Web Storage
LocalStorage / SessionStorage

Client-seitiges Speichern von Daten ermöglicht die Web Storage API.
Web Storage hat den Vorteil, dass die Daten auf dem Computer des Nutzers bleiben und er die Kontrolle über seine Daten hat. Wenn der Browsercache geleert wird, werden alle gespeicherten Daten gelöscht.

Es gibt zwei Möglichkeiten des Web Storage.

Beim **sessionStorage** werden die Daten für eine Session gespeichert, die Daten werden gelöscht, wenn der Browsertab geschlossen wird.

localStorage speichert dagegen die Daten unbegrenzt, ohne Ablaufdatum. Sie sind auch nach Schließen des Browsertabs wieder abrufbbar.

Die Methoden zum Bearbeiten sind für beide Storage-Arten gleich.

Das Storage bietet die Möglichkeit, Daten in Form eines Key-Value-Paares zu speichern.
Dabei wird sowohl für den Key als auch für den Value ein **String** erwartet, d.h. dass Javascript-Objekte zum Speichen erst in einen String umgewandelt werden müssen. Gleichzeitig erhalten wir beim “Auslesen“ des Storage auch einen String, der zum weiteren Be- und Verarbeiten wieder in ein Javascript-Objekt umgewandelt werden muss (siehe auch **JSON.parse() / JSON.stringify()**)

.setItem()

Mit .setIten() werden die Key-Value-Paare sowohl im sessionStorage als auch im localStorage gespeichert.

```
localStorage.setItem(`Bewohner von Entenhausen`, `Micky Maus`);
console.log(localStorage);
// gibt auf der Console aus:

Storage {Bewohner von Entenhausen: "Micky Maus", length: 1}
```

ACHTUNG: Wenn ein Eintrag mit diesem Keynamen schon besteht, wird er ohne Warnung überschrieben.

.getItem()

Mit .getItem() wird der Value des entsprechenden Keys aus dem Storage ausgelesen.

```
console.log(localStorage.getItem(`Bewohner von Entenhausen`));
// gibt auf der Console aus:

Micky Maus
```

ACHTUNG: Wenn ein Eintrag mit diesem Keynamen nicht besteht, wird **null** zurückgegeben.

```
console.log(localStorage.getItem(`Bewohner`));
// gibt auf der Console aus:

null
```

.removeItem()

Mit .removeItem() wird ein ein Key-Value-Paar komplett aus dem Storage gelöscht, andere Key-Value.Paare bleiben davon unberührt.

```
localStorage.setItem(`Bewohner von Entenhausen`, `Micky Maus`);
localStorage.setItem(`Weiterer Bewohner von Entenhausen`, `Donald Duck`);
console.log(localStorage);
// gibt auf der Console aus:

Storage {Weiterer Bewohner von Entenhausen: "Donald Duck", Bewohner von Entenhausen: "Micky Maus", length: 2}


localStorage.removeItem(`Weiterer Bewohner von Entenhausen`);
console.log(localStorage);
// gibt auf der Console aus:

Storage {Bewohner von Entenhausen: "Micky Maus", length: 1}
```

Hier wurde also vom localStorage das Key-Value-Paar “Weiterer Bewohner von Entenhausen“, “Donald Duck“ entfernt.

.clear()

Mit .clear wird das entsprechende Storage unter dieser Domain komplett gelöscht.

```
localStorage.setItem(`Bewohner von Entenhausen`, `Micky Maus`);
localStorage.setItem(`Weiterer Bewohner von Entenhausen`, `Donald Duck`);
console.log(localStorage);
// gibt auf der Console aus:

Storage {Weiterer Bewohner von Entenhausen: "Donald Duck", Bewohner von Entenhausen: "Micky Maus", length: 2}


localStorage.clear();
console.log(localStorage);
// gibt auf der Console aus:

Storage {length: 0}
```

Her wurde also das komplette localStorage gelöscht.

Questions? You‘re welcome!

holger.zerbe@digitalcareerinstitute.org