

## Join GitHub today

Dismiss

GitHub is home to over 20 million developers working together to host and review code, manage projects, and build software together.

Sign up

### Material Design Data Table for Angular Material

 296 commits

 4 branches

 43 releases

 13 contributors

 MIT

Branch: master ▼

New pull request








Find file





Clone or download ▼



**daniel-nagy** committed on Jun 27, 2017 Update README.md ...

Latest commit d8627e1 on Jun 27, 2017

 <a href="#">app</a>	map language to limit options	2 years ago
 <a href="#">dist</a>	Convert mdTableProgress class directive into an attribute directive.	a year ago
 <a href="#">src</a>	Convert mdTableProgress class directive into an attribute directive.	a year ago
 <a href="#">.gitignore</a>	adding dist	3 years ago
 <a href="#">.jshintrc</a>	early implementation of md-disable-select	3 years ago
 <a href="#">CHANGELOG.md</a>	update docs	a year ago
 <a href="#">Gruntfile.js</a>	fixing select menu styles for angular material v1.0.4	2 years ago

 <a href="#">LICENSE.md</a>	Create LICENSE.md	3 years ago
 <a href="#">README.md</a>	Update README.md	9 months ago
 <a href="#">bower.json</a>	bump version	a year ago
 <a href="#">index.js</a>	fixing select menu styles for angular material v1.0.4	2 years ago
 <a href="#">package.json</a>	bump version	a year ago

## README.md

# Material Design Data Table

---

This module is an effort to implement Material Design data tables in [Angular Material](#). Data tables are used to present raw data sets and usually appear in desktop enterprise applications. Data tables are particularly useful for visualizing and manipulating large data sets.

Specification for Material Design data tables can be found [here](#).

- [License](#)
- [Demo](#)
- [Installation](#)
- [Usage](#)
- [Change Log](#)
- [API Documentation](#)
- [Contributing](#)

## License

---

This software is provided free of charge and without restriction under the [MIT License](#)

## Demo

---

A live [demo](#).

A fork-able [Codepen](#). Please use this to reproduce any issues you may be experiencing.

## Installation

---

### Using Bower

This package is installable through the Bower package manager.

```
bower install angular-material-data-table --save
```

In your `index.html` file, include the data table module and style sheet.

```
<!-- style sheet -->
<link href="bower_components/angular-material-data-table/dist/md-data-table.min.css" rel="stylesheet" type="text/css"
<!-- module -->
<script type="text/javascript" src="bower_components/angular-material-data-table/dist/md-data-table.min.js"></script>
```



Include the `md.data.table` module as a dependency in your application.

```
angular.module('myApp', ['ngMaterial', 'md.data.table']);
```

### Using npm and Browserify (or JSPM)

In addition, this package may be installed using npm.

```
npm install angular-material-data-table --save
```

You may use Browserify to inject this module into your application.

```
angular.module('myApp', [require('angular-material-data-table')]);
```

## Usage

---

### Example Controller

```
// Assume we have a $nutrition service that provides an API for communicating with the server

angular.module('demoApp').controller('sampleController', ['$nutrition', '$scope', function ($nutrition, $scope) {
  'use strict';

  $scope.selected = [];

  $scope.query = {
    order: 'name',
    limit: 5,
    page: 1
  };

  function success(desserts) {
    $scope.desserts = desserts;
  }

  $scope.getDesserts = function () {
    $scope.promise = $nutrition.desserts.get($scope.query, success).$promise;
  };
}]);
```

```
});
```

## Example Template

```
<md-toolbar class="md-table-toolbar md-default">
  <div class="md-toolbar-tools">
    <span>Nutrition</span>
  </div>
</md-toolbar>

<!-- exact table from live demo -->
<md-table-container>
  <table md-table md-row-select multiple ng-model="selected" md-progress="promise">
    <thead md-head md-order="query.order" md-on-reorder="getDesserts">
      <tr md-row>
        <th md-column md-order-by="nameToLower"><span>Dessert (100g serving)</span></th>
        <th md-column md-numeric md-order-by="calories.value"><span>Calories</span></th>
        <th md-column md-numeric>Fat (g)</th>
        <th md-column md-numeric>Carbs (g)</th>
        <th md-column md-numeric>Protein (g)</th>
        <th md-column md-numeric>Sodium (mg)</th>
        <th md-column md-numeric>Calcium (%)</th>
        <th md-column md-numeric>Iron (%)</th>
      </tr>
    </thead>
    <tbody md-body>
      <tr md-row md-select="dessert" md-select-id="name" md-auto-select ng-repeat="dessert in desserts.data">
        <td md-cell>{{dessert.name}}</td>
        <td md-cell>{{dessert.calories.value}}</td>
        <td md-cell>{{dessert.fat.value | number: 1}}</td>
        <td md-cell>{{dessert.carbs.value}}</td>
        <td md-cell>{{dessert.protein.value | number: 1}}</td>
        <td md-cell>{{dessert.sodium.value}}</td>
        <td md-cell>{{dessert.calcium.value}}{{dessert.calcium.unit}}</td>
        <td md-cell>{{dessert.iron.value}}{{dessert.iron.unit}}</td>
      </tr>
    </tbody>
  </table>
</md-table-container>
```

```
        </tr>
      </tbody>
    </table>
  </md-table-container>
```

```
<md-table-pagination md-limit="query.limit" md-limit-options="[5, 10, 15]" md-page="query.page" md-total="{{desserts.
```

## API Documentation

---

### v0.10.x

- [Column Sorting](#)
- [Edit Dialogs](#)
- [Inline Menus](#)
- [Numeric Columns](#)
- [Pagination](#)
- [Row Selection](#)
- [Table Progress](#)
- [Table Toolbars](#)

### Earlier Versions

- [0.9.x](#)
- [<=0.8.14](#)

Tables are sorted alphabetically by their first column. I will be **camelCasing** attributes in tables (otherwise the cells would wrap and be difficult to read) but don't forget to **kebab-case** them in your template.

## Column Sorting

Attribute	Element	Type	Description
mdDesc	mdColumn	[expression]	If present, the column will sort descending first. The default is to sort ascending first.
mdOnReorder	mdHead	function	A callback function for when the order changes. The callback will receive the new order.
mdOrder	mdHead	string	A variable to bind the sort order to.
mdOrderBy	mdColumn	string	The value to bind to the sort order.

When the user clicks the `md-column` element, the value of the `md-order-by` attribute will be bound to the variable provided to the `md-order` attribute on the `md-head` element. If the column is already sorted by that value, a minus sign `-` will be prefixed to the value. For most query languages, this is the universal symbol to sort descending.

The variable can be used to send a query to the server or as the `orderBy` property of an `ng-repeat` expression.

### Example Using ngRepeat

```
<md-table-container>
  <table md-table>
    <thead md-head md-order="myOrder">
      <!-- when the user clicks this cell, the myOrder variable will get the value 'nameToLower' -->
      <th md-column md-order-by="nameToLower">Dessert (100g serving)</th>
      <!-- the variable myOrder will not be changed when this cell is clicked -->
      <th md-column md-numeric>Calories</th>
    </thead>
    <tbody md-body>
      <!-- we can let ng-repeat sort the columns for us -->
      <tr ng-repeat="dessert in desserts | orderBy: myOrder"></tr>
    </tbody>
  </table>
</md-table-container>
```

## Edit Dialogs

Tables may require basic text editing. This module includes a service for displaying edit dialogs to modify text or anything else really. The service provides presets for both small and large edit dialogs designed for manipulating text. It also has full support for creating custom dialogs so you can be as creative as you want to be.

Unlike Angular Material dialogs, the preset methods will open the dialog.

### Restrictions

- The dialog will always receive a new isolated scope.
- You must provide a `targetEvent` and the event target must be a table cell.

### Example

```
$scope.editComment = function (event, dessert) {  
  // if auto selection is enabled you will want to stop the event  
  // from propagating and selecting the row  
  event.stopPropagation();  
  
  /*  
   * messages is commented out because there is a bug currently  
   * with ngRepeat and ngMessages were the messages are always  
   * displayed even if the error property on the ngModelController  
   * is not set, I've included it anyway so you get the idea  
   */  
  
  var promise = $mdEditDialog.small({  
    // messages: {  
    //   test: 'I don\'t like tests!'  
    // },  
    modelValue: dessert.comment,  
    placeholder: 'Add a comment',  
    save: function (input) {  
      dessert.comment = input.$modelValue;
```



```

    },
    targetEvent: event,
    validators: {
      'md-maxlength': 30
    }
  });

  promise.then(function (ctrl) {
    var input = ctrl.getInput();

    input.$viewChangeListeners.push(function () {
      input.$setValidity('test', input.$modelValue !== 'test');
    });
  });
});

```

## Small Edit Dialogs

```
$mdEditDialog.small(options);
```

Parameter	Type	Description
options	object	Small preset options.

Property	Type	Default	Description
clickOutsideToClose	bool	true	The user can dismiss the dialog by clicking anywhere else on the page.
disableScroll	bool	true	Prevent user scroll while the dialog is open.
escToClose	bool	true	The user can dismiss the dialog by clicking the esc key.
focusOnOpen	bool	true	Will search the template for an <code>md-autofocus</code> element.

Property	Type	Default	Description
messages	object	null	Error messages to display corresponding to errors on the <code>ngModelController</code> .
modelValue	string	null	The initial value of the input element.
placeholder	string	null	Placeholder text for input element.
save	function	null	A callback function for when the use clicks the save button. The callback will receive the <a href="#">ngModelController</a> . The dialog will close unless the callback returns a rejected promise.
targetEvent	event	null	The event object. This must be provided and it must be from a table cell.
type	string	"text"	The value of the <code>type</code> attribute on the <code>input</code> element.
validators	object	null	Optional attributes to be placed on the input element.

The `small` method will return a `promise` that will resolve with the controller instance. The controller has two public methods, `dismiss` which will close the dialog without saving and `getInput` which will return the `ngModelController` .

## Large Edit Dialogs

Large edit dialogs are functionally identical to small edit dialogs but have a few additional options.

```
$mdEditDialog.large(options);
```

Parameter	Type	Description
options	object	Large preset options.

Property	Type	Default	Description
cancel	string	"Cancel"	Text to dismiss the dialog without saving.
clickOutsideToClose	bool	true	The user can dismiss the dialog by clicking anywhere else on the page.
disableScroll	bool	true	Prevent user scroll while the dialog is open.
escToClose	bool	true	The user can dismiss the dialog by clicking the esc key.
focusOnOpen	bool	true	Will search the template for an <code>md-autofocus</code> element.
messages	object	null	Error messages to display corresponding to errors on the <code>ngModelController</code> .
modelValue	string	null	The initial value of the input element.
ok	string	"Save"	Text to submit and dismiss the dialog.
placeholder	string	null	Placeholder text for input element.
save	function	null	A callback function for when the use clicks the save button. The callback will receive the <code>ngModelController</code> . The dialog will close unless the callback returns a rejected promise.
targetEvent	event	null	The event object. This must be provided and it must be from a table cell.
title	string	"Edit"	Dialog title text.
type	string	"text"	The value of the <code>type</code> attribute on the <code>input</code> element.
validators	object	null	Optional attributes to be placed on the input element.

The `large` method will return a `promise` that will resolve with the controller instance. The controller has two public methods, `dismiss` which will close the dialog without saving and `getInput` which will return the `ngModelController`.

## Roll Your Own

```
$mdEditDialog.show(options);
```

Parameter	Type	Description
options	object	Dialog options.

Property	Type	Default	Description
<code>bindToController</code>	<code>bool</code>	<code>false</code>	If true, properties on the provided scope object will be bound to the controller
<code>clickOutsideToClose</code>	<code>bool</code>	<code>true</code>	The user can dismiss the dialog by clicking anywhere else on the page.
<code>controller</code>	<code>function</code> <code>string</code>	<code>null</code>	Either a constructor function or a string register with the <code>\$controller</code> service. The controller will be automatically injected with <code>\$scope</code> and <code>\$element</code> . Remember to annotate your controller if you will be minifying your code.
<code>controllerAs</code>	<code>string</code>	<code>null</code>	An alias to publish your controller on the scope.
<code>disableScroll</code>	<code>bool</code>	<code>true</code>	Prevent user scroll while the dialog is open.
<code>escToClose</code>	<code>bool</code>	<code>true</code>	The user can dismiss the dialog by clicking the esc key.
<code>focusOnOpen</code>	<code>bool</code>	<code>true</code>	Will search the template for an <code>md-autofocus</code> element.

Property	Type	Default	Description
<code>locals</code>	<code>object</code>	<code>null</code>	Optional dependancies to be injected into your controller. It is not necessary to inject registered services, the <code>\$injector</code> will provide them for you.
<code>resolve</code>	<code>object</code>	<code>null</code>	Similar to <code>locals</code> but waits for promises to be resolved. Once the promises resolve, their return value will be injected into the controller and the dialog will open.
<code>scope</code>	<code>object</code>	<code>null</code>	Properties to bind to the new isolated scope.
<code>targetEvent</code>	<code>event</code>	<code>null</code>	The event object. This must be provided and it must be from a table cell.
<code>template</code>	<code>string</code>	<code>null</code>	The template for your dialog.
<code>templateUrl</code>	<code>string</code>	<code>null</code>	A URL to fetch your template from.

The `show` method will return a `promise` that will resolve with the controller instance.

Table cells have a `md-placeholder` CSS class that you can use for placeholder text.

### Example: A Table Cell That Opens An Edit Dialog

```
<td md-cell ng-click="editComment($event, dessert)" ng-class="{ 'md-placeholder': !dessert.comment }">
  {{dessert.comment || 'Add a comment'}}
</td>
```

## Inline Menus

Table cells support inline menus. To use an inline menu, place an `md-select` element inside an `md-cell` element.

## Example

```
<td md-cell>
  <md-select ng-model="dessert.type" placeholder="Other">
    <md-option ng-value="type" ng-repeat="type in getTypes()">{{type}}</md-option>
  </md-select>
</td>
```

Clicking anywhere in the cell will activate the menu. In addition, if you have automatic row selection enabled the row will not be selected when the cell is clicked.

## Numeric Columns

Numeric columns align to the right of table cells.

Attribute	Element	Type	Description
mdNumeric	mdColumn	[expression]	If the expression is <code>null</code> or evaluates to <code>true</code> then all the cells in that column will be right aligned

You may use Angular's [number](#) filter on a cell to set the decimal precision.

```
<!-- 2 decimal places -->
<td md-cell>{{dessert.protein.value | number: 2}}</td>
```

If you are using `colspan` you may need to manually correct the alignment and padding of cells. You can override the cell's style with a custom CSS class.

## Pagination

Attribute	Type	Description
mdBoundaryLinks	[expression]	Displays first and last page navigation links
mdLabel	object	Change the pagination label (see more below).
mdLimit	integer	A row limit.
mdLimitOptions	array	Row limit options (see more below).
mdOnPaginate	function	A callback function for when the page or limit changes. The page is passed as the first argument and the limit is passed as the second argument.
mdPage	integer	Page number. Pages are not zero indexed. The directive assumes the first page is one.
mdPageSelect	[expression]	Display a select dropdown for the page number
mdTotal	integer	Total number of items.
ngDisabled	[expression]	Disable pagination elements.

The `md-label` attribute has the following properties.

Property	Type	Default
of	string	'of' (e.g. x - y of z)
page	string	'Page:'
rowsPerPage	string	'Rows per page:'

The `md-limit-options` attribute supports integers or objects with the properties `label` and `value`. The latter is convenient for when you want to use language to give meaning to individual options, e.g.

```
// the 'All' option will show all items in the collection
ctrl.limitOptions = [5, 10, 15, {
  label: 'All',
  value: function () {
    return collection.length;
  }
}];
```

### Example: Changing The Pagination Label

```
<!-- how to change the pagination label -->
<md-table-pagination md-label="{page: 'Página:', rowsPerPage: 'Filas por página:', of: 'de'}"></md-table-pagination>

<!-- or if the label is defined on the scope -->
<md-table-pagination md-label="{{label}}"></md-table-pagination>
```



I used Google translate so if the translations are wrong please fix them and make a pull request.

### Example: Client Side Pagination Using ngRepeat

```
<tr md-row ng-repeat="item in array | orderBy: myOrder | limitTo: myLimit: (myPage - 1) * myLimit">

<!-- and your pagination element will look something like... -->
<md-table-pagination md-limit="myLimit" md-page="myPage" md-total="{{array.length}}"></md-table-pagination>
```

### My Pagination Isn't Working?!

- Make sure you pass `md-page`, `md-limit`, and `md-total` to the directive and that they are finite numbers.
- Pages are not zero indexed. The directive will assume pages start at one. If your query language expects pages to be zero indexed then just subtract one before making the query.



## Row Selection

Attribute	Element	Type	Description
mdAutoSelect	mdRow	[expression]	Select a row by clicking anywhere in the row.
mdOnDeselect	mdRow	function	A callback function for when an item is deselected. The item will be passed as an argument to the callback.
mdOnSelect	mdRow	function	A callback function for when an item is selected. The item will be passed as an argument to the callback.
mdRowSelect	mdTable	[expression]	Enable row selection.
mdSelect	mdRow	any	The item to be selected.
mdSelectId	mdRow	number string	A unique identifier for the selected item. The identifier must be a property of the item.
multiple	mdTable	[expression]	Allow multiple selection. When enabled a master toggle will be prepended to the last row of table header.
ngDisabled	mdRow	expression	Conditionally disable row selection.
ngModel	mdTable	array	A variable to bind selected items to.

By default selected items will persist. Equality between items is determined using the `===` operator. In cases where items may not be strictly equal, you must provide a unique identifier for the item.

You may manually add or remove items from the model but be aware that select and deselect callbacks will not be triggered. When operating in single select mode, the deselect callback will not be triggered when a user selects another item. It will be trigger, however, if the user directly deselects the item. In multi-select mode, the master toggle will trigger select and deselect callbacks for all items selected or deselected respectfully.

### Example: Row Selection From The Live Demo.

```
<tr md-row md-select="dessert" md-select-id="name" md-auto-select ng-repeat="dessert in desserts.data">
```

### Example: Clearing Selected Items On Pagination

```
$scope.onPaginate = function () {  
  $scope.selected = [];  
}
```

## Table Progress

Attribute	Target	Type	Description
mdProgress	mdTable	promise promise<Array>	The table will display a loading indicator until all promises are resolved or rejected.

The table module can display a loading indicator for you whenever asynchronous code is executing. It accepts a promise or an array of promises. If another promise is received before the previous promise is resolved or rejected it will be placed in a queue.

Because I spent almost an hour debugging this I thought I would share with you. I'm not sure why this is the case but if you put the deferred object on the scope and try to pass the promise to the directive from that, the queue mechanism will not work properly.

### This Will Not Work

```
function () {  
  $scope.deferred = $q.defer();  
  // ...
```

```
$scope.deferred.resolve();  
}
```

```
<table md-table md-progress="deferred.promise"></table>
```

## This Will Work

```
function () {  
  var deferred = $q.defer();  
  $scope.promise = deferred.promise;  
  // ...  
  deferred.resolve();  
}
```

```
<table md-table md-progress="promise"></table>
```

In addition, if you are dealing with something that returns a promise directly (not a deferred object) you don't need to worry about it.

```
function () {  
  $scope.promise = $timeout(function () {  
    // ...  
  }, 2000);  
}
```

## Table Toolbars

Tables may be embedded within cards that offer navigation and data manipulation tools available at the top and bottom. You can use the `md-table-toolbar` and `md-default` class on a `md-toolbar` element for a plain white toolbar.

If you need to display information relative to a particular column in the table you may use use a `<md-foot>` element. For example, say you had a `calories.total` property that summed the total number of calories and you wanted to display that information directly beneath the Calories column.

```
<tfoot md-foot>
  <tr md-row>
    <td md-cell></td>
    <td md-cell><strong>Total: </strong>{{calories.total}}</td>
    <td md-cell colspan="6"></td>
  </tr>
</tfoot>
```

Observe that Calories is the second column in the table. Therefore, we need to offset the first column with an empty cell. If you need to offset many columns you can use `<td colspan="{n}"></td>` where `n` is the number of columns to offset.

You may need to manually correct the the text alignment and cell padding if you use `colspan` .

## Contributing

---

### Requires

- node
- grunt-cli

This repository contains a demo application for developing features. As you make changes the application will live reload itself.

### Update

I noticed the nutrition app was an inconvenience for people trying to run the app locally and contribute. I have updated the demo application to remove the dependency for the nutrition app. This is also a good example of how you can take advantage of `ngRepeat` to easily achieve client side sorting and pagination.

### Running the App Locally

Clone this repository to your local machine.

```
git clone https://github.com/daniel-nagy/md-data-table.git
cd md-data-table
```

Create a new branch for the issue you are working on.

```
git checkout -b my-issue
```

Install the package dependencies.

```
npm install
bower install
```

Run the application and visit `127.0.0.1:8000` in the browser.

```
grunt
```

Make your modifications and update the build.

```
grunt build
```

Create a pull request!