Questions 2

What is the question framework?

The Kuali Question framework provides a set of classes for creating questions that can be asked of a user. For example, this framework is used when you click the "Cancel" button on a document. For the most part, this is another step that can be used to ask the important "are you sure you meant to do that" questions.

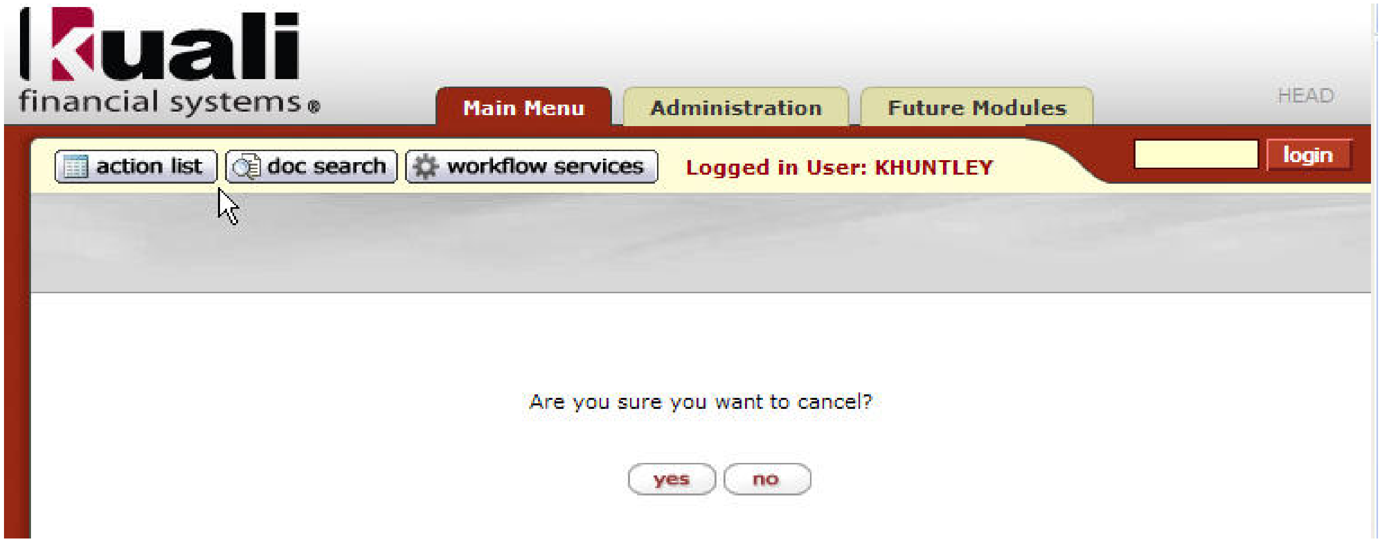
Some other examples of questions that may be useful to ask would be:

- *Are you sure you want to delete all these values?*
- *Are you sure you want to remove this permission?*
- *Are you sure you want to close this document?*

Asking confirmation questions using pre-rules
If the developer's goal is to ask a confirmation question of the user prior to approving, blanket approving, routing, or performing a route report on a document, then consult the pre-rules page, as it allows for easier implementation question asking.

How to Implement a Confirmation Question within a Struts action

Looking at the Cancel Question in the document framework



Logged in User: KHUNTLEY

Are you sure you want to cancel?

The following code block from the `org.kuali.core.web.struts.action.KualiDocumentActionBase.cancel(ActionMapping, ActionForm, HttpServletRequest, HttpServletResponse)` method is used to illustrate the code necessary to implement a question: *See the full method for a complete example of the code used to perform this functionality.*

```
1        Object question =
request.getParameter(RiceConstants.QUESTION_INST_ATTRIBUTE_NAME);
5        // logic for cancel question
6        if (question == null) {
7            // ask question if not already asked
8            return this.performQuestionWithoutInput(mapping, form, request, response,
Constants.DOCUMENT_CANCEL_QUESTION,

kualiConfigurationService.getPropertyString("document.question.cancel.text"),
Constants.CONFIRMATION_QUESTION,
                                                     Constants.MAPPING_CANCEL, "");
9        }
10       else {
11           Object buttonClicked =
request.getParameter(Constants.QUESTION_CLICKED_BUTTON);
12           if ((Constants.DOCUMENT_CANCEL_QUESTION.equals(question)) &&
ConfirmationQuestion.NO.equals(buttonClicked)) {
13               // if "no" button clicked just reload the doc
14               return mapping.findForward(RiceConstants.MAPPING_BASIC);
15           }
16           // else go to cancel logic below
17       }
```

Step 1 (Line 1) - Within your action look for the question parameter

```
Object question = request.getParameter(RiceConstants.QUESTION_INST_ATTRIBUTE_NAME);
```

Step 2 (Line 6) - Check if you are coming into this method from a question already

```
if (question == null)
```

Explanation: The `KualiDocumentActionBase.cancel(...)` method is invoked twice. Recall that this method is a Struts action handler. It's called once to ask the question (i.e. when the cancel button is clicked), and once after the user has answered the question (i.e. when yes or no has been clicked).

When the cancel button's clicked, the request does not include a parameter named with the value of `RiceConstants.QUESTION_INST_ATTRIBUTE_NAME` (i.e. "questionIndex"). However, when the response is rendered, the HTML output contains a form with the parameter. The code will enter line 8.

Therefore, when the user clicks on yes or no, the resulting request will contain the "questionIndex" parameter, so the code will enter line 11.

Step 3 (Line 8) - Calling the question framework to ask the question

```
return this.performQuestionWithoutInput(... RiceConstants.DOCUMENT_CANCEL_QUESTION,
kualiConfigurationService.getPropertyString("document.question.cancel.text"),
                                        RiceConstants.CONFIRMATION_QUESTION,
RiceConstants.MAPPING_CANCEL, "");
```

```
protected ActionForward performQuestionWithoutInput(... String questionId, String
questionText, String questionType, String caller, String context)
```

Explanation: As can be seen above this is for a question without input you can also process input with the preformQuestionWithInput method. The method parameters displayed above for performQuestionWithoutInput are as follows:

- questionId : the questionId is the unique identifier for the question
- questionText: the questionText is the message displayed to the user as a question
- questionType: the questionType in this example is ConfirmationQuestion but it is possible to define other question types through spring
- caller: the caller is the method that the question is being called from so that control can be returned to this method once the question is answered
- context: the context which can be set

Step 4 (Line 9-16) Processing the answer from the question

```
else {
        Object buttonClicked =
request.getParameter(Constants.QUESTION_CLICKED_BUTTON);
        if ((Constants.DOCUMENT_CANCEL_QUESTION.equals(question)) &&
ConfirmationQuestion.NO.equals(buttonClicked)) {
                 // if "No" button clicked just reload the doc
                 return mapping.findForward(Constants.MAPPING_BASIC);
        }
        // else go to cancel logic below
}
```

> **Explanation***: This is where you want to do any special processing based on the button that was clicked, in this case it either returns to the doc or does some special cancel logic based on whether the button is "No" or not.*

NOTE: Questions can also be triggered through prerules, and are more appropriately called there when the question has to do with more of a pre-rule/ data massaging type of function. See the PreRule documentation above for more details.

Unable to render {include} The included page could not be found.