

Callback

- Eine Funktion, die später aufgerufen wird, wenn etwas fertig ist.

```
function verdoppeln(zahl, callback) {  
  callback(zahl * 2);  
}
```

- Ältere Methode, Kann schnell zur „Callback-Hölle führen“

Promise

- Ein Objekt, das ein zukünftiges Ergebnis repräsentiert.

```
return new Promise((resolve, reject) => {  
  resolve("ok");  
});
```

- .then() & .catch()
- Pending, Fulfilled, Rejected

Async / Await

- Syntax auf Basis von Promises, aber sieht aus wie synchroner Code.

```
async function test() {  
  try {  
    const result = await promise;  
  } catch(err) {  
  }  
}
```

- Am lesbarsten (asynchroner Code sieht synchron aus)
- Nutzt intern trotzdem Promises (await wartet auf einen Promise)

HTTP:

- Client/Server, Stateless, Request -> Response

Bestandteile einer Abfrage:

- URL, Methode (POST, PUT, ...), Header, Body, Statuscode

Wichtige StatusCodes:

200 = OK	201 = Created	204 = No Content				500 = Server Error
400 = Bad Request	401 = Unauthorized	403 = Forbidden	404 = Not Found	422 = Unprocessable Content	418 = I'm a Teapot	

Array-Methoden:

- users.find(u => u.id === 1); // Erstes passendes Element
- users.filter(u => u.active); // Alle passenden Elemente
- users.map(u => u.name); // Transformiert Array
- users.reduce((acc, u) => acc + u.age, 0); // Reduziert auf einen Wert (Zusammenrechnen)

Spread Operator:

```
const newArray = [...array, neuesElement];  
  
const newObj = {...obj, name: "Neu"};
```

Express:

```
const express = require('express');  
const app = express();  
app.use(express.json());  
  
app.get('/books', (req, res) => {  
  res.status(200).json(data);  
});
```