Angular Material 8|9 Mat-table Multiple Column Filters Using Select Boxes

Last updated on April 9, 2020 Jolly.exe

> Angular Material Table provides a great default filter component to filter data shown to the user, but this takes into account all columns and their cells. In this tutorial, we will create custom selection filters for the table's values with multiple selections.

Why do this?

Sometimes we have data in the datatable with similar data values for example **STATUS** column may have **Active**, **InActive**, **Blocked** values. So we can create a filter for Status to show only Active rows.

Here we will also create a function to fetch Unique values from Table rows itself and generate Filte drop down to the filter.

This is how it will work

Check the working demo here.

GitHub repository link

The filters above table are created by using data objects in the table itself, but it will only show unique values and remove the duplicates. We will discuss this special function to create these filters out of table data.

Let's get started and implement it by creating a new Angular application and installing Angular Material in it. Here we are using **Angular version 9.0.6**

Create An Angular Project

To quickly create an Angular project, we use the `Ng CLI` tool which makes it very easy to create an Angular project with all boilerplates required. Install it by running

```
$ npm install -g @angular/cli
```

Run the following command to create a project:

```
$ ng new angular-material-table-filters
$ cd my-first-project
$ code .
```

Install Material Package

After creating the project and moving to its root, run following NPM command and answer some configuration questions to install the Material package in your project.

```
$ ng add @angular/material
? Choose a prebuilt theme name, or "custom" for a custom theme: I
? Set up global Angular Material typography styles? Yes
? Set up browser animations for Angular Material? Yes
```

Update App Module

In our tutorial, we will add Material Table and other elements like SelectBox for filters, Button to reset. Also to fetch server JSON data we will make an HTTP call using `HttpClientModule`.

For using all these modules, we will update the **app.module.ts** file as shown below:

```
// app.module.ts
```

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { BrowserAnimationsModule } from '@angular/platform-browse

import { FormsModule, ReactiveFormsModule } from '@angular/forms'
import { MatTableModule } from '@angular/material/table';
import { MatSelectModule } from '@angular/material/select';
import { MatInputModule } from '@angular/material/input';
import { MatButtonModule } from '@angular/material/button';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    BrowserAnimationsModule,

    FormsModule,
    ReactiveFormsModule,

    MatButtonModule,
    MatInputModule,
    MatTableModule,
    MatSelectModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Adding Material Table

To keep this tutorial simple, we will add the material table in the App component which is created by default when we create a project using the Ng CLI tool.

For creating a Material table update the **app.component.html** template with below code:

```
<table mat-table [dataSource]="dataSource">

  <ng-container matColumnDef="id">
    <th mat-header-cell *matHeaderCellDef> id </th>
    <td mat-cell *matCellDef="let element"> {{element.id}} </td
  </ng-container>

  <ng-container matColumnDef="name">
    <th mat-header-cell *matHeaderCellDef> name </th>
    <td mat-cell *matCellDef="let element"> {{element.name}} </
  </ng-container>

  <ng-container matColumnDef="username">
    <th mat-header-cell *matHeaderCellDef> username </th>
    <td mat-cell *matCellDef="let element"> {{element.username}
  </ng-container>

  <ng-container matColumnDef="email">
    <th mat-header-cell *matHeaderCellDef> email </th>
    <td mat-cell *matCellDef="let element"> {{element.email}} <
  </ng-container>

  <ng-container matColumnDef="phone">
    <th mat-header-cell *matHeaderCellDef> phone </th>
    <td mat-cell *matCellDef="let element"> {{element.phone}} <
  </ng-container>

  <ng-container matColumnDef="website">
    <th mat-header-cell *matHeaderCellDef> website </th>
    <td mat-cell *matCellDef="let element"> {{element.website}}
  </ng-container>

  <ng-container matColumnDef="status">
    <th mat-header-cell *matHeaderCellDef> status </th>
    <td mat-cell *matCellDef="let element"> {{element.status}}
  </ng-container>

  <tr mat-header-row *matHeaderRowDef="displayedColumns;"></tr>
  <tr mat-row *matRowDef="let row; columns: displayedColumns;">

</table>
```

To feed this table with remote data, update the **app.component.ts** file with the following code:

```
// app.component.ts
import { Component } from '@angular/core';
import { MatTableDataSource } from '@angular/material/table';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.scss']
})
export class AppComponent {

  dataSource = new MatTableDataSource();
  displayedColumns: string[] = ['id', 'name', 'username', 'email'

  constructor(
  ) {
  }

  ngOnInit() {
    this.getRemoteData();
  }

  // Get remote serve data using HTTP call
  getRemoteData() {

    const remoteDummyData = [
      {
        "id": 1,
        "name": "Leanne Graham",
        "username": "Bret",
        "email": "Sincere@april.biz",
        "phone": "1-770-736-8031 x56442",
        "website": "hildegard.org",
        "status": "Active"
      },
      {
        "id": 2,
        "name": "Ervin Howell",
        "username": "Antonette",
        "email": "Shanna@melissa.tv",
        "phone": "010-692-6593 x09125",
        "website": "anastasia.net",
        "status": "Blocked"
```

```
  },
  {
    "id": 3,
    "name": "Clementine Bauch",
    "username": "Samantha",
    "email": "Nathan@yesenia.net",
    "phone": "1-463-123-4447",
    "website": "ramiro.info",
    "status": "Blocked"
  },
  {
    "id": 4,
    "name": "Patricia Lebsack",
    "username": "Karianne",
    "email": "Julianne.OConner@kory.org",
    "phone": "493-170-9623 x156",
    "website": "kale.biz",
    "status": "Active"
  },
  {
    "id": 5,
    "name": "Chelsey Dietrich",
    "username": "Kamren",
    "email": "Lucio_Hettinger@annie.ca",
    "phone": "(254)954-1289",
    "website": "demarco.info",
    "status": "Active"
  },
  {
    "id": 6,
    "name": "Mrs. Dennis Schulist",
    "username": "Leopoldo_Corkery",
    "email": "Karley_Dach@jasper.info",
    "phone": "1-477-935-8478 x6430",
    "website": "ola.org",
    "status": "In-Active"
  },
  {
    "id": 7,
    "name": "Kurtis Weissnat",
    "username": "Elwyn.Skiles",
    "email": "Telly.Hoeger@billy.biz",
    "phone": "210.067.6132",
    "website": "elvis.io",
    "status": "Active"
  },
  {
    "id": 8,
    "name": "Nicholas Runolfsdottir V",
```

```
        "username": "Maxime_Nienow",
        "email": "Sherwood@rosamond.me",
        "phone": "586.493.6943 x140",
        "website": "jacynthe.com",
        "status": "In-Active"
      },
      {
        "id": 9,
        "name": "Glenna Reichert",
        "username": "Delphine",
        "email": "Chaim_McDermott@dana.io",
        "phone": "(775)976-6794 x41206",
        "website": "conrad.com",
        "status": "In-Active"
      },
      {
        "id": 10,
        "name": "Clementina DuBuque",
        "username": "Moriah.Stanton",
        "email": "Rey.Padberg@karina.biz",
        "phone": "024-648-3804",
        "website": "ambrose.net",
        "status": "Active"
      }
    ];
    this.dataSource.data = remoteDummyData;
  }

}
```

Above we have `getRemoteData` function to fetch data using the HTTP `get` method.

Up to here, we have a basic Material Table ready with data. Next, we will learn how to add filters for that we will update the above template and class code in coming sections.

Adding Filters On Table

In the template, add Material select boxes which will be created dynamically using *ngFor loop over the `filterSelectObj` object. The

Reset button will clear the filters and show all table data.

```
<div>
    <mat-form-field *ngFor="let filter of filterSelectObj" style=
      <mat-label>Filter {{filter.name}}</mat-label>
      <select matNativeControl name="{{filter.columnProp}}" [(ngM
        (change)="filterChange(filter,$event)">
        <option value="">-- Select {{filter.name}} --</option>
        <option [value]="item" *ngFor="let item of filter.options
      </select>
    </mat-form-field>
     
    <button mat-flat-button color="warn" (click)="resetFilters()"
  </div>

  <table mat-table [dataSource]="dataSource">

    <ng-container matColumnDef="id">
      <th mat-header-cell *matHeaderCellDef> id </th>
      <td mat-cell *matCellDef="let element"> {{element.id}} </td
    </ng-container>

    <ng-container matColumnDef="name">
      <th mat-header-cell *matHeaderCellDef> name </th>
      <td mat-cell *matCellDef="let element"> {{element.name}} </
    </ng-container>

    <ng-container matColumnDef="username">
      <th mat-header-cell *matHeaderCellDef> username </th>
      <td mat-cell *matCellDef="let element"> {{element.username}
    </ng-container>

    <ng-container matColumnDef="email">
      <th mat-header-cell *matHeaderCellDef> email </th>
      <td mat-cell *matCellDef="let element"> {{element.email}} <
    </ng-container>

    <ng-container matColumnDef="phone">
      <th mat-header-cell *matHeaderCellDef> phone </th>
      <td mat-cell *matCellDef="let element"> {{element.phone}} <
    </ng-container>

    <ng-container matColumnDef="website">
      <th mat-header-cell *matHeaderCellDef> website </th>
      <td mat-cell *matCellDef="let element"> {{element.website}}
    </ng-container>
```

```
    <ng-container matColumnDef="status">
      <th mat-header-cell *matHeaderCellDef> status </th>
      <td mat-cell *matCellDef="let element"> {{element.status}}
    </ng-container>

    <tr mat-header-row *matHeaderRowDef="displayedColumns;"></tr>
    <tr mat-row *matRowDef="let row; columns: displayedColumns;">

  </table>
```

Next, update the app.component.ts class file with following the code:

```
// app.component.ts
import { Component } from '@angular/core';
import { MatTableDataSource } from '@angular/material/table';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.scss']
})
export class AppComponent {

  filterValues = {};
  dataSource = new MatTableDataSource();
  displayedColumns: string[] = ['id', 'name', 'username', 'email'

  filterSelectObj = [];
  constructor(
  ) {

    // Object to create Filter for
    this.filterSelectObj = [
      {
        name: 'ID',
        columnProp: 'id',
        options: []
      }, {
        name: 'NAME',
        columnProp: 'name',
        options: []
      }, {
        name: 'USERNAME',
        columnProp: 'username',
        options: []
```

```
      }, {
        name: 'EMAIL',
        columnProp: 'email',
        options: []
      }, {
        name: 'STATUS',
        columnProp: 'status',
        options: []
      }
    ]
  }

  ngOnInit() {
    this.getRemoteData();

    // Overrride default filter behaviour of Material Datatable
    this.dataSource.filterPredicate = this.createFilter();
  }

  // Get Uniqu values from columns to build filter
  getFilterObject(fullObj, key) {
    const uniqChk = [];
    fullObj.filter((obj) => {
      if (!uniqChk.includes(obj[key])) {
        uniqChk.push(obj[key]);
      }
      return obj;
    });
    return uniqChk;
  }

  // Get remote serve data using HTTP call
  getRemoteData() {

    const remoteDummyData = [
      {
        "id": 1,
        "name": "Leanne Graham",
        "username": "Bret",
        "email": "Sincere@april.biz",
        "phone": "1-770-736-8031 x56442",
        "website": "hildegard.org",
        "status": "Active"
      },
      {
        "id": 2,
        "name": "Ervin Howell",
        "username": "Antonette",
        "email": "Shanna@melissa.tv",
```

```
    "phone": "010-692-6593 x09125",
    "website": "anastasia.net",
    "status": "Blocked"
  },
  {
    "id": 3,
    "name": "Clementine Bauch",
    "username": "Samantha",
    "email": "Nathan@yesenia.net",
    "phone": "1-463-123-4447",
    "website": "ramiro.info",
    "status": "Blocked"
  },
  {
    "id": 4,
    "name": "Patricia Lebsack",
    "username": "Karianne",
    "email": "Julianne.OConner@kory.org",
    "phone": "493-170-9623 x156",
    "website": "kale.biz",
    "status": "Active"
  },
  {
    "id": 5,
    "name": "Chelsey Dietrich",
    "username": "Kamren",
    "email": "Lucio_Hettinger@annie.ca",
    "phone": "(254)954-1289",
    "website": "demarco.info",
    "status": "Active"
  },
  {
    "id": 6,
    "name": "Mrs. Dennis Schulist",
    "username": "Leopoldo_Corkery",
    "email": "Karley_Dach@jasper.info",
    "phone": "1-477-935-8478 x6430",
    "website": "ola.org",
    "status": "In-Active"
  },
  {
    "id": 7,
    "name": "Kurtis Weissnat",
    "username": "Elwyn.Skiles",
    "email": "Telly.Hoeger@billy.biz",
    "phone": "210.067.6132",
    "website": "elvis.io",
    "status": "Active"
  },
```

```
      {
        "id": 8,
        "name": "Nicholas Runolfsdottir V",
        "username": "Maxime_Nienow",
        "email": "Sherwood@rosamond.me",
        "phone": "586.493.6943 x140",
        "website": "jacynthe.com",
        "status": "In-Active"
      },
      {
        "id": 9,
        "name": "Glenna Reichert",
        "username": "Delphine",
        "email": "Chaim_McDermott@dana.io",
        "phone": "(775)976-6794 x41206",
        "website": "conrad.com",
        "status": "In-Active"
      },
      {
        "id": 10,
        "name": "Clementina DuBuque",
        "username": "Moriah.Stanton",
        "email": "Rey.Padberg@karina.biz",
        "phone": "024-648-3804",
        "website": "ambrose.net",
        "status": "Active"
      }
    ];
    this.dataSource.data = remoteDummyData;

    this.filterSelectObj.filter((o) => {
      o.options = this.getFilterObject(remoteDummyData, o.columnP
    });
  }

  // Called on Filter change
  filterChange(filter, event) {
    //let filterValues = {}
    this.filterValues[filter.columnProp] = event.target.value.tri
    this.dataSource.filter = JSON.stringify(this.filterValues)
  }

  // Custom filter method fot Angular Material Datatable
  createFilter() {
    let filterFunction = function (data: any, filter: string): bo
      let searchTerms = JSON.parse(filter);
      let isFilterSet = false;
      for (const col in searchTerms) {
        if (searchTerms[col].toString() !== '') {
```

```
          isFilterSet = true;
        } else {
          delete searchTerms[col];
        }
      }

      console.log(searchTerms);

      let nameSearch = () => {
        let found = false;
        if (isFilterSet) {
          for (const col in searchTerms) {
            searchTerms[col].trim().toLowerCase().split(' ').forE
              if (data[col].toString().toLowerCase().indexOf(word
                found = true
              }
            });
          }
          return found
        } else {
          return true;
        }
      }
      return nameSearch()
    }
    return filterFunction
  }


  // Reset table filters
  resetFilters() {
    this.filterValues = {}
    this.filterSelectObj.forEach((value, key) => {
      value.modelValue = undefined;
    })
    this.dataSource.filter = "";
  }
}
```

Let's have a look at the important functions we used above.

`getFilterObject()` : This method is accepting the data and property name of the column from which we want unique values to return. It will

return the column values which are unique and updates the `filterSelectObj` 's options property for each filter specified.

`this.dataSource.filterPredicate` : The `filterPredicate` method can be overridden to use custom logic for table filtration. Here we used the `createFilter()` method to filter out rows on the filter selected.

`filterChange()` : This method is called on each filter selected.

That's it you can run the app by executing `$ ng serve --open` to see it in action.

Also Check

- **Angular 8|9 Material Table Column Width, Text Alignment Customiz...**
- **Angular + Material | How to Install Angular Material in Angular Proj...**
- **Angular Material 8 | Adding Multiple Tables on Single Component T...**
- **Angular 7|8|9 Edit/ Add/ Delete Rows in Material Table with using ...**
- **Angular Material 7 | Data Tables Example with Sorting, Filter, Pagin...**
- **Angular Material 8 | Implement Dialog Modal and Pass Data betwe...**
- **Angular Material 8/9 Tree Tutorial By Example**
- **Angular 8/7 | How to Add Tooltips using Angular Material compone...**

Posted in **Angular**, **Angular 7**, **Angular 8**, **Angular 9**, **Angular Material**

Tutorials

Angular

Ionic

Firebase

Similar Posts

Subscribe to Latest Tutorials

email address Subscribe