‘Versioning’ in Jade

Introduction:

Reorgs: It has always been the case that when . . .

a) There is data in a schema’s database, and

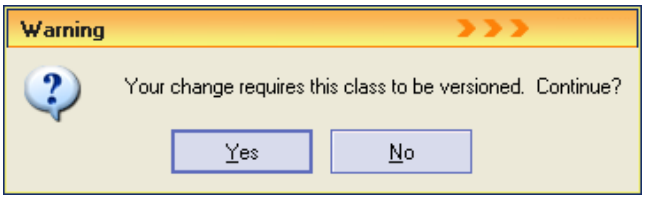
b) A change is made to the structure of that data (e.g. a new property added to a class)

. . . Jade will note that a ‘reorg’ (i.e. a database reorganisation) will be needed before applications in the schema can be run. The ‘traffic light’ tool on the toolbar turns red. Clicking that tool initiates a reorg dialog. If the developer intends to make several successive changes the reorg can be deferred until all are completed.

Versioning: With Jade 6.1.06 a feature called ‘Versioning’ was added. Versioning builds upon the reorg foundation by allowing the ‘current’ (i.e. original) schema and its applications to continue to be run. Meanwhile, schema changes and development can continue to be made inside the same database but are made to a cloned version called the ‘latest’ version. When all is ready and tested the changes can be ‘instantiated’. They become consolidated into the current version and then there is only one, updated version, which is now the current version.

Versioning is especially useful when a live schema must be updated but is required to be on-line 24/7 and therefore can be shut down only for a few seconds at a quiet moment to ‘instantiate’ the changes, that is to make them the active schema

When you make changes that require a Reorg Jade will also invoke versioning. You will see . . .



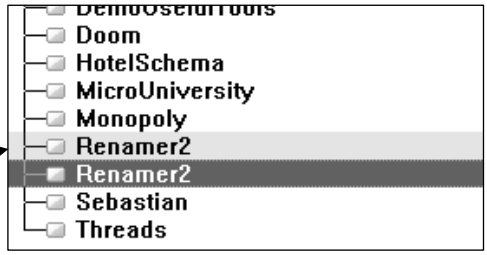
What to do?

For the teacher of initial classes in Jade versioning can be a nuisance. It is an intrusion which confronts the student even during the development of her first, small Jade application. However, for professional developers it is an improvement on reorg and it cannot be turned off.

This note has been written to help you manage the ‘versioning’ behaviour of Jade.

Continuing to work on the new, ‘Latest’ version:

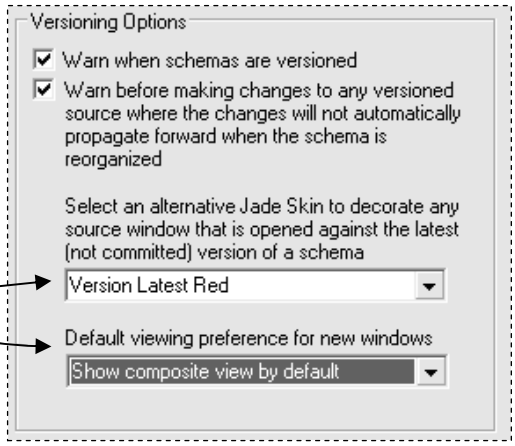
After versioning, the *Schema Browser* will show the schema name twice. The new (latest) version—the one including your changes—has a pale green bar.



In a similar manner and as an option, the *Class Browser* can be made to show both versions of classes and methods. Give the command Edit | Preferences | Miscellaneous and Select . . .

A change to a class can propagate to other classes (such as collection classes, for example) in a ‘ripple effect’ if they are dependent upon the change.

You can ask Jade to change the colour of the development environment’s title bars.



What options do you have?

(i) **To take no action at this point:** Continue to further develop the latest version and, if necessary, run (or keep running) the 'current' version.

Or

(ii) **To Reorg** (reorganise) the schema. This will instantiate the changes and thus collapse the two versions into one, 'current' version.

You reorganise a schema as usual by clicking on the red traffic-light Reorg symbol on the toolbar.[1]

Of course, since each such reorg commits the changes it makes sense only when you have no need to employ both the 'current' amd the 'latest' version.



or, more rarely, you can . . .

(iii) Abandon the 'latest version'.

To do this, give the menu command, '**Schema | Unversion**'.

Before the **Unversion** step you might make a Schema | Extract of the 'Latest' version. That will preserve a copy of the changes in case you wish to revert to them.

Versioning is a marvellous feature for professional developers but seems intrusive for complete beginners and developers who have no wish at all to keep a current version running while they develop a newer version.

In a small or prototype development (or a teaching environment) one way to minimise versioning interruptions is to build in two Jade methods; one to erase all persistent data[2] and another to rebuild the initial state of data. The latter may be a necessity anyway; it is common to build a test database off an external file of test-data with such a 'build' method.

---

[1] It is possible to defer a Reorg until many changes have been made. However, with versioning, to reorg is to commit the changes and revert to one schema. In principle then: if versioning is in force, hold off on the re-org until you are ready to commit all the changes and are ready to return to a single, consolidated schema.

[2] If there is no persistent data in the database your changes will not trigger a Reorg and thus not require Versioning.