
Firebase

Firebase är en såkallad **Backend as a Service**

Vi behöver inte bry oss om hanteringen av databasen

Databasen blir som ett API vi kallar på

Istället för att behöva sätta upp en MySQL-databas t.ex.

MySQL är en **relationell databas**: **tabeller**

Firebase är **noSQL**: **objekt**

Vi kommer att jobba med en JSON-struktur

```
todos: {
  todo1: {
    title: "Learn Firebase",
    completed: false
  }
}
```

```
todos: {
  todo1: {
    title: "Learn Firebase",
    completed: false
  },
  todo2: {
    title: "Learn React",
    completed: false
  }
}
```

Det betyder att vi behöver gräva ibland

```
todos: {
  todo1: {
    title: "Learn Firebase",
    completed: false
  }
}

console.log(todos.todo1.completed);
```

Firebase underlättar dock detta för oss

Store and sync data between users and devices in realtime using a cloud-hosted, noSQL database

Vi är i molnet 🌤️, betyder bara att det är någon annan stackars jobb att få det att fungera ÅT oss 🤔

Alltid samma sak oavsett: CRUD

CREATE **READ** **UPDATE** **DELETE**

Bara ett annat API för att göra det

- Gå till https://firebase.google.com/
- Registrera konto på Firebase
- Tryck på Get Started
- Tryck på Create new Project
- Döp projektet och välj "Sweden" i dropdown
- **Nu har du en databas!**


READ / WRITE

Allting baserar sig på HTTP(S): GET / POST

Firebase har sitt egna API för detta

Firebase använder sig mest av **Listeners**

Istället för att lyssna efter `onClick` t.ex. så lyssnar Firebase på uppdateringar i databasen

Firebase lyssnar till referenser till olika objekt

```
firebase.database().ref("posts")
```

```
firebase.database().ref("users")
```

```
firebase.database().ref("comments")
```

Istället för tabeller har vi objekt, samt referenser till dessa objekt

Ska vi komma åt ett speciellt värde med ID:

```
firebase.database().ref("movie/id")
```

Har det värdet flera värden?

```
firebase.database().ref("movie/id/imdbRating")
```

Ännu mer nästlat??

```
firebase.database().ref("movie/id/genres/action")
```

Allting är **key/value**-objekt

För att faktiskt skriva och hämta data så har vi följande API

```
firebase.database()
  .ref("messages")
  .push("HELLO FROM SLIDES")
```

Ovan lagrar meddelandet i samlingen `"messages"` med **ett unikt ID**

Manipulera data, skriva och ta bort

```
.push() //POST
.set() //UPDATE
.remove() //DELETE
```

Läsa data

```
.on() //GET but REALTIME, GETS REAL!
.once() //TRADITIONAL GET
```

`on` & `.once()` har callbacks

```
firebase.database().ref("messages")
  .on('value', function(snapshot){
    console.log(snapshot.val());
});
```

Vi måste plocka ut våra värden med `.val()`

`.on` returnerar en `"snapshot"`, en bild av hur vår databas ser ut för tillfället.

Den är typ som ett promise, dv.s. massa metadata, vi måste plocka ut värdena med:

```
snapshot.val()
```

Snapshot måste inte heta snapshot

```
firebase.database().ref("messages")
  .once('value', function(snapshot){
    console.log(snapshot.val());
});
```

`.once()` är mer som en klassisk FETCH

Många funktioner returnerar även ett promise

```
firebase.database().ref("messages")
  .once('value')
  .then(snapshot => {
    console.log(snapshot.val());
  });
```

T.ex. `.set()` och `.push()`

KODEXEMPEL

Hämta och skicka data till databasen

Eftermiddagen:

Övning: Implementera TODO med firebase

(Leka med WebVR? 🤑)