

Toogler

```
<nav class="navbar navbar-  
expand-lg navbar-dark bg-dark">  
    <a class= " nav bar -br -  
and " href="#" ">  
        Brand  
    </a>  
    <button class= " nav bar -  
to ggl er" type="b utt on"  
        (cl -  
ick )="t ogg leN avb ar( )">  
        <span class= " nav -  
bar -to ggl er- ico n">< /sp an>  
    </b utt on>  
    <div class= " col lapse  
navbar -co lla pse "  
        [ng Cla ss] ="({  
'show': navbarOpen })">  
        <ul class= " nav bar-nav  
mr-aut o">  
            <li class= " nav -  
it em">  
                <a class= " -  
nav -li nk" href="#" ">Item  
1</ a>  
            </l i>  
            <li class= " nav -  
it em">  
                <a class= " -  
nav -li nk" href="#" ">Item  
2</ a>  
            </l i>  
        </u l>  
    </d iv>  
</n av>  
  
-----  
- -----  
- -----  
- -----  
- -----
```

Toogler (cont)

```
> toggleNavbar() {  
  this.navbarOpen = !this.navbarOpen;  
}
```

Document

```

getProduct(id: number):
  Observable<Product> {
    const products = this.db.products
    return products.find(p => p.id === id)
      .pipe(
        map(changes =>
          {
            const data = changes[0].new
            const id = changes[0].old.id
            return { id, ...data }
          }
        ))
  }
}

```

Collection

```

getProduct(id: string):
Observable<Product[]> {
    const produc tsD ocu -
ments = this.d b.c oll ect ion -
<Pr odu ct[ ]>( 'pr odu cts');
    return produc tsD ocu -
men ts.s na psh otC han ges()
        .pipe(
            map (ch anges =>
change s.m ap(({ payload: { doc
} }) => {

```

Collection (cont)

```
> const data = doc.data();  
const id = doc.id  
return { id, ...data };  
))),  
map((products) => products.find(doc  
=> doc.id === id)))  
}
```

Add/Create A Document To Cloud Firestore

```
const db = firebase.firestore()

db.collection('users').add({
  name: "Anbu Selvan ",
  email: "anbu.selv -
an@email.com ",
  age: 25
})

db.collection('users').doc()
.set({
  name: "Anbu Selvan ",
  email: "anbu.selv -
an@email.com ",
  age: 25
})
```

Update A Document Data to Cloud Firestore

```
db.collection("users")
  .doc("3 P86 VJx cpB K0D 0ls -
0ls AyY x")
  .set({
    name: "Lee
Kuan",
  });
db.collection("users")
  .doc("3 P86 VJx cpB K0D 0ls -
AyY x")
  .set(
    {
      name: "Anbu
Selvan ",
      age: 25
    },
    { merge: true }
  );
```

Delete Data from Cloud Firestore

```
db.collection("users")
  .doc("3 P86 VJx cpB K0D 0ls -
AyY x")
  .delete()
  .then(function () {
    console.log(" Document
successfully delete d!");
  }).catch(
    function(error) {
      console.error("Error
removing document: ", error);
    });
```

Delete Data from Cloud Firestore (cont)

```
> db.collection("users")
.doc("3P86VJxcpBK0D0lsAyYx")
.update({
  email: firebase.FieldValue.delete()
});
```

Pretty straight forward.

```
addUsersToFirestore() {
  var users = [{
    name:
" Raj a",
    email:
" raj a.t ami l@e mai l.c om",
    createdAt: new Date("2 019 -01-01
12:08: 00")
  }, {
    name:
" Ari vu",
    email:
" ari vu.s el van @em ail.co m",
    createdAt: new Date("2 018 -01-23
09:13: 00")
  }, {
    name:
" Mik e",
    email:
" mik e.a uth or@ ema il.c -
om ",
    createdAt: new Date("2 018 -08-08
06:37: 00")
  }, {
    name:
" Pra ba",
    email:
" pra ba.k ar an@ ema il.c -
om ",
    createdAt: new Date("2 018 -10-09
18:26: 00")
  }];
```

Pretty straight forward. (cont)

```
> },
{
  name: "Muhammad",
  email: "muhammad.ali@email.com",
  createdAt: new Date("2018-03-13
12:13:00")
};
const db = firebase.firestore();
users.forEach(user => {
  db.collection("users").doc().set(user);
});
}
```

Get Documents Data from Firestore Database

```
db.collection("users")
  .get()
  .then(snap => {
    snap.forEach(doc => {
      console.log(doc.data());
      console.log(doc.id);
    });
  });

db.collection("users")
  .onSnapshot()
  .then(snap => {
    snap.forEach(doc => {
      console.log(doc.data());
    });
  });
```



Get A Single Document Data

```
db.collection("users")
.doc("c AwT iq7 IYK AbF Gnh -
gKT 3")
.get()
.then(doc => {
    console.log(doc.data())
})
```

Get Data from Sub-collection in Firestore

```
db.collection("users")
.doc("c AwT iq7 IYK AbF Gnh -
gKT 3")
.collection("posts")
.get()
.then(snap => {
    snap.forEach(doc => {
        console.log(doc.data());
    });
});

db.collection("users")
.doc("c AwT iq7 IYK AbF Gnh -
gKT 3")
.collection("posts")
.doc("B jLZ Hiu QfV QVO u9n -
EG7 k")
.get()
.then(snap => {
    console.log(snap.data());
})
```

Firestore Single/Multiple Where Query Filter

```
db.collection("users")
.where("age", "<=", 30)
.get()
.then(snap => {
    snap.forEach(doc => {
        console.log(doc.data());
    });
});

db.collection("users")
.where("age", "<=", 30)
.where("age", ">=", 20)
.get()
.then(snap => {
    snap.forEach(doc => {
        console.log(doc.data());
    });
});

db.collection("users")
.where("age", ">=", 20)
.where("gender", "==", "female")
.get()
.then(snap => {
    snap.forEach(doc => {
        console.log(doc.data());
    });
});
```

Firestore Single/Multiple Where Query Filter (cont)

```
> });
});
```

OrderBy and Limit Filters

```
db.collection("users")
.where("age", ">=", 20)
.orderBy("age", "desc")
.get()
.then(snap => {
    snap.forEach(doc => {
        console.log(doc.data());
    });
});
```

```
db.collection("users")
.where("age", ">=", 20)
.orderBy("age", "desc")
.limit(2)
.get()
.then(snap => {
    snap.forEach(doc => {
        console.log(doc.data());
    });
});
```

Collection Group Queries

```
users/{userID}/posts/{postID}
db.collectionGroup("posts")
.where("publishedAt", ">=", new Date("2018-01-01 00:00"))
.where("publishedAt", "<=", new Date("2018-12-31 23:59"))
.get()
.then(snap => {
    snap.forEach(doc => {
        console.log(doc.data());
    });
});
```

Collection Group Queries (cont)

```
> });
});
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write;
    }
  }
}
var cities = [];
const cityRef = firebase.firestore().collection("cities")
var lastVisibleCitySnapshot = {};
const query = await this.cityRef.orderBy("-city").limit(10);
query.get().then(snap => {
  snap.forEach(doc => {
    this.cities.push(doc.data());
  });
  this.lastVisibleCitySnapshot = snap.docs[snap.docs.length - 1];
});
async next() {
  this.cities = [];
  const query = await this.cityRef
```

Collection Group Queries (cont)

```
> .orderBy("city")
.startAfter(this.lastVisibleCitySnapshot)
.limit(10);
query.get().then(snap => {
  snap.forEach(doc => {
    this.cities.push(doc.data());
  });
  this.lastVisibleCitySnapshot = snap.docs[snap.docs.length - 1];
});
}
async prev() {
  this.cities = [];
  const query = await this.cityRef
    .orderBy("city")
    .endBefore(this.lastVisibleCitySnapshot)
    .limit(10);
  query.get().then(snap => {
    snap.forEach(doc => {
      this.cities.push(doc.data());
    });
    this.lastVisibleCitySnapshot = snap.docs[snap.docs.length - 1];
  });
}
```

Collection Group Queries (cont)

```
> },
```



By **john2222**

cheatography.com/john2222/

Not published yet.

Last updated 2nd February, 2020.

Page 4 of 4.

Sponsored by **Readable.com**

Measure your website readability!

<https://readable.com>