THM CAMPUS GIESSEN | MNI Mathematik, Naturwissenschaften und Informatik
TECHNISCHE HOCHSCHULE MITTELHESSEN

TypeScript Cheatsheet

Variablen

Syntax	let bezeichner: Typ = Wert
Typen	
Wahrheitswert	let isDone: boolean = false;
Zahl	let myNumber: number = 42;
Buchstabe/Text	let myText: string = "THM";

Allgemeine Operatoren

=	Zuweisung
	x = 42;
// Text	Einzeiliges Kommentar
/* Text */	Mehrzeiliges Kommentar

Arithmetische Operatoren

+	Addition oder Konkatenation
	x = 10 + 5; // 15
	y = "Hallo " + "Welt";
-	Subtraktion
	x = 10 - 5; // 5
*	Multiplikation
	x = 5 * 2; // 10
/	Divison
	x = 10 / 5; // 2
%	Modulo (Rest)
	x = 9 % 5; // 4
++ oder --	In-/dekrementieren
	x++; // x = x + 1

Logische Operatoren

==	Gleichheit
	5 == 5; // true
!=	Ungleichheit
	5 != 5 // false
< und >	Kleiner/größer als
	5 < 6 // true
	5 > 6 // false
<= und >=	Kleinergleich/Größergleich
	5 <= 5 // true
	5 < 5 // false
||	Oder
	false || false // false
	false || true // true
&&	Und
	true && true // true
	false && true // false

Bedingte Anweisungen

if

```
if ( <Bedingung> ) {
        <Anweisung1..n>
}
```

if - else

```
if ( <Bedingung> ) {
        <Anweisung1..n>
} else {
        <Anweisung1..n>
}
```

if - else if

```
if ( <Bedingung> ) {
        <Anweisung1..n>
} else if ( <Bedingung> ) {
        <Anweisung1..n>
}
```

Switch-Case

```
switch ( <Ausdruck> ) {
        case <erg1..n>:
                <Anweisung1..n>
                break;
        default: <Anweisung1..n>
}
```

Schleifen

while

```
while ( <Bedingung> ) {
        <Anweisung1..n>
}
```

do-while

```
do {
        <Anweisung1..n>
} while ( <Bedingung> );
```

for

```
for (<Init>; <Bedingung>; <Letzter Ausdruck>) {
        <Anweisung1..n>
}
```

Arrays

Initialisierung und Zugriff

```
let bezeichner: Typ[] = [ wert1, wertn ];
bezeichner[0] = "THM"; // Zuweisung
myVar = bezeichner[1]; // Indizierter Zugriff
```

Funktionen und Felder

Länge	bezeichner.length
Hinzufügen	bezeichner.push(<Wert>)
Entfernen	bezeichner.pop()

Assoziative Zugriffe

```
let myStringArray: String[] =[];
myStringArray["farbe"] = "blau";
let myString: String = myStringArray["farbe"] //blau
```

Funktionen

Syntax

```
function bez(paraBez1: Typ1): RTyp {
        <Anweisung1..n>
        return varVonRTyp;
}
Aufruf: let ergebnis: RTyp = bez(parameter1);
```