Data Layer

Array Store

The ArrayStore is a store that provides an interface for loading and editing an in-memory array and handling related events.

Properties

- data - Specifies the store's associated array.
- errorHandler - Specifies the function that is executed when the store throws an error.
- key - Specifies the key property (or properties) that provide(s) key values to access data items. Each key value must be unique.
- onInserted: A function that is executed after a data item is added to the store.
- onInserting - A function that is executed before a data item is added to the store.
- onLoaded - A function that is executed after data is loaded to the store.
- onLoading - A function that is executed before data is loaded to the store.
- onModified - A function that is executed after a data item is added, updated, or removed from the store.
- onModifying - A function that is executed before a data item is added, updated, or removed from the store.
- onPush - The function executed before changes are pushed to the store.
- onRemoved - A function that is executed after a data item is removed from the store.
- onRemoving - A function that is executed before a data item is removed from the store.
- onUpdated - A function that is executed after a data item is updated in the store.
- onUpdating - A function that is executed before a data item is updated in the store.

Methods

- byKey(key) - Gets a data item with a specific key.
- clear() - Clears all the ArrayStore's associated data.
- createQuery - Creates a Query for the underlying array.
- insert(values) - Adds a data item to the store.
- key() - Gets the key property (or properties) as specified in the key property.
- keyOf(obj) - Gets a data item's key value.
- load() - Starts loading data.
- load(options) - Starts loading data.
- push(changes) - Pushes data changes to the store and notifies the DataSource.
- remove(key) - Removes a data item with a specific key from the store.
- totalCount(options) - Gets the total count of items the load() function returns.
- update(key, values) - Updates a data item with a specific key.

Custom Store

The CustomStore enables you to implement custom data access logic for consuming data from any source.

Properties

- byKey
- cacheRawData - Specifies whether raw data should be saved in the cache. Applies only if loadMode is "raw".
- errorHandler
- insert - Specifies a custom implementation of the insert(values) method.
- key
- load - Specifies a custom implementation of the load(options) method.
- loadMode - Specifies how data returned by the load function is treated. ‘processed’ | ‘raw’
- remove - Specifies a custom implementation of the remove(key) method.
- totalCount - Specifies a custom implementation of the totalCount(options) method.
- update - Specifies a custom implementation of the update(key, values) method.
- useDefaultSearch - Specifies whether the store combines the search and filter expressions. Defaults to true if the loadMode is "raw" and false if it is "processed".

Methods

- byKey(key)
- clearRawDataCache() - Deletes data from the cache. Takes effect only if the cacheRawData property is true.
- insert(values)
- key()
- keyOf(obj)
- load(), load(options)
- remove(key)
- totalCount(options)
- update(key, values)

Load Options

This object is used to specify settings according to which the server should process data. More often these settings are passed as a parameter to the load function and depend on the operations (paging, filtering, sorting, etc.) that you have enabled in the DataSource or UI component.

- filter - A filter expression.
- group - A group expression. (selector, desc, isExpanded, groupInterval)
- groupSummary - A group summary expression. Used with the group setting.{ selector: "field", summaryType: "sum" }
- parentIds - The IDs of the rows being expanded. Relevant only when the CustomStore is used in the TreeList UI component.

- requireGroupCount - Indicates whether a top-level group count is required. Used in conjunction with the filter, take, skip, requireTotalCount, and group settings.
- requireTotalCount - Indicates whether the total count of data objects is needed.
- searchExpr - A data field or expression whose value is compared to the search value.
- searchOperation - A comparison operation. Can have one of the following values: "=", "<>", ">", ">=", "<", "<=", "startswith", "endswith", "contains", "notcontains", "isblank" and "isnotblank".
- searchValue - The current search value.
- select - A select expression.
- skip - The number of data objects to be skipped from the result set's start. In conjunction with take, used to implement paging.
- sort - A sort expression.
- take - The number of data objects to be loaded. In conjunction with skip, used to implement paging.
- totalSummary - A total summary expression.
- userData - An object for storing additional settings that should be sent to the server.

Data Source

The DataSource is an object that provides an API for processing data from an underlying store.

Options

- filter - Specifies data filtering conditions.
- group - Specifies data grouping properties.
- map - Specifies an item mapping function.
- onChanged - A function that is executed after data is loaded.
- onLoadError - A function that is executed when data loading fails.
- onLoadingChanged - A function that is executed when the data loading status changes.
- pageSize - Specifies the maximum number of data items per page. Applies only if paginate is true.
- paginate - Specifies whether the DataSource loads data items by pages or all at once. Defaults to false if group is set; otherwise, true.
- postProcess -Specifies a post processing function.
- pushAggregationTimeout - Specifies the period (in milliseconds) when changes are aggregated before pushing them to the DataSource.
- requireTotalCount - Specifies whether the DataSource requests the total count of data items in the storage.
- reshapeOnPush - Specifies whether to reapply sorting, filtering, grouping, and other data processing operations after receiving a push.
- searchExpr - Specifies the fields to search.
- searchOperation - Specifies the comparison operation used in searching. The following values are accepted: "=", "<>", ">", ">=", "<", "<=", "startswith", "endswith", "contains", "notcontains".
- searchValue - Specifies the value to which the search expression is compared.
- select - Specifies the fields to select from data objects.

- sort - Specifies data sorting properties.
- store - Configures the store underlying the DataSource.
    - type: ‘array’, ‘local’, ‘odata’

Methods

- cancel(operationId) - Cancels the load operation with a specific identifier.
- filter() - Gets the filter property's value.
- filter(filterExpr) - Sets the filter property's value.
- group() - Gets the group property's value.
- group(groupExpr) - Sets the group property's value.
- isLastPage() - Checks whether the count of items on the current page is less than the pageSize. Takes effect only with enabled paging.
- isLoaded() - Checks whether data is loaded in the DataSource.
- isLoading() - Checks whether data is being loaded in the DataSource.
- items() - Gets an array of data items on the current page.
- key() - Gets the value of the underlying store's key property.
- load() - Starts loading data.
- loadOptions() - Gets an object with current data processing settings.
- pageIndex() - Gets the current page index.
- pageIndex(newIndex) - Sets the index of the page that should be loaded on the next load() method call. -
- pageSize() - Gets the page size.
- pageSize(value)Sets the page size.
- paginate() - Gets the paginate property's value.
- paginate(value) - Sets the paginate property's value.
- reload() - Clears currently loaded DataSource items and calls the load() method.
- requireTotalCount() - Gets the requireTotalCount property's value.
- requireTotalCount(value) - Sets the requireTotalCount property's value.
- searchExpr() - Gets the searchExpr property's value.
- searchExpr(expr) - Sets the searchExpr property's value.
- searchOperation() - Gets the searchOperation property's value.
- searchOperation(op) - Sets the searchOperation property's value.
- searchValue() - Gets the searchValue property's value.
- searchValue(value) - Sets the searchValue property's value.
- select() - Gets the select property's value.
- select(expr) - Sets the select property's value.
- sort() -Gets the sort property's value.
- sort(sortExpr) - Sets the sort property's value.
- store() - Gets the instance of the store underlying the DataSource.
- totalCount() - Gets the number of data items in the store after the last load() operation without paging. Takes effect only if requireTotalCount is true.

Local Store

Properties

The LocalStore is a store that provides an interface for loading and editing data from HTML Web Storage (also known as window.localStorage) and handling related events.

- data - Specifies the store's associated array.
- errorHandler - Specifies the function that is executed when the store throws an error.
- flushInterval - Specifies a delay in milliseconds between when data changes and the moment these changes are saved in the local storage.
- immediate - Specifies whether the LocalStore saves changes in the local storage immediately.
- key - Specifies the key property (or properties) that provide(s) key values to access data items. Each key value must be unique.
- name - Specifies the name under which data should be saved in the local storage. The dx-data-localStore- prefix will be added to the name.
- onInserted
- onInserting
- onLoaded
- onLoading
- omModifying
- onModified
- onPush
- onRemoved
- onRemoving
- onUpdated
- onUpdating

Methods

- byKey(key)
- clear() - Removes data from the local storage.
- createQuery() - Creates a Query for the underlying array.
- insert(values)
- key()
- keyOf(obj)
- load()
- load(options)
- push(changes)
- remove(key)
- totalCount(options)
- update(key, values)

Query

The Query is an object that provides a chainable interface for making data queries.
To create a Query, call the query(array) or query(url, queryOptions) method, depending on the type of the storage you access. The Query supports method chaining. This enables you to execute several methods in a single statement.

```
var dataObjects = [
   { name: "Amelia", birthYear: 1991, gender: "female" },
   { name: "Benjamin", birthYear: 1983, gender: "male" },
   { name: "Daniela", birthYear: 1987, gender: "female" },
   { name: "Lee", birthYear: 1981, gender: "male" }
```

```
];

var processedArray = DevExpress.data.query(dataObjects)
    .filter([ "gender", "=", "female" ])
    .sortBy("birthYear")
    .select("name", "birthYear")
    .toArray();
```

Methods

- aggregate(seed, step, finalize) - Calculates a custom summary for all data items.
- aggregate(step) - Calculates a custom summary for all data items.
- avg() - Calculates the average of all values. Applies only to numeric arrays.
- avg(getter) - Calculates the average of all values found using a getter.
- count() - Calculates the number of data items.
- enumerate() - Executes the Query. This is an asynchronous alternative to the toArray() method.
- filter(criteria) - Filters data items using a filter expression.
- filter(predicate) - Filters data items using a custom function.
- groupBy(getter) - Groups data items by the specified getter.
- max() - Calculates the maximum value. Applies only to numeric arrays.
- max(getter) - Calculates the maximum of all values found using a getter.
- min() - Calculates the minimum value. Applies only to numeric arrays.
- min(getter) - Calculates the minimum of all values found using a getter.
- select(getter) - Selects individual fields from data objects.
- slice(skip, take) - Gets a specified number of data items starting from a given index.
- sortBy(getter) - Sorts data items by the specified getter in ascending order.
- sortBy(getter, desc) - Sorts data items by the specified getter in the specified sorting order.
- sum() - Calculates the sum of all values.
- sum(getter) - Calculates the sum of all values found using a getter.
- thenBy(getter) - Sorts data items by one more getter in ascending order.
- thenBy(getter, desc) - Sorts data items by one more getter in the specified sorting order.
- toArray() - Gets data items associated with the Query. This is a synchronous alternative to the enumerate() method.