Guía de migración de aplicaciones angular de 1.1.x a 1.2.x

24 de Marzo de 2017

Inscripción Registro Mercantil de Pontevedra, Tomo 3116, Libro 3116, Folio 30, Hoja PO-38276
C.I.F.: B-36.499.960 · contact@imatia.com · www.ontimize.com

Índice de contenidos

1 Introducción

En este documento se explica paso a paso como actualizar una aplicación web realizada con la versión 1.1.x del framework Ontimize Web a las versiones 1.2.x.

1. Actualizar dependencias en el fichero *package.json*
2. Actualizar el fichero *system-config.ts*
3. Actualizar el componente principal de la aplicación *o-app* (*app.component.ts* y *app.component.html*)
4. Actualizar el módulo principal de la aplicación *app.module.ts*
5. Actualizar los componentes

2 Package.json

Modificar las siguientes dependencias en el fichero package.json

- Actualizar la versión de los módulos estándar angular2 a `2.4.2`
- Actualizar los siguientes módulos
  - `@angular/router versión 3.4.2`
  - `@angular/material versión 2.0.0-beta.1`
  - `rxjs versión 5.0.1`
  - `zone.js versión 0.7.4`
  - `moment versión 2.17.1`
  - `ng2-translate versión 5.0.0`
- Subir versión de `ontimize-web-ng2 a 1.2.x`

Para realizar las actualizaciones de versiones se recomienda ejecutar la consola con permisos de administrador y previamente borrar las carpetas de las versiones antiguas contenidas en *node_modules*.

3 System-config.ts

En el fichero *system-config.ts* se deben modificar algunas dependencias, mostradas a continuación:

```
    const map: any = {
      ...

      // Remove this
      '@angular/material': 'vendor/@angular/material/material.umd.js'

      // Add this
      @angular/material': 'vendor/@angular/material/bundles/material.umd.js'

      ...
    };

    ...

    // Remove this
    cliSystemConfigPackages['ng2-translate'] = { main: 'ng2-translate' };

    // Add this
    cliSystemConfigPackages['ng2-translate'] = { main: 'bundles/ng2-translate.umd.js',
defaultExtension: 'js'};

    ...
```

4 app.component

El componente *o-app* (*app.component*) debe ser modificado para eliminar las referencias al *ODialogComponent*. Como se muestra a continuación.

Antes:

```
import { Component, OnInit, ViewChild, Inject } from '@angular/core';
import { DialogService, ODialogComponent } from 'ontimize-web-ng2/ontimize';

@Component({
  moduleId: module.id,
  selector: 'o-app',
  templateUrl: 'app.component.html',
  styleUrls: ['app.component.css']
})
export class AppComponent implements OnInit{

  @ViewChild('dialog')
  protected dialog: ODialogComponent;

  constructor(@Inject(DialogService) private dialogService: DialogService) {
    }

  ngOnInit() {
    this.dialogService.dialog = this.dialog;
  }
}
```

Después:

```
import { Component, OnInit } from '@angular/core';

@Component({
  moduleId: module.id,
  selector: 'o-app',
  templateUrl: 'app.component.html',
  styleUrls: ['app.component.css']
})
export class AppComponent implements OnInit {

  constructor() {
  }

  ngOnInit() {
  }
}
```

También se debe modificar su plantilla (*app.component.html*) eliminando la referencia al diálogo.

Antes :

```
<router-outlet></router-outlet>

<o-dialog #dialog></o-dialog>
```

Después:

```
<router-outlet></router-outlet>
```

5 app.module.ts

En el módulo principal de la aplicación se debe hacer referencia al componente *ODialogComponent* de la siguiente forma:

```
import { NgModule } from '@angular/core';

import {
  ONTIMIZE_MODULES,
  ONTIMIZE_DIRECTIVES,
  ontimizeProviders,
  // New import
  ODialogComponent
} from 'ontimize-web-ng2/ontimize';

import { CONFIG } from './app.config';
import { AppComponent } from './app.component';
import { routing } from './app.routes';
import { APP_DIRECTIVES } from './app.directives';

// Standard providers...
let standardProviders = ontimizeProviders({
  'config': CONFIG
});
// Defining custom providers (if needed)...
// let customProviders = [
// ];

@NgModule({
  imports: [
    ONTIMIZE_MODULES,
    routing
  ],
  declarations: [
    AppComponent,
    ONTIMIZE_DIRECTIVES,
    ...APP_DIRECTIVES
  ],
  // New property
  entryComponents: [
    ODialogComponent
  ],
  bootstrap: [
    AppComponent
  ],
  providers: [
    ...standardProviders
    // ...customProviders
  ]
})
export class AppModule { }
```

6 Actualizar los componentes

Existen una serie de cambios en los componentes que pueden afectar al funcionamiento de la aplicación:

BREAKING CHANGES (https://github.com/OntimizeWeb/ontimize-web-ng2/blob/master/CHANGELOG.md)

- Interfaces: The interface 'IFormComponent' was renamded to 'IComponent'.
- Components: Several components have changed its constructor.
- o-selectable-list: Deleted component (use 'selectable' attribute in o-list instead).
- o-md-selectable-list-item: Deleted directive (use 'selectable' attribute in o-list instead).
- o-selectable-list-item: Deleted component (use 'selectable' attribute in o-list instead).
- o-list: 'onReload' method changed (marked as deprecated) for 'reloadData'.
- o-table: 'onReload' method changed (marked as deprecated) for 'reloadData'.
- o-table: 'data' attribute changed for 'static-data'.
- o-table-button: 'click' output changed for 'onClick'.
- o-table-option: 'click' output changed for 'onClick'.

6.1 Componente o-form

Si en la aplicación se ha realizado alguna extensión del componente *OFormComponent* se debe actualizar su constructor para añadir un nuevo argumento.

Antes :

```
...

constructor(router: Router,
  actRoute: ActivatedRoute,
  zone: NgZone,
  cd: ChangeDetectorRef,
  injector: Injector
) {
  super(router, actRoute, zone, cd, injector);
}

...
```

Después:

```
...

constructor(router: Router,
  actRoute: ActivatedRoute,
  zone: NgZone,
  cd: ChangeDetectorRef,
  injector: Injector,
  elRef: ElementRef
) {
  super(router, actRoute, zone, cd, injector, elRef);
}

...
```