

JSFoundSetUpdater

Method 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.

Method Details

[next](#)
Boolean [next\(\)](#)
Go to next record in this updater, returns true if successful.

Returns
Boolean – true if proceeded to next record, false otherwise

Sample

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

[performUpdate](#)
Boolean [performUpdate\(\)](#)
Do the actual update in the database, returns true if successful.
There are 3 types of possible use with the foundset updater
1) update entire foundset, not possible when the table of the foundset has tracking enabled, will fall back to 3.
2) update part of foundset, for example the first 4 row (starts with selected row)
3) safely loop through foundset (starts with selected row)

Returns
Boolean – true if succeeded, false if failed.

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](#)
void [resetIterator\(\)](#)

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

Returns

void

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

Boolean 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 to be stored in the specified column.

Returns

Boolean – true if succeeded, false if failed.

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++)
}
```