AcademiCS Application Framework

Specifications for the Fifth Set of Milestones and Deliverables

The fifth set of milestones/deliverables is concerned only with the basics of documents and document presentation in the demonstration application called The Big Pixel.

1. Glossary

No new terminology needs to be introduced for this set of milestones/deliverables.

2. Engineering Design

The relationships between the various classes that must be implemented for the first set of milestones/deliverables is illustrated in the UML class diagram (that is available as an SVG file). In addition to the specifications in that diagram, the classes/interfaces must comply with the following specifications.

2.1 The `AbstractEditableWritingWorker` Class

This class is a partial encapsulation of a `SwingWorker` that can be used to write a document to the file system. The parameter `D` denotes the class of the document.

The `done()` method must be empty.

The `doInCallersThread()` method must call the `writeInCallersThread()` method.

2.2 The `EditableWritingWorkerFactory` Interface

This interface describes the capabiltiies of a factory that can create an `AbstractEditableWritingWorker`.

2.3 The `StringWritingWorker` Class

This class is an encapsulation of an `AbstractEditableWritingWorker` that can write a `String` representation of a document. The parameter `D` denotes the class of the document.

The `doInBackground()` method must call the `writeInCallersThread()` method.

The `writeInCallersThread()` method must write the document to the file system using a UTF-8 encoding.

2.4 The `StringWritingWorkerFactory` Class

The `StringWritingWorkerFactory` class is an encapsulation of a factory that can create `StringWritingWorker` objects. The parameter `D` denotes the class of the document.

2.5 The `AbstractGeneralSave` Class

The `AbstractGeneralSave` class is an abstract action that starts the process of saving a document. The parameter `D` denotes the class of the document and the parameter `F` denotes the class of the writer factory.

It must listen to `DOCUMENT_ACTIVATED`, `DOCUMENT_CLOSED`, `DOCUMENT_EDITED` and `FILE_CHANGED` property change events in order to enforce the appropriate work flow. The `propertyChange()` method must enforce this work flow.

The `saveUsingWorker()` method must create an `AbstractEditableWritingWorker`, create an associated `BackgroundTaskDialog`, execute the dialog, and set the state of the document appropriately.

The `saveUsingCallersThread()` method must create an `AbstractEditableWritingWorker`, invoke its `doInCallersThread()` method, and set the state of the document appropriately.

2.6 The `AbstractSave` Class

The `AbstractSave` class is an abstract action that starts the process of saving a document with its current name. The parameter `D` denotes the class of the document and the parameter `F` denotes the class of the writer factory.

Its `actionPerformed()` method must invoke its `saveUsingWorker()` method.

2.7 The `AbstractSaveAs` Class

The `AbstractSave` class is an abstract action that starts the process of saving a document with a new name. The parameter `D` denotes the class of the document and the parameter `F` denotes the class of the writer factory.

Its `actionPerformed()` method must prompt the user to select a file (using its `JFileChooser`) and invoke its `saveUsingWorker()` method. It must also notify the `DocumentManager` that the `File` has changed and update the `CURRENT_DIRECTORY` in the `Configuration`.

2.8 The `SaveString` Class

The `SaveString` class is an encapsulation of a concrete action that saves a `String` representation of a document. The parameter `D` denotes the class of the document.

2.9 The `SaveAsString` Class

The `SaveString` class is an encapsulation of a concrete action that prompts the user for a `File` to save to and then saves a `String` representation of a document in that `File`. The parameter `D` denotes the class of the document.