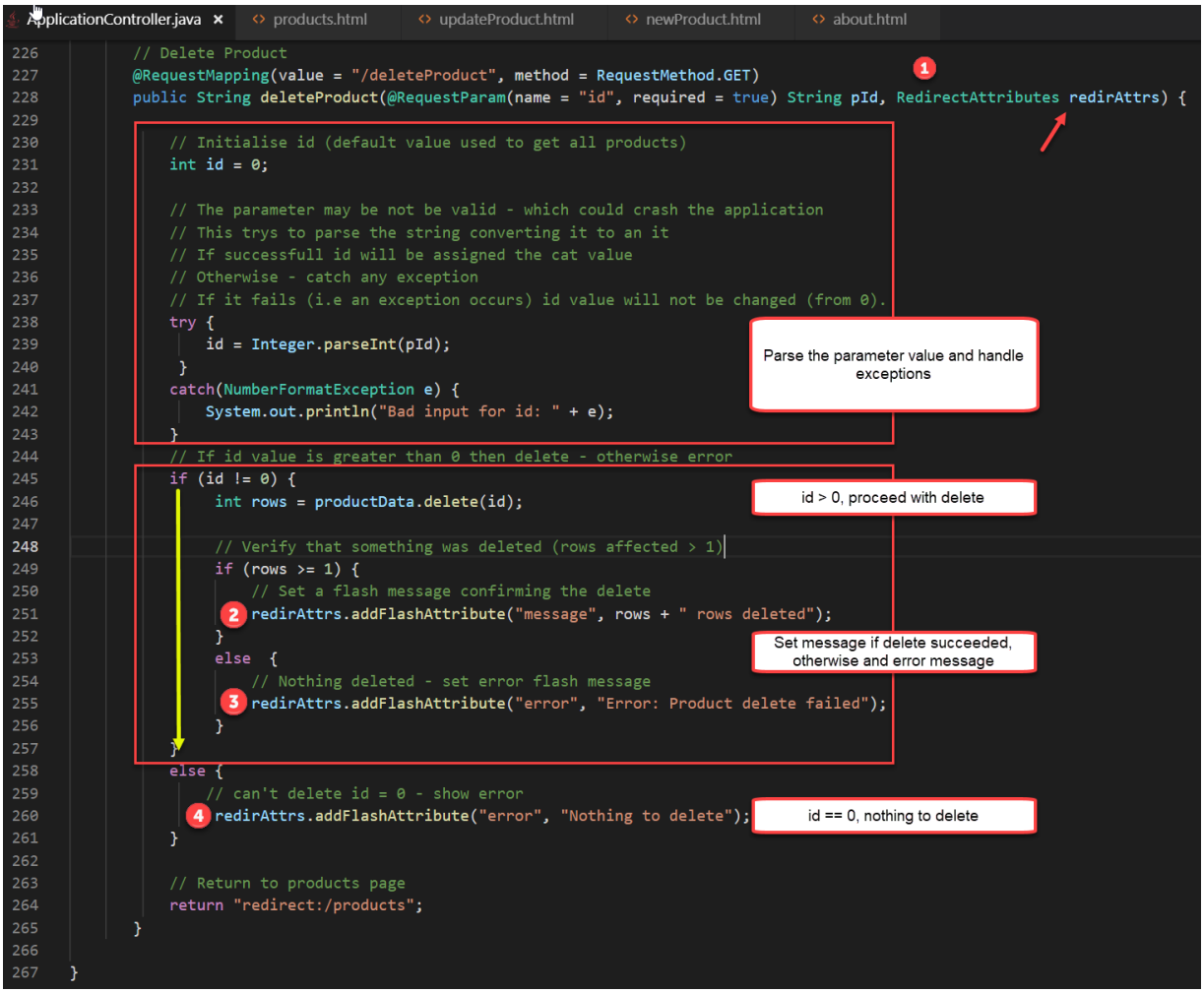
The **delete** method which will fill in the values and execute the query (**see code comments**)

```
    // Delete a product by id
    public int delete(int id) {

        // Use the delete sql, setting the id paramater
        // This method returns the number of rows affected
        return jdbcTemplate.update(DELETE_SQL_BY_ID, new Object[] {id});

    }
```

3. A Controller Method to execute the delete

This method will call the delete and redirect back to the products page. Also use **RedirectAttributes** (flash attribute) to confirm delete or report error.

1. **RedirectAttributes** allows a confirmation message to be sent to the next page via redirect.
    a. Useful for confirming actions
    b. Requires **import org.springframework.web.servlet.mvc.support.RedirectAttributes;**
    c. (1 in code) Inject **RedirectAttributes redirAttrs** as a parameter to the **deleteProduct** method
2. (2 in code) Add a flash attribute named **message** (value shows rows successfully deleted)
3. (3,4 in code) Add a flash attribute named **error** (value shows error condition – delete failed)



Copy code:

```
    // Delete Product
    @RequestMapping(value = "/deleteProduct", method = RequestMethod.GET)
    public String deleteProduct(@RequestParam(name = "id", required = true) String pId, RedirectAttributes redirAttrs) {

        // Initialise id (default value used to get all products)
        int id = 0;

        // The parameter may be not be valid - which could crash the application
        // This trys to parse the string converting it to an it
        // If successfull id will be assigned the cat value
        // Otherwise - catch any exception
        // If it fails (i.e an exception occurs) id value will not be changed (from 0).
        try {
            id = Integer.parseInt(pId);
         }
        catch(NumberFormatException e) {
            System.out.println("Bad input for id: " + e);
        }
        // If id value is greater than 0 then delete - otherwise error
        if (id != 0) {
             int rows = productData.delete(id);

             // Verify that something was deleted (rows affected > 1)
             if (rows >= 1) {
                 // Set a flash message confirming the delete
                 redirAttrs.addFlashAttribute("message", rows + " rows deleted");
             }
             else  {
                 // Nothing deleted - set error flash message
                 redirAttrs.addFlashAttribute("error", "Error: Product delete failed");
             }
        }
        else {
            // can't delete id = 0 - show error
             redirAttrs.addFlashAttribute("error", "Nothing to delete");
        }

        // Return to products page
        return "redirect:/products";
    }
```

4. Display the results (after delete)

After delete, the flash message should be displayed in the products page. These are added in **`products.html`** – if they exist (i.e. they were set in the controller before redirecting to this page).

The result after deleting a product: