JavaScript Language Features

Mark Zamoyta
SOFTWARE DEVELOPER AND EDUCATOR

@markzamoyta

Introduction

- Constants
- let and var
- Rest Parameters
- Destructuring
- Spread Syntax
- typeof()
- Common Type Conversions
- Controlling Loops

Constants

Constants

```
const carId = 42;
```

Constants

```
const carId = 42;

const bodyStyle;    // error!
```

Constants

```
const carId = 42;

...

carId = 100;     // error!
```

let and var for Variable Declarations

let and var

```
let carId = 42;
```

let and var

```
console.log(carId); // error!
let carId = 42;
```

let and var

```
console.log(carId); // undefined (not an error)
var carId = 42;
```

let and var

```
if (true) {
    var foo = 9;
}
console.log(foo);   // 9
```

let and var

```
if (true) {
    let foo = 9;
}
console.log(foo);  // error!
```

Rest Parameters

Rest Parameters

```
function sendCars(...allCarIds) {
    allCarIds.forEach( id => console.log(id));
}

sendCars(100, 200, 555);

// 100  200  555
```

Rest Parameters

```
function sendCars(day, ...allCarIds) {
    allCarIds.forEach( id => console.log(id));
}

sendCars('Monday', 100, 200, 555);


// 100  200  555
```

Destructuring Arrays

Destructuring Arrays

```
let carIds = [1, 2, 5];
let [car1, car2, car3] = carIds;

console.log(car1, car2, car3);


// 1  2  5
```

Destructuring Arrays

```
let carIds = [1, 2, 5];

let car1, remainingCars;

[car1, ... remainingCars] = carIds;

console.log(car1, remainingCars);

// 1  [ 2,  5 ]
```

Destructuring Arrays

```
let carIds = [1, 2, 5];
let remainingCars;
[, ... remainingCars] = carIds;

console.log(remainingCars);
// [ 2,  5 ]
```

Destructuring Objects

Destructuring Objects

```
let car = { id: 5000, style: 'convertible' };
let { id, style } = car;

console.log(id, style);
// 5000 convertible
```

Destructuring Objects

```
let car = { id: 5000, style: 'convertible' };
let id, style;
{ id, style } = car;  // error!

console.log(id, style);
```

Destructuring Objects

```
let car = { id: 5000, style: 'convertible' };
let id, style;
({ id, style } = car);

console.log(id, style);
// 5000 convertible
```

Spread Syntax

Spread Syntax

```
function startCars(car1, car2, car3) {
    console.log(car1, car2, car3);
}

let carIds = [100, 300, 500];
startCars(...carIds);
// 100  300  500
```

Spread Syntax

```
function startCars(car1, car2, car3) {
    console.log(car1, car2, car3);
}

let carCodes = 'abc';
startCars(...carCodes);
// a  b  c
```

typeof()

typeof()

```
typeof(1);     // number

typeof(true); // boolean

typeof('Hello'); // string

typeof( function(){} ); // function

typeof({}); // object

typeof(null); // object

typeof(undefined); // undefined

typeof(NaN); // number
```

Common Type Conversions

Common Type Conversions

```
// convert to string
foo.toString();

// convert string to integer
Number.parseInt('55');    // 55 as a number

// convert string to number
Number.parseFloat('55.99');     // 55.99 as a number
```

Controlling Loops

Controlling Loops

```
let i=0;
for (; i<12; i++) {
  if (i === 8) {
    break;
  }
}
console.log(i);  // 8
```

Controlling Loops

```
for (let i=0; i<4; i++) {
  if (i === 2) {
    continue;
  }
  console.log(i);
}
// 0  1  3
```

Summary

Constants

- const

let and var

Rest Parameters

- ...theRest

Destructuring

- arrays: [a, b] = arr;
- objects: ({a, b} = obj);

Spread Syntax

- fn(...arr)

Summary

typeof()

Common Type Conversions

- toString()
- Number.parseInt()
- Number.parseFloat()

Controlling Loops

- break
- continue