Appendix: JavaScript Getting Started Guide

What is JavaScript?

JavaScript is a programming language and one of the most popular programming languages in the world.It has a profound impact on all aspects of our lives, usually open the browser to see all the web pages, a lot of App and so on can see it.

JavaScript itself is a very easy-to-use programming language, which is very suitable for beginner programming of rookies without any experience.At the same time, JavaScript is supported by most major browsers, and you can run JavaScript code without deploying any environment and only one browser.

Tutorial recommendation

JavaScript itself is a very easy-to-use programming language, which is very suitable for beginner programming of rookies without any experience.At the same time, JavaScript is supported by most major browsers, and you can run JavaScript code without deploying any environment and only one browser.

- MDN JavaScript Guide
- javascript.info
- javascript.info in Chinese

Getting started

> For more information on how the sample code works, please see here.

Basic grammar

Like most programming languages, JavaScript code is made up of statements.For example, the following code snippet contains one statement per line, with a total of two lines:

```
console.log("Hello World");
const name = "bob";
```

Meanwhile, by default, each statement needs to end with `;`.In theory, we also support two statements on the same line, but this is not recommended:

```
console.log("Hello World");
const name = "bob";
```

Under normal circumstances, in order to achieve a function, many different statements will be written, and different statements also have different functions.It is the corresponding logic on the collocation of these different statements that constitute the code fragments in our daily cognition.The following is an overview of some of the basic writing methods and specific functions of JavaScript.

Data types and Variables

Just like the column types of AITable, different data types are defined in JavaScript.

String

One of the most frequently used data types is any text enclosed in single or double quotes.

```
const name = "bob";
```

Number

Numbers do not need to be enclosed in single or double quotation marks, so the wrapped numbers are not of the Number type.

```
const myString = "1"; // String
const myNumber = 1; // Number
```

Boolean

For Boolean values, there are only two cases: `true` and `false`, again without quotation marks.

```
const myString = "false"; // String
const myFalse = false; // Boolean
```

Array

Array is a little more complex than the above data types, it can contain any data type, and it is very convenient to read and write the data in the array.

```
// Define an Array
const myArray = [1, "test", true, 10];

// Read data in Array
console.log(myArray[0]); // result is 1
console.log(myArray[1]); // result is test
```

Object

Objects are the way JavaScript describes the world.In real life, when we describe a person through language, we usually start with height, age, etc.: his name is Bob, he is 22 years old this year, and his height is 170. Using JavaScript as a programming language can be described in a more structured way, so let's extract the information expressed above and sort it out:

Attributes	Content
Name	Bob
Age	22
Height	170

An object is actually a special expression of the above structured information by JavaScript. Name, age, etc., are usually called keys (which can be understood as a special variable name), while the specific content is the value:

```
// Define an Object
const people = {
  name: "Bob",
  age: 22,
  height: 170,
};

// Read data in Object
console.log(people.name); // result is Bob
console.log(people.age); // result is 22
```

At this point, we can think of `people` as an object with `name`, `age` and other key information.

Variables and Constants

In JavaScript, both Variables and Constants are used to store data, and the former is similar to variables in algebra-the data it stores can be changed, while the latter is not.

```
let myFalse = false; // Define a Variable: myFalse
const myNumber = 1; // Define a Constant: myNumber, value: 1
const myString = "test"; // Define a Constant: myString, value: 'test'
myFalse = true; // update myFalse, the updated value is true
console.log(myFalse);
console.log(myNumber);
console.log(myString);
```

You can see that the above constants `myNumber` and `myString` start with `const`, followed by a specific name and `=`, and finally write the specific value.Variables are defined in a similar way, but they need to start with `let`, for example, the variable `myFalse` above.Variables defined using `let` can be updated, just like the last `myFalse = true` in the above code to change `myFalse` from false to true.

If you want to learn more about `const` and `let`, I suggest searching in the [recommended tutorial](#Tutorial recommendation) above.

Operator

Computing is one of the most common uses of programming, so how to use operators is also a very important part of learning a language.

Assignment

`=`: This one has been mentioned in the chapter Variables and Constants above

Addition, Subtraction, Multiplication and Division

```
+

// Two numbers of type Number will be added
const myNumber1 = 1;
const myNumber2 = 2;
console.log(myNumber1 + myNumber2); // result is 3

// Two String types of data will be concatenated
const firstName = "Bob";
const lastName = "APITable";
console.log(firstName + " " + lastName); // result is Bob APITable

- * /

// Usually only used for Number types
const myNumber1 = 1;
const myNumber2 = 2;

console.log(myNumber1 - myNumber2); // result is -1
console.log(myNumber1 * myNumber2); // the result is 2
console.log(myNumber1 / myNumber2); // the result is 0.5
```

Compare

Here are some commonly used operators for reference

>=: greater than or equal to

>: greater than

<=: less than or equal to

<: less than

==: equal to

!=: not equal to

Conditional statement

The if statement is a very common way of conditional judgment in JavaScript, and is usually used in conjunction with comparison operators:

```
const muNumber1 = 1;
const muNumber2 = 2;

if (muNumber1 == muNumber2) {
  console.log(1);
} else if (muNumber1 > muNumber2) {
  console.log(2);
} else {
  console.log(3);
}
// result is 3
```

In addition, JavaScript also supports switch statement for conditional judgment. If you are interested, you can go to Recommended tutorial search in.

Loop statement

If you want to be able to execute the same code repeatedly when writing code, you can use the for statement to achieve it.

```
const myNumber = 0;
const circleNum = 100; // number of cycles

for (i = 0; i < circleNum; i++) {
  myNumber = myNumber + i;
}

console.log(myNumber); // the result is 4950
```

- i=0 is to initialize the loop variable
- i<circleNum is to judge whether the loop variable is less than the defined number of loops, if so, it will continue to execute
- i++ means i will be incremented by 1 after each loop

In addition, JavaScript can also use for/in and While statements to loop. If you are interested, you can go to the Recommended tutorial search in.

Previous
« Generate latitude and longitude from IP address

Next
Script API Reference »

Developer Center

Development guide

API Reference

Social

Twitter

More

Homepage

Help Center

GitHub