

JSDataSource



Apr 16, 2024 18:49

Supported Clients

SmartClient WebClient NGClient

Methods Summary

QBSelect	createSelect()	Create a query builder for a data source.
QBSelect	createSelect(tableAlias)	Create a query builder for a data source with given table alias.
Array	getColumnNames()	Get the column names of a datasource.
String	getDataSource()	Get the datasource string.
JSFoundSet	getFoundSet()	Returns a foundset object for a specified datasource or server and tablename.
JSFoundSet	getFoundSet(name)	An existing foundset under that name will be returned, or created if there is a definition (there is a form with a named foundset property with that name).
JSRecord	getRecord(pk)	Get a single record from a datasource.
JSTable	getTable()	Get the table of a datasource.
JSFoundSet	loadRecords(dataset)	get a new foundset containing records based on a dataset of pks.
JSFoundSet	loadRecords(qbSelect)	get a new foundset containing records based on a QBSelect query.
JSFoundSet	loadRecords(query)	get a new foundset containing records based on an SQL query string.
JSFoundSet	loadRecords(query, args)	get a new foundset containing records based on an SQL query string with parameters.

Methods Details

createSelect()
Create a query builder for a data source.

Returns

[QBSelect](#) query builder

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var q = datasources.db.example_data.book_nodes.createSelect()  
q.result.addPk()  
q.where.add(q.columns.label_text.not.isIn(null))  
datasources.db.example_data.book_nodes.getFoundSet().loadRecords(q)
```

createSelect(tableAlias)
Create a query builder for a data source with given table alias.
The alias can be used inside custom queries to bind to the outer table.

Parameters

[String](#) tableAlias the table alias to use

Returns

[QBSelect](#) query builder

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var q = datasources.db.example_data.book_nodes.createSelect('b')  
q.result.addPk()  
q.where.add(q.columns.label_text.isIn('select comment_text from book_text t where t.note_text = ? and t.node_id  
= b.node_id', ['test']))  
datasources.db.example_data.book_nodes.getFoundSet().loadRecords(q)
```

getColumnNames()
Get the column names of a datasource.

Returns

[Array](#) [String\[\]](#) column names

Supported Clients

SmartClient,WebClient,NGClient

Sample**getDataSource()**

Get the datasource string.

Returns[String](#) String datasource**Supported Clients**

SmartClient,WebClient,NGClient

Sample

```
datasources.db.example_data.orders.getDataSource() // returns 'db:/example_data/orders'
```

getFoundSet()

Returns a foundset object for a specified datasource or server and tablename.
It is important to note that this is a FACTORY method, it constantly creates new foundsets.

Returns[JSFoundSet](#) A new JSFoundset for the datasource.**Supported Clients**

SmartClient,WebClient,NGClient

Sample

```
var fs = datasources.db.example_data.orders.getFoundSet()
var ridx = fs.newRecord()
var record = fs.getRecord(ridx)
record.emp_name = 'John'
databaseManager.saveData()
```

getFoundSet(name)

An existing foundset under that name will be returned, or created if there is a definition (there is a form with a named foundset property with that name).
If named foundset datasource does not match current datasource will not be returned (will return null instead).

Parameters[String](#) name The named foundset to get for this datasource.**Returns**[JSFoundSet](#) An existing named foundset for the datasource.**Supported Clients**

SmartClient,WebClient,NGClient

Sample

```
var fs = datasources.db.example_data.orders.getFoundSet('myname')
var ridx = fs.newRecord()
var record = fs.getRecord(ridx)
record.emp_name = 'John'
databaseManager.saveData()
```

getRecord(pk)

Get a single record from a datasource.
For the sake of performance, if more records are needed,
don't call this method in a loop but try using other methods instead.

Parameters[Object](#) pk The primary key of the record to be retrieved. Can be an array, in case of a composite pk.**Returns**[JSRecord](#) a record

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var detailsRecord = datasources.db.example_data.order_details.getRecord([10248, 11])
var orderRecord = datasources.db.example_data.orders.getRecord(10248)
var customerRecord = datasources.db.example_data.customers.getRecord('ANATR')
```

getTable()

Get the table of a datasource.

Returns

[JSTable](#) JSTable table

Supported Clients

SmartClient,WebClient,NGClient

Sample**loadRecords(dataSet)**

get a new foundset containing records based on a dataset of pks.

Parameters

[JSDataSet](#) dataSet;

Returns

[JSFoundSet](#) a new JSFoundset

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var fs = datasources.db.example_data.customers.loadRecords(pkDataSet)
```

loadRecords(qbSelect)

get a new foundset containing records based on a QBSelect query.

Parameters

[QBSelect](#) qbSelect a query builder object

Returns

[JSFoundSet](#) a new JSFoundset

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var qb = datasources.db.example_data.orders.createSelect();
qb.result.add(qb.columns.orderid);
var fs = datasources.db.example_data.orders.loadRecords(qb);
```

loadRecords(query)

get a new foundset containing records based on an SQL query string.

Parameters

[String](#) query an SQL query

Returns

[JSFoundSet](#) a new JSFoundset

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var query = "SELECT * FROM public.orders WHERE customerid = 'ROMEY' ORDER BY orderid ASC";  
var fs = datasources.db.example_data.orders.loadRecords(query);
```

loadRecords(query, args)

get a new foundset containing records based on an SQL query string with parameters.

Parameters

[String](#) query an SQL query string with parameter placeholders

[Array](#) args an array of arguments for the query string

Returns

[JSFoundSet](#) a new JSFoundset

Supported Clients

SmartClient, WebClient, NGClient

Sample

```
var query = "SELECT * FROM public.orders WHERE customerid = ? OR customerid = ? order by orderid asc";  
var args = ['ROMEY', 'BERGS'];  
var fs = datasources.db.example_data.orders.loadRecords(query, args);
```