

Toogller

```
<nav class="navbar navbar-
expand-lg navbar-dark bg-dark">
  <a class="navbar-brand"
href="#">
    Brand
  </a>
  <button class="navbar-toggler"
type="button"
    (click)="toggleNavb-
ar()">
    <span class="navbar-toggl-
er-icon"></span>
  </button>
  <div class="collapse navbar-
collapse"
    [ngClass]="{ 'show':
navbarOpen }">
    <ul class="navbar-nav mr-
auto">
      <li class="nav-item">
        <a class="nav-link"
href="#">Item 1</a>
      </li>
      <li class="nav-item">
        <a class="nav-link"
href="#">Item 2</a>
      </li>
    </ul>
  </div>
</nav>
-----
-----
-----
-----
-----
navbarOpen = false;
toggleNavbar() {
  this.navbarOpen = !this.nav-
barOpen;
}
```

Document

```
getProduct(id: number):
Observable<Product> {
  const productsDocuments =
this.db.doc<Product>('products/'
+ id);
  return productsDocuments.s-
napshotChanges()
    .pipe(
      map(changes => {
        const data = change-
s.payload.data();
        const id = changes.p-
ayload.id;
        return { id, ...data
      });
    )
}
```

Collection

```
getProduct(id: string):
Observable<Product[]> {
  const productsDocuments =
this.db.collection<Product[]>
('products');
  return productsDocuments.s-
napshotChanges()
    .pipe(
      map(changes => change-
s.map(({ payload: { doc } }) =>
{
        const data = doc.da-
ta();
        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.selvan@email.c-
om",
  age: 25
})
db.collection("users")
.doc()
.set({
  name: "Anbu Selvan",
  email: "anbu.selvan@email.c-
om",
  age: 25
})
```

Update A Document Data to Cloud Firestore

```
db.collection("users")
.doc("3P86VJxcpBK0D0lsAyYx")
.set({
  name: "Lee Kuan",
});
db.collection("users")
.doc("3P86VJxcpBK0D0lsAyYx")
.set(
{
  name: "Anbu Selvan",
  age: 25
},
{ merge: true }
);
```



Delete Data from Cloud Firestore

```
db.collection("users")
.doc("3P86VJxcpBK0D0lsAyYx")
.delete()
.then(function () {
  console.log("Document
successfully deleted!");
}).catch(
function(error) {
  console.error("Error
removing document: ", error);
});
db.collection("users")
.doc("3P86VJxcpBK0D0lsAyYx")
.update({
  email.firestore.FieldVal-
ue.delete()
})
```

Pretty straight forward.

```
addUsersToFirestore() {
  var users = [{
    name: "Raja",
    email: "raja.tami-
l@email.com",
    createdAt: new
Date("2019-01-01 12:08:00")
  },
  {
    name: "Arivu",
    email: "arivu.sel-
van@email.com",
    createdAt: new
Date("2018-01-23 09:13:00")
  }, {
    name: "Mike",
    email: "mike.auth-
or@email.com",
```

Pretty straight forward. (cont)

```
    createdAt: new
Date("2018-08-08 06:37:00")
  }, {
    name: "Praba",
    email: "praba.kar-
an@email.com",
    createdAt: new
Date("2018-10-09 18:26:00")
  },
  {
    name: "Muhammad",
    email: "muhammad.a-
li@email.com",
    createdAt: new
Date("2018-03-13 12:13:00")
  }
];
const db = firebase.firest-
ore();
users.forEach(user => {
  db.collection("user-
s").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("cAwTiq7IYKAbFGnhgKT3")
.get()
.then(doc => {
  console.log(doc.data())
})
```

Get Data from Sub-collection in Firestore

```
db.collection("users")
.doc("cAwTiq7IYKAbFGnhgKT3")
.collection("posts")
.get()
.then(snap => {
  snap.forEach(doc => {
    console.log(doc.data());
  });
});
db.collection("users")
.doc("cAwTiq7IYKAbFGnhgKT3")
.collection("posts")
.doc("BjLZHiuQfVQVOu9nEG7k")
.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());
    });
  });
```

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());
    });
  });

service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write;
    }
  }
}

var cities = [];
```

Collection Group Queries (cont)

```
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
    .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 = [];
```



Collection Group Queries (cont)

```
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.doc-
ocs.length - 1];
});
},
```

C

By [john2222](#)

cheatography.com/john2222/

Not published yet.

Last updated 2nd February, 2020.

Page 4 of 4.

Sponsored by [CrosswordCheats.com](#)

Learn to solve cryptic crosswords!

<http://crosswordcheats.com>