OmniWebService

Class OmniWebService_dev

```
java.lang.Object
  └ OmniWebService.OmniWebService_dev
```

All Implemented Interfaces:

ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev, java.rmi.Remote

---

public class **OmniWebService_dev**
extends java.lang.Object
implements ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev

$Id: OmniWebService_dev.html,v 1.1 2004/11/07 07:12:22 c2durdle Exp $ CSC408 Software Engineering OmniEditor Project Mark Isaac Dylan Durdle Yani Ioannou Ashish Kaila Sandy Lam Gilles Lising OmniEditor plugin

Version:

$Revision: 1.1 $ $Log: OmniWebService_dev.html,v $ Revision 1.1 2004/11/07 07:12:22 c2durdle JavaDoc update Revision 1.7 2004/11/07 06:56:41 c2durdle *** empty log message *** Revision 1.6 2004/11/07 06:05:29 c2durdle *** empty log message *** Revision 1.5 2004/11/07 02:22:56 c2durdle isolated configuration Revision 1.4 2004/11/06 10:09:32 c2durdle *** empty log message *** Revision 1.3 2004/11/06 08:51:19 c2durdle *** empty log message *** Revision 1.20 2004/11/06 08:04:14 c2durdle *** empty log message *** Revision 1.19 2004/11/06 05:49:31 c2durdle fixed delete() Revision 1.18 2004/11/06 04:24:40 c4ioanno Sorry...can't stand those unused imports. Revision 1.17 2004/11/06 04:00:22 c2durdle *** empty log message *** Revision 1.10 2004/11/06 03:36:07 c2durdle updated structure Revision 1.9 2004/11/06 01:30:38 c2durdle Added ErrorCodes.LINE_DOES_NOT_EXIST Revision 1.8 2004/11/06 00:12:45 c2durdle *** empty log message *** Revision 1.7 2004/11/05 18:53:41 c2durdle *** empty log message *** Revision 1.6 2004/11/05 17:31:19 c2durdle fixed hashcode reader Revision 1.5 2004/11/05 15:55:32 c2durdle *** empty log message *** Revision 1.4 2004/11/05 06:57:25 c4ioanno Sorry, couldn't stand those eclipse warnings anymore. Got rid of unused imports. Revision 1.3 2004/11/04 15:11:18 c2durdle *** empty log message *** Revision 1.2 2004/11/04 00:41:58 c2durdle *** empty log message *** Revision 1.1 2004/11/03 17:43:26 c2durdle Virgin version Revision 1.7 2004/11/03 04:37:30 c2durdle *** empty log message *** Revision 1.6 2004/11/03 02:42:31 c2durdle *** empty log message *** Revision 1.5 2004/11/02 05:30:03 c2durdle *** empty log message *** Revision 1.11 2004/11/02 05:20:03 c2durdle *** empty log message *** Revision 1.4 2004/11/02 04:26:05 c2durdle Minor changes Revision 1.8 2004/11/01 07:56:34 c2durdle *** empty log message *** Revision 1.7 2004/11/01 07:02:47 c2durdle ErrorCodes Revision 1.6 2004/11/01 06:04:25 c2durdle JavaDocs added Revision 1.5 2004/11/01 03:18:23 c2durdle *** empty log message *** Revision 1.4 2004/10/26 08:05:23 c2durdle Updated the CVS keyword Type comment. This is a better comment to hopefully show how cvs keywords are useful. Revision 1.3 2004/10/26 07:34:28 c2durdle comments :$ Overall

conditions: - line number starts from 1

Author:

Dylan Durdle, ddurdle@gmail.com

Constructor Summary

OmniWebService_dev()

Method Summary

long	**addLine**(java.lang.String documentHash, long line, long userID, java.lang.String toAdd, long sequenceNo) Add lines of text document.
java.lang.String	**browse**() Browse the server looking for uploaded/open documents.
long	**closeDocument**(java.lang.String documentHash) Close and remove all traces of the document.
long	**delete**(java.lang.String documentHash, long line, long userID, int startPosition, int endPosition, long sequenceNo) Delect highlighted text.
long	**deleteLine**(java.lang.String documentHash, long line, long userID, long sequenceNo) Delete lines of text document Preconditions: the document represented by "documentHash" exists the "line" exists in the document in relation to the sequence number provided by the client Postconditions: sequence number returned represents the transaction log of this operation
java.lang.String	**getDocument**(java.lang.String document) Fetch reference to document.
java.lang.String	**getHashCode**(java.lang.String document) Get the hashcode of a document.
java.lang.String	**getLine**(java.lang.String documentHash, long line, long sequenceNo) Fetch contents of a single line.
java.lang.String	**getLocks**(java.lang.String documentHash) Get a list of locks on the document.
java.lang.String	**getLocksAndUIDs**(java.lang.String documentHash) Get a list of locks on the document with their respective UIDs who have locked them.
long	**getSequenceNo**(java.lang.String documentHash, long sequenceNoOld, long lineID) Fetch the current sequence number.

long	**insert**(java.lang.String documentHash, long line, long userID, int startPosition, int endPosition, java.lang.String toInsert, long sequenceNo) Insert text to the current position.
java.lang.String	**loadDocument**(java.lang.String documentName, java.lang.String document) Create and load a document on the server. Preconditions: "document" doesn't already exist.
long	**lockLine**(java.lang.String documentHash, long line, long userID, long sequenceNo) Lock or renew the lock for the line for exclusive editing.
static void	**main**(java.lang.String[] args) Test Driver
java.lang.String	**saveDocument**(java.lang.String documentHash) Download/Save the document to the local client's buffer.
long	**unlockLine**(java.lang.String documentHash, long line, long userID, long sequenceNo) Release a lock before it times out Preconditions: the document represented by "documentHash" exists the "line" exists in the document with regard to the current sequence number in relation to the sequence number provided by the client a lock already exists on the "line" by "userID" and has not expired (LOCK_TIMEOUT) Postconditions: the lock is released for others to acquire lineID that is returned is an exact reference to the line relative to "line" regard to the current sequence number in relation to the sequence number provided by the client
long	**updateLine**(java.lang.String documentHash, long line, long userID, java.lang.String updatedLine, long sequenceNo) Update line the line with the replacement line.
java.lang.String	**updates**(java.lang.String documentHash, long sequenceNo) Updates for the document.

Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

OmniWebService_dev

public **OmniWebService_dev**()

Method Detail

main

```
public static void main(java.lang.String[] args)
                throws java.lang.Exception
```

Test Driver

Throws:
`java.lang.Exception`

---

loadDocument

```
public java.lang.String loadDocument(java.lang.String documentName,
                                     java.lang.String document)
```

Create and load a document on the server.

- Preconditions:
  - "document" doesn't already exist.
  - "document" is a string with new lines deliminated with \n
- Postconditions:
  - document is loaded on the server and can be referenced using the documentHash and via the "documents" table.

Specified by:
`loadDocument` in interface `ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:
`documentName` - is the name of the document file
`document` - contains the document in a string with new lines deliminated with \n

Returns:
hashcode as a string of open document

Throws:
`java.lang.Exception` - if the table already exists

---

updates

```
public java.lang.String updates(java.lang.String documentHash,
                                long sequenceNo)
```

Updates for the document.

- Preconditions:
  - Document doesn't already exist.
- Postconditions:
  - document is loaded on the server and can be referenced using the documentHash and via the "documents" table.

Specified by:
`updates` in interface
`ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:
`documentHash` - the hashcode representing the open document on the system
`sequenceNo` - the state of the user's document

Returns:
a String containing the updates in this form: lastSequenceNumber + "\n" + the message of changes deliminated by \n

---

getLine

```
public java.lang.String getLine(java.lang.String documentHash,
                                long line,
                                long sequenceNo)
```

Fetch contents of a single line.

- Preconditions:
  - the document represented by "documentHash" exists
  - "line" <= length of the document represented by "documentHash" "line" <= length of the document represented by "documentHash"
- Postconditions:
  - contents for line is returned via a string - the actual line returned will be offsetted from the original "line" to account for operations that have occurred between "sequenceNo" and the document's current sequence number.

Specified by:
`getLine` in interface
`ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:
`documentHash` - the hashcode representing the open document on the system
`line` - the line number
`sequenceNo` - the state of the user's document

Returns:
a String containing the line or null if preconditions fail

---

getLocks

```
public java.lang.String getLocks(java.lang.String documentHash)
```

Get a list of locks on the document.

- Preconditions:
  - the document represented by "documentHash" exists
- Postconditions:

- a string composed of "linenumber\n" for each line that is locked which exist and haven't expired by the time the function is run

Specified by:
`getLocks` in interface `ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Returns:
a string compose of "linenumber\n" for each line that is locked

---

saveDocument

```
public java.lang.String saveDocument(java.lang.String documentHash)
```

Download/Save the document to the local client's buffer.

- Preconditions:
  - the document represented by "documentHash" exists
- Postconditions:
  - a string composed of the name of the file + "\n" + the document contained in a string with lines deliminated by \n, up-to-date at the call of the function

Specified by:
`saveDocument` in interface `ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:
`documentHash` - the hashcode representing the open document on the system

Returns:
the name of the file + "\n" + the document contained in a string with lines deliminated by \n

---

getDocument

```
public java.lang.String getDocument(java.lang.String document)
```

Fetch reference to document.

- Preconditions:
  - the "document" exists on the server
  - "document" is a string with new lines deliminated with \n
- Postconditions:
  - the "documentHash" represented by "document" as loaded already on the server

Specified by:
`getDocument` in interface `ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:
`document` - contains the document in a string with newlines deliminated with \n

Returns:

documentHash the hashcode representing the open document on the system

---

getHashCode

```
public java.lang.String getHashCode(java.lang.String document)
```

Get the hashcode of a document.

- Preconditions:
  - "document" is a string with new lines deliminated with \n
- Postconditions:
  - the "documentHash" represented by "document"

Specified by:
`getHashCode` in interface `ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:
`document` - contains the document in a string with newlines deliminated with \n

Returns:
documentHash the hashcode representing the open document on the system

---

getSequenceNo

```
public long getSequenceNo(java.lang.String documentHash,
                          long sequenceNoOld,
                          long lineID)
```

Fetch the current sequence number.

- Preconditions:
  - the document represented by "documentHash" exists
  - the "sequenceNoOld" document represented by "documentHash" exists
- Postconditions:
  - the last transaction involving "lineID"

Specified by:
`getSequenceNo` in interface `ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:
`documentHash` - the hashcode representing the open document on the system
`sequenceNoOld` - represents the sequence number relayed by the client
`lineID` - of last line affected in our transaction

Returns:
latest sequence number or an appropriate error code

---

closeDocument

```
public long closeDocument(java.lang.String documentHash)
```

Close and remove all traces of the document.

- Preconditions:
  - the document represented by "documentHash" exists
- Postconditions:
  - all existance of the document represented by "documentHash" is removed from the server

Specified by:
`closeDocument` in interface `ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:
`documentHash` - the hashcode representing the open document on the system

Returns:
corresponding sequence number or an appropriate ErrorCode

---

browse

```
public java.lang.String browse()
```

Browse the server looking for uploaded/open documents.

- Preconditions:
  - none
- Postconditions:
  - a list of documents available (and open) on the server following the format 'document_hash','document_name','last_modified_timestamp'

Specified by:
`browse` in interface `ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Returns:
a String composed of 'document_hash','document_name','last_modified_timestamp'

---

getLocksAndUIDs

```
public java.lang.String getLocksAndUIDs(java.lang.String documentHash)
```

Get a list of locks on the document with their respective UIDs who have locked them.

- Preconditions:
  - the document represented by "documentHash" exists
- Postconditions:

- a string composed of "linenumber,userid\n" for each line that is locked which exist and haven't expired by the time the function is run

Specified by:

`getLocksAndUIDs` in interface `ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:

`documentHash` - representing the documentHash desired for opening

Returns:

a string compose of "linenumber,userid\n" for each line that is locked

---

lockLine

```
public long lockLine(java.lang.String documentHash,
                     long line,
                     long userID,
                     long sequenceNo)
```

Lock or renew the lock for the line for exclusive editing.

- Preconditions:
  - the document represented by "documentHash" exists
  - the "line" exists in the document with regard to the current sequence number in relation to the sequence number provided by the client
  - a lock doesn't already exist on the "line", except
  - if the lock has expired, or
  - if the line is already locked by the user.
- Postconditions:
  - a lock is placed on "line" for LOCK_TIMEOUT seconds
  - lineID that is returned is an exact reference to the line relative to "line" regard to the current sequence number in relation to the sequence number provided by the client

Specified by:

`lockLine` in interface `ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:

`documentHash` - the hashcode representing the open document on the system

`line` - the line number

`userID` - the userid for the lock

`sequenceNo` - the state of the user's document

Returns:

corresponding lineID or an appropriate ErrorCode

Throws:

`java.lang.Exception` - if document doesn't exist

---

unlockLine

```
public long unlockLine(java.lang.String documentHash,
                       long line,
                       long userID,
                       long sequenceNo)
```

Release a lock before it times out

- Preconditions:
  - the document represented by "documentHash" exists
  - the "line" exists in the document with regard to the current sequence number in relation to the sequence number provided by the client
  - a lock already exists on the "line" by "userID" and has not expired (LOCK_TIMEOUT)
- Postconditions:
  - the lock is released for others to acquire
  - lineID that is returned is an exact reference to the line relative to "line" regard to the current sequence number in relation to the sequence number provided by the client

Specified by:

`unlockLine` in interface `ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:

`documentHash` - representing the documentHash desired for releasing the lock
`line` - representing the line to lock
`userID` - representing the user to use for locking
`sequenceNo` - the state of the user's document

Returns:

corresponding lineID or an appropriate ErrorCode

---

updateLine

```
public long updateLine(java.lang.String documentHash,
                       long line,
                       long userID,
                       java.lang.String updatedLine,
                       long sequenceNo)
```

Update line the line with the replacement line.

- Preconditions:
  - the document represented by "documentHash" exists
  - the "line" exists in the document with regard to the current sequence number in relation to the sequence number provided by the client
  - the user has a lock on the line or the line is available to be locked
- Postconditions:
  - the "line" is replaced with "updatedLine"
  - sequence number returned represents the transaction log of this operation

Specified by:

`updateLine` in interface

`ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:

`documentHash` - representing the documentHash desired for adding to
`line` - as the marker to begin adding to the document
`userID` - to which to add the lines with
`updatedLine` - representing 1 to many lines of text to add
`sequenceNo` - the state of the user's document

Returns:

corresponding sequence number or an appropriate ErrorCode

Throws:

`java.lang.Exception` - if update spans multiple lines

---

insert

```
public long insert(java.lang.String documentHash,
                   long line,
                   long userID,
                   int startPosition,
                   int endPosition,
                   java.lang.String toInsert,
                   long sequenceNo)
```

Insert text to the current position.

- Preconditions:
  - the document represented by "documentHash" exists
  - the "line" exists in the document with regard to the current sequence number in relation to the sequence number provided by the client
  - the user has a lock on the line or the line is available to be locked
  - startPosition and endPosition are correct lower and upper bounds on the line
  - startPosition < endPosition
  - startPosition > 0 and endPosition > 0
- Postconditions:
  - the text between startPosition and endPosition on "line" is replaced with "toInsert"
  - sequence number returned represents the transaction log of this operation
  - if the inserted text exceeds the bounds, the line expands in length to compensate

Specified by:

`insert` in interface
`ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:

`documentHash` - the hashcode representing the open document on the system
`line` - the line for this operation
`userID` - to which to add the lines with
`startPosition` - the start position for this operation
`endPosition` - the end position for this operation
`toInsert` - the string to insert
`sequenceNo` - the state of the user's document

Returns:

corresponding sequence number or an appropriate ErrorCode

Throws:

`java.lang.Exception` - if endPosition doesn't exist

---

delete

```
public long delete(java.lang.String documentHash,
                   long line,
                   long userID,
                   int startPosition,
                   int endPosition,
                   long sequenceNo)
```

Delect highlighted text.

- Preconditions:
  - the document represented by "documentHash" exists
  - the "line" exists in the document with regard to the current sequence number in relation to the sequence number provided by the client
  - the user has a lock on the line or the line is available to be locked
  - startPosition and endPosition are correct lower and upper bounds on the line
  - startPosition < endPosition
  - startPosition > 0 and endPosition > 0
- Postconditions:
  - the text between startPosition and endPosition on "line" is replaced with "toInsert"
  - sequence number returned represents the transaction log of this operation

Specified by:

`delete` in interface `ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:

`documentHash` - the hashcode representing the open document on the system

`line` - the line for this operation

`userID` - to which to add the lines with

`startPosition` - the start position for this operation

`endPosition` - the end position for this operation

`sequenceNo` - the state of the user's document

Returns:

corresponding sequence number or an appropriate ErrorCode

Throws:

`java.lang.Exception` - if endPosition doesn't exist

---

addLine

```
public long addLine(java.lang.String documentHash,
                    long line,
                    long userID,
                    java.lang.String toAdd,
```

```
                long sequenceNo)
```

Add lines of text document.

- Preconditions:
  - the document represented by "documentHash" exists
  - the "line" is not > the maximum line number + 1 in the document in relation to the sequence number provided by the client
  - toAdd is a line or multiple lines of text (deliminated by \n) line
- Postconditions:
  - the lines defined by "toAdd" are inserted into the document at "line", sequentially if the "toAdd" is multiple lines (sequentially in line numbers and line IDs)
  - sequence number returned represents the transaction log of this operation

Specified by:

`addLine` in interface `ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:

`documentHash` - representing the documentHash desired for adding to
`line` - as the marker to begin adding to the document
`userID` - to which to add the lines with
`toAdd` - representing 1 to many lines of text to add
`sequenceNo` - the state of the user's document

Returns:

corresponding sequence number or an appropriate ErrorCode

---

deleteLine

```
public long deleteLine(java.lang.String documentHash,
                       long line,
                       long userID,
                       long sequenceNo)
```

Delete lines of text document

- Preconditions:
  - the document represented by "documentHash" exists
  - the "line" exists in the document in relation to the sequence number provided by the client
- Postconditions:
  - sequence number returned represents the transaction log of this operation

Specified by:

`deleteLine` in interface `ca.utoronto.cdf.werewolf.axis.c408h015.OmniServer_dev_jws.OmniServer_dev`

Parameters:

`documentHash` - representing the documentHash desired for adding to
`line` - as the marker to begin adding to the document
`userID` - to which to add the lines with

`sequenceNo` - the state of the user's document

Returns:

corresponding sequence number or an appropriate ErrorCode