File handling in Bean Shell Form Store Binder

If you are not using AppService.storeFormData to handle data storing in your Bean Shell Form Store Binder, you may face the fact that you will have to handle file storing yourself as well. The following code shown you how you can achieve it.

```
import java.io.File;
import java.util.ArrayList;
import java.util.Collection;
import java.util.HashMap;
import java.util.Map;
import org.joget.apps.app.model.AppDefinition;
import org.joget.apps.app.service.AppService;
import org.joget.apps.app.service.AppUtil;
import org.joget.apps.form.dao.FormDataDao;
import org.joget.apps.form.model.Element;
import org.joget.apps.form.model.Form;
import org.joget.apps.form.model.FormBinder;
import org.joget.apps.form.model.FormData;
import org.joget.apps.form.model.FormDataDeletableBinder;
import org.joget.apps.form.model.FormLoadBinder;
import org.joget.apps.form.model.FormLoadMultiRowElementBinder;
import org.joget.apps.form.model.FormRow;
import org.joget.apps.form.model.FormRowSet;
import org.joget.apps.form.model.FormStoreBinder;
import org.joget.apps.form.model.FormStoreMultiRowElementBinder;
import org.joget.apps.form.service.FormUtil;
import org.joget.commons.util.LogUtil;
import org.joget.commons.util.FileManager;
import org.joget.apps.form.service.FileUtil;

public FormRowSet store(Element element, FormRowSet rows, FormData formData) {
    if (rows == null) {
        return null;
    }

    try {
        String tableName = "grid_form";
        String formDefId = "grid_form";
        String foreignKey = "fk";
        String fileField = "files";

        FormDataDao formDataDao = (FormDataDao) AppUtil.getApplicationContext().getBean("formDataDao");

        Form parentForm = FormUtil.findRootForm(element);
        String primaryKeyValue = parentForm.getPrimaryKeyValue(formData);

        Map exist = new HashMap();

        //Check for deletion
        FormRowSet originalRowSet = formDataDao.find(formDefId, tableName, " WHERE " + FormUtil.
PROPERTY_CUSTOM_PROPERTIES + "." + foreignKey + " = ?", new Object[]{primaryKeyValue}, "dateCreated", false,
null, null);

        if (originalRowSet != null && !originalRowSet.isEmpty()) {
            Collection ids = new ArrayList();
            for (FormRow r : originalRowSet) {
                if (rows == null || (rows != null && !rows.contains(r))) {
                    ids.add(r.getId());

                    //delete files
                    String files = r.getProperty(fileField);
                    if (files != null && !files.isEmpty()) {
                        String[] file_paths = files.split(";");
                        for (String path : file_paths) {
                            File file = FileUtil.getFile(path, tableName, r.getId());
                            if (file != null && file.exists()) {
                                File folder = file.getParentFile();
                                FileManager.deleteFile(folder);
                                break;
```

```
                            }
                        }
                    }
                } else {
                    exist.put(r.getId(), r);
                }
            }
            if (ids.size() > 0) {
                formDataDao.delete(formDefId, tableName, ids.toArray(new String[ids.size()]));
            }
        }
        Date currentDate = new Date();

        for (FormRow row : rows) {
            //set id and update data
            String id = row.getId();
            if (id != null && exist.containsKey(id)) {
                FormRow temp = new FormRow();
                temp.putAll(row);

                row.putAll(exist.get(id));
                row.putAll(temp);
            } else if (id == null || id.isEmpty()) {
                row.setId(UuidGenerator.getInstance().getUuid());
                // set created date
                row.setDateCreated(currentDate);
            }
            // set modified date
            row.setDateModified(currentDate);
            //set foreign key
            row.put(foreignKey, primaryKeyValue);
        }

        //Check and update filename if exists
        FileUtil.checkAndUpdateFileName(rows, tableName, primaryKeyValue);

        // save data
        formDataDao.saveOrUpdate(formDefId, tableName, rows);

        //store files
        FileUtil.storeFileFromFormRowSet(rows, tableName, primaryKeyValue);
    } catch (Exception e) {
        LogUtil.error("File Handling in Bean Shell Binder Sample - Grid Form", e, "Store data error!!");
    }
    return rows;
}

//call store method with injected variable
return store(element, rows, formData);
```

Sample app: APP_beanshell_file_sample-1-20220901154812.jwa