Überblick Funktionen in Javascript

Funktionen

1. dienen zur Mehrfachverwendung von Code
2. können eine (fast) beliebige Anzahl von Parametern bekommen
3. können (müssen aber nicht) einen Rückgabewert returnen
4. Innerhalb der Funktion kann mit `throw` eine Exception geworfen werden, sodaß die Funktion frühzeitig verlassen wird. Dieses frühzeitige Verlassen kann auch durch ein `throw` in einer anderen, ”tieferen” - von der Funktion aufgerufenen - Funktion verursacht werden.
5. Innerhalb der Funktion besteht Zugriff auf den globalen Scope (Varibalen, welche mit `const` oder `let` deklariert wurden)
6. Eine Funktion kann selber Parameter für eine andere Funktion sein. (callbacks, wie z.B. `setTimeout(function, milliSeconds)`)
7. Funktionen *können* einen Namen haben
8. Funktion können auch anonym übergeben werden (lambdas bzw. umgangssprachlich ”Arrow functions”)

9. Zum Returnen von mehreren Werten kann ein Objekt oder Array verwendet werden (s.u.)
10. Generator functions (*) TODO

Anmerkungen

ad 2) Anzahl der Parameter

Für die Sprache selbst ist es egal, ob der Aufrufer die ”richtige” Anzahl an Parametern mitgibt. Beispiel ohne Parameter, aber dennoch werden welche übergeben:

```
function f() {
    for (let i = 0; i < arguments.length; i++) {
        console.log(i, arguments[i]);
    }
}
f(23, "javascript", new Date());
```

```
0 23
1 javascript
2 2023-10-18T20:28:55.244Z
```

Hier werden ”zu wenige” Parameter übergeben, beachte das undefined im output:

```
function f(a, b, c) {
    console.log(a);
    console.log(b);
    console.log(c);
    for (let i = 0; i < arguments.length; i++) {
        console.log(i, arguments[i]);
    }
}
f(23, 3.14159);

23
3.14159
undefined
```

```
0 23
1 3.14159
```

ad 3) Kein Rückgabewert

Gleiches gilt für den Rückgabewert einer Funktion:

```
function f() {}
y = f();
console.log(y);
```

```
undefined
```

ad 6) Eine Funktion als Argument

```
function f(funcPar, actualPar) {
    funcPar("The answer:", actualPar);
}
f(console.log, 42);
```

(`console.log` ist hier der Parameter.)

```
The answer: 42
```

ad 8) Eine Arrow-Funktion als Argument

Code für einen verschwindendes html-Element:

```
const title = document.querySelector("h1");
title.addEventListener("click", (e) => {
    e.target.style.display = "none";
});
```

Hier wird eine anonyme Funktion als Parameter an die `addEventListener()` Methode (Funktion) übergeben.

ad 9) Return-Type: Objekt

Ein Objekt wird returned

```
function f() {
    return { ort: "Wien", temp: 7 };
}
a = f().ort;
console.log(a);
```

```
Wien
```