JSFoundSetUpdater

Feb 17, 2024 02:33

Supported Clients

SmartClient WebClient NGClient

Methods Summary

Boolean	next()	Go to next record in this updater, returns true if successful.
Boolean	performUpdate()	Do the actual update in the database, returns true if successful.
void	resetIterator()	Start over with this iterator 'next' function (at the foundset selected record).
Boolean	setColumn(name, value)	Set the column value to update, returns true if successful.

Methods Details

next()

```
Go to next record in this updater, returns true if successful.
NOTE: this method doesn't take into account deletes and inserts that may happen at same time. For more reliable
iterator see foundset.forEach
```

Returns

Boolean true if proceeded to next record, false otherwise

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
controller.setSelectedIndex(1)
var count = 0
var fsUpdater = databaseManager.getFoundSetUpdater(foundset)
while(fsUpdater.next())
{
        fsUpdater.setColumn('my_flag',count++)
}
```

performUpdate()

```
Do the actual update in the database, returns true if successful. It will first try to save all editing records
(from all foundsets), if cannot save will return false before doing the update.
There are 3 types of possible use with the foundset updater
1) update entire foundset by a single sql statement; that is not possible when the table of the foundset has
tracking enabled then it will loop over the whole foundset.
   When a single sql statement is done, modification columns will not be updated and associated Table Events
won't be triggered, because it does the update directly in the database, without getting the records.
  NOTE: this mode will refresh all foundsets based on same datasource
2) update part of foundset, for example the first 4 row (starts with selected row)
3) safely loop through foundset (starts with selected row)

after the perform update call there are no records in edit mode, that where not already in edit mode, because
all of them are saved directly to the database,
or in mode 1 the records are not touched at all and the database is updated directly.
```

Returns

Boolean true if succeeded, false if failed.

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
//1) update entire foundset
var fsUpdater = databaseManager.getFoundSetUpdater(foundset)
fsUpdater.setColumn('customer_type',1)
fsUpdater.setColumn('my_flag',0)
fsUpdater.performUpdate()

//2) update part of foundset, for example the first 4 row (starts with selected row)
var fsUpdater = databaseManager.getFoundSetUpdater(foundset)
fsUpdater.setColumn('customer_type',new Array(1,2,3,4))
fsUpdater.setColumn('my_flag',new Array(1,0,1,0))
fsUpdater.performUpdate()

//3) safely loop through foundset (starts with selected row)
controller.setSelectedIndex(1)
var count = 0
var fsUpdater = databaseManager.getFoundSetUpdater(foundset)
while(fsUpdater.next())
{
        fsUpdater.setColumn('my_flag',count++)
}
```

resetIterator()

Start over with this iterator 'next' function (at the foundset selected record).

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
controller.setSelectedIndex(1)
var count = 0
var fsUpdater = databaseManager.getFoundSetUpdater(foundset)
while(fsUpdater.next())
{
        fsUpdater.setColumn('my_flag',++count)
}
fsUpdater.resetIterator()
while(fsUpdater.next())
{
        fsUpdater.setColumn('max_flag',count)
}
```

setColumn(name, value)

Set the column value to update, returns true if successful.

Parameters

String name The name of the column to update.
Object value The new value (can be an array with data for x number of rows) to be stored in the specified column.

Returns

Boolean true if succeeded, false if failed.

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
//1) update entire foundset
var fsUpdater = databaseManager.getFoundSetUpdater(foundset)
fsUpdater.setColumn('customer_type',1)
fsUpdater.setColumn('my_flag',0)
fsUpdater.performUpdate()

//2) update part of foundset, for example the first 4 row (starts with selected row)
var fsUpdater = databaseManager.getFoundSetUpdater(foundset)
fsUpdater.setColumn('customer_type',new Array(1,2,3,4))
fsUpdater.setColumn('my_flag',new Array(1,0,1,0))
fsUpdater.performUpdate()

//3) safely loop through foundset (starts with selected row)
controller.setSelectedIndex(1)
var count = 0
var fsUpdater = databaseManager.getFoundSetUpdater(foundset)
while(fsUpdater.next())
{
        fsUpdater.setColumn('my_flag',count++)
}
```