µWeb - Bug #557

Record.Save() should utilize a single transaction

2011-12-15 16:32 - Elmer de Looff

Status:	Closed	**Start date:**	2011-12-15
Priority:	High	**Due date:**	
Assignee:	Elmer de Looff	**% Done:**	100%
Category:	Model	**Estimated time:**	0.00 hour
Target version:		**Spent time:**	2.50 hours

Description

Currently, saving a Record and its foreign keys happens in separate methods. This means that a tree of items could be only partially saved, leading at the very least to wrong expectations, and at worst actual data corruption.

Save() should happen in a single transactions, and all of it should be successful or a failure.

Associated revisions

Revision 3221:cd633316ce0d - 2011-12-16 10:18 - Elmer de Looff

Added single-transaction Save() as major feature. This resolves #557. A Save() where child elements are saved, will now be processed atomically. Also fixed a bug where creating the SQL-record for a Record object would trigger a full deep-load. VersionedRecord now has a Create method that is safe from race conditions (via single-transaction Save that acquires the next-in-line recordKey.

Revision 141:08d609ae8b9b - 2011-12-16 10:18 - Elmer de Looff

Added single-transaction Save() as major feature. This resolves #557. A Save() where child elements are saved, will now be processed atomically. Also fixed a bug where creating the SQL-record for a Record object would trigger a full deep-load. VersionedRecord now has a Create method that is safe from race conditions (via single-transaction Save that acquires the next-in-line recordKey.

History

#1 - 2011-12-16 11:21 - Elmer de Looff

- Status changed from In Progress to Closed

- % Done changed from 30 to 100

This has been fixed and verified as of r3221.

Record.Save() now opens a transaction and within this, performs the saving of all foreign-related records and itself. If any of these fail, the transaction will automatically roll back.

Also with this bugfix, are a number of small fixes to the Save process. Creating an sql-safe representation from the Record used to trigger a comprehensive deep-load of the record to be saved. This no longer occurs, since we iterate using the superclass instead.