Branch: master ▾ Find file Copy path

angularfire2 / docs / install-and-setup.md

ashishpatelcs docs(): typo and grammar fixes in docs (#1931)

b2d44a8 on 23 Oct 2018

6 contributors

Raw Blame History

197 lines (150 sloc) 5.23 KB

1. Installation and Setup

> Using Ionic and the Ionic CLI? Check out these specific instructions for Ionic and their CLI.

0. Prerequisites

AngularFire provides multiple module formats for different types of builds. The guide is based on the Angular CLI. It is possible to do a manual setup with Webpack or a SystemJS build as well.

```
npm install @angular/cli
```

1. Create a new project

```
ng new <project-name>
cd <project-name>
```

The Angular CLI's `new` command will set up the latest Angular build in a new project structure.

2. Test your setup

```
ng serve
open http://localhost:4200
```

You should see a message on the page that says *App works!*

3. Install AngularFire and Firebase

```
npm install @angular/fire firebase --save
```

Now that you have a new project setup, install AngularFire and Firebase from npm.

4. Add Firebase config to environments variable

Open `/src/environments/environment.ts` and add your Firebase configuration. You can find your project configuration in the Firebase Console. From the project overview page, click **Add Firebase to your web app**.

```
export const environment = {
  production: false,
  firebase: {
    apiKey: '<your-key>',
    authDomain: '<your-project-authdomain>',
    databaseURL: '<your-database-URL>',
    projectId: '<your-project-id>',
    storageBucket: '<your-storage-bucket>',
    messagingSenderId: '<your-messaging-sender-id>'
  }
};
```

5. Setup @NgModule for the AngularFireModule

Open `/src/app/app.module.ts`, inject the Firebase providers, and specify your Firebase configuration.

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { AppComponent } from './app.component';
import { AngularFireModule } from '@angular/fire';
import { environment } from '../environments/environment';

@NgModule({
  imports: [
    BrowserModule,
    AngularFireModule.initializeApp(environment.firebase)
  ],
  declarations: [ AppComponent ],
  bootstrap: [ AppComponent ]
})
export class AppModule {}
```

Custom `FirebaseApp` names

You can optionally provide a custom FirebaseApp name with `initializeApp`.

```
@NgModule({
  imports: [
    BrowserModule,
    AngularFireModule.initializeApp(environment.firebase, 'my-app-name')
  ],
  declarations: [ AppComponent ],
  bootstrap: [ AppComponent ]
})
export class AppModule {}
```

6. Setup individual `@NgModules`

After adding the AngularFireModule you also need to add modules for the individual @NgModules that your application needs.

- `AngularFireAuthModule`
- `AngularFireDatabaseModule`
- `AngularFireFunctionsModule`
- `AngularFirestoreModule`
- `AngularFireStorageModule`
- `AngularFireMessagingModule`

Adding the Firebase Database and Auth Modules

For example if your application was using both Firebase authentication and the Firebase database you would add:

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { AppComponent } from './app.component';
import { AngularFireModule } from '@angular/fire';
import { AngularFirestoreModule } from '@angular/fire/firestore';
import { AngularFireStorageModule } from '@angular/fire/storage';
import { AngularFireAuthModule } from '@angular/fire/auth';
import { environment } from '../environments/environment';

@NgModule({
  imports: [
    BrowserModule,
    AngularFireModule.initializeApp(environment.firebase, 'my-app-name'), // impc
    AngularFirestoreModule, // imports firebase/firestore, only needed for databa
    AngularFireAuthModule, // imports firebase/auth, only needed for auth feature
    AngularFireStorageModule // imports firebase/storage only needed for storage
  ],
  declarations: [ AppComponent ],
  bootstrap: [ AppComponent ]
```

```
})
export class AppModule {}
```

7. Inject `AngularFirestore`

Open `/src/app/app.component.ts`, and make sure to modify/delete any tests to get the sample working (tests are still important, you know):

```
import { Component } from '@angular/core';
import { AngularFirestore } from '@angular/fire/firestore';

@Component({
  selector: 'app-root',
  templateUrl: 'app.component.html',
  styleUrls: ['app.component.css']
})
export class AppComponent {
  constructor(db: AngularFirestore) {

  }
}
```

8. Bind a Firestore collection to a list

In `/src/app/app.component.ts`:

```
import { Component } from '@angular/core';
import { AngularFirestore } from '@angular/fire/firestore';
import { Observable } from 'rxjs';

@Component({
  selector: 'app-root',
  templateUrl: 'app.component.html',
  styleUrls: ['app.component.css']
})
export class AppComponent {
  items: Observable<any[]>;
  constructor(db: AngularFirestore) {
    this.items = db.collection('items').valueChanges();
  }
}
```

Open `/src/app/app.component.html`:

```
<ul>
  <li class="text" *ngFor="let item of items | async">
    {{item.name}}
```

```
    </li>
  </ul>
```

9. Run your app

```
ng serve
```

Run the serve command and navigate to `localhost:4200` in your browser.

And that's it! If it's totally *borked*, file an issue and let us know.

Next Step: Documents in AngularFirestore