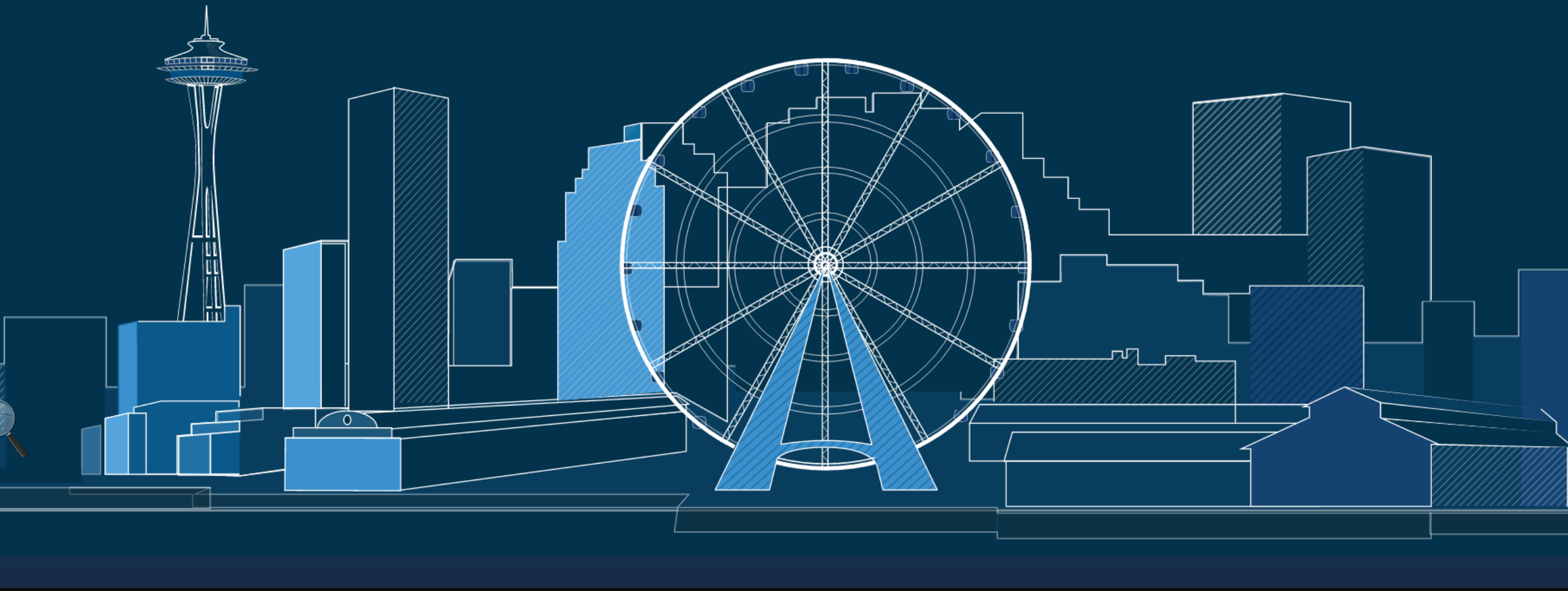


Asynchronität mit Promises

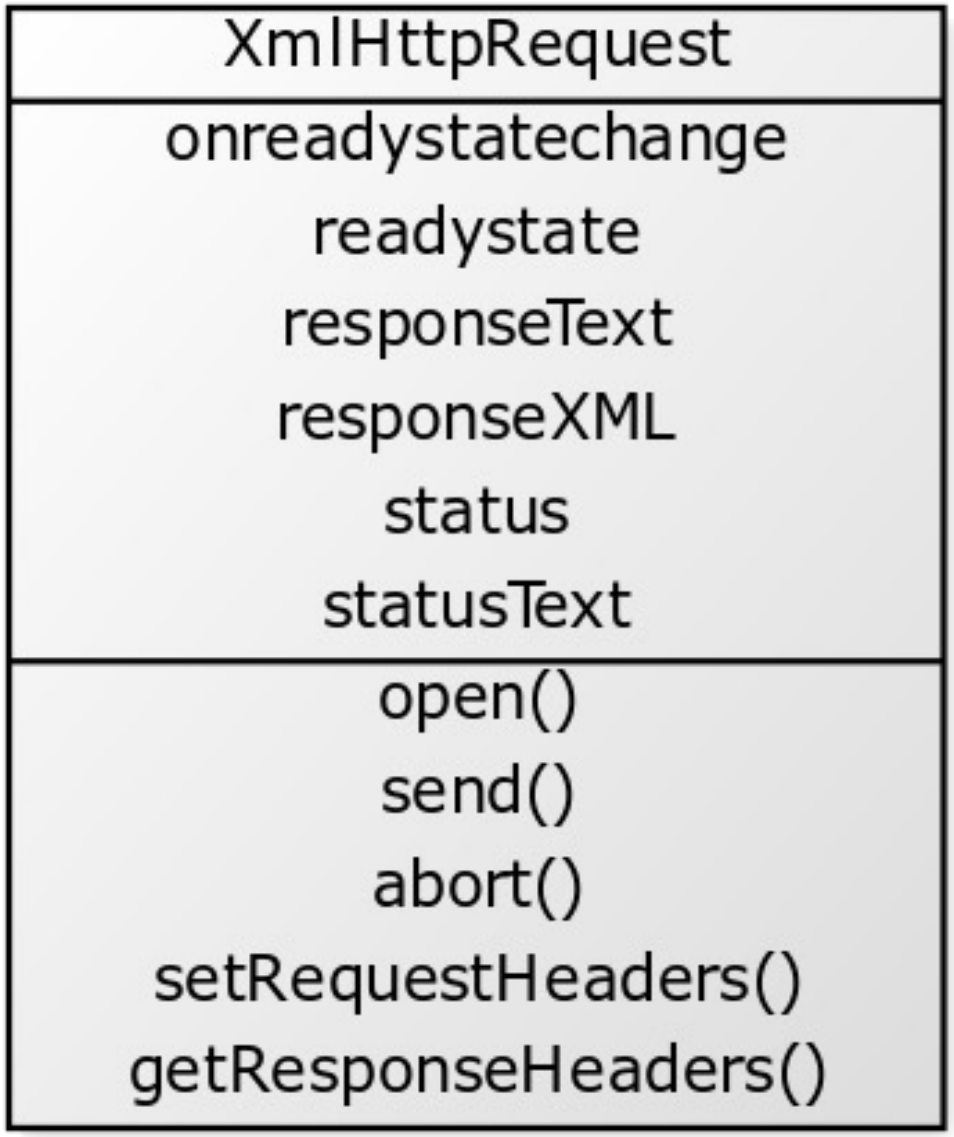
Alex Thalhammer

Inhalt

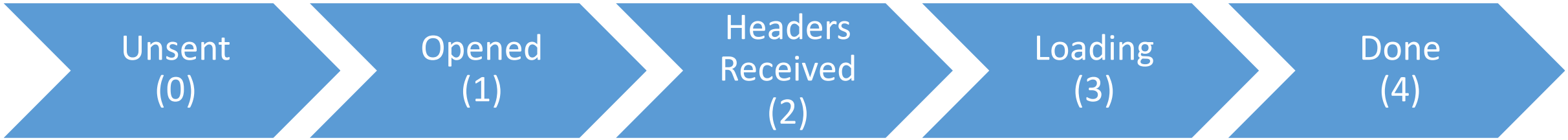
- Http-Zugriff
- DEMO: Erste Lösung
- Promises
- DEMO: Bessere Lösung

HTTP-Zugriff als Beispiel für asynchrone Operationen

XHR


XmlHttpRequest
onreadystatechange
readystate
responseText
responseXML
status
statusText
open()
send()
abort()
setRequestHeaders()
getResponseHeaders()

Readystate



XHR

```
function loadData() {
    var xmlhttp;
    xmlhttp = new XMLHttpRequest();

    […]
}
```

XHR

```
function loadData() {
    var xmlhttp;
    xmlhttp = new XMLHttpRequest();

    xmlhttp.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
            console.debug(this.responseText);
        }
    };

    […]
}
```

XHR

```
function loadData() {
    var xmlhttp;
    xmlhttp = new XMLHttpRequest();

    xmlhttp.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
            console.debug(this.responseText);
        }
    };

    xmlhttp.open("GET", "http://...", true);
    xmlhttp.send();
}
```

DEMO

Promises

Promises

- Definiert einen Wert, der (noch) nicht zwingend bekannt ist.
- Zustände:
  - pending, fulfiled, rejected

Promises

```
function loadData() {
    return new Promise((resolve, reject) => {
        //Async:
        resolve(4711);
        //reject("err!");
    });
}
```

Promises

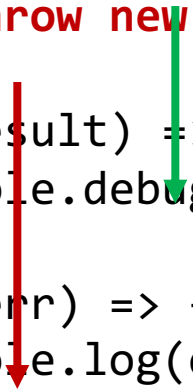
```
function loadData() {
    return new Promise((resolve, reject) => {
        //Async:
        resolve(4711);
        //reject("err!");
    });
}

loadData()
    .then((result) => {
        console.log(result);
    })
    .catch((err) => {
        console.log(err);
    });
```

Wird sofort ausgeführt!

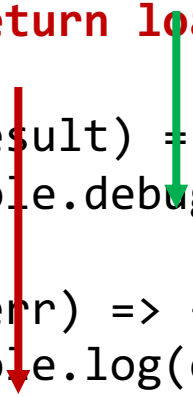
Promise Chaining

```
loadData()
    .then((result) => {
        return 'XYZ';
        // throw new Error();
    })
    .then((result) => {
        console.debug(result == 'XYZ');
    })
    .catch((err) => {
        console.log(err);
    });
```



Promise Chaining

```
loadData()
    .then((result) => {
        return loadData(…); // erfolgreich
        // return loadData(…); // nicht erfolgreich
    })
    .then((result) => {
        console.debug(result == 'XYZ');
    })
    .catch((err) => {
        console.log(err);
    });
```



DEMO