Front-end

JavaScript

JS. Scope

- Область видимости

JS. Scope

Область видимости переменных или просто “Область видимости” (англ. variable scope или просто scope) — важная концепция, определяющая доступность переменных.

Данная концепция лежит в основе замыканий, разделяя переменные на глобальные и локальные.

Все переменные и константы в JavaScript имеют определенную область видимости, в пределах которой они могут действовать.

В JavaScript есть три области видимости:

- глобальная
- область видимости функции (функциональная область видимости)
- блочная

Область видимости переменной – это участок исходного кода программы, в котором переменные и функции видны и их можно использовать.

JS. Scope

Область программы, в пределах которой установлена связь между некоторой переменной и её идентификатором (именем), по которому можно получить значение этой переменной. За пределами области видимости тот же самый идентификатор может быть связан с другой переменной, либо быть свободным (вообще не связанным ни с какой из переменных).

JS. Scope

Создаются области видимости во время выполнения программы. Самая первая область, которая создаётся и которая включает в себя все остальные называется глобальной.

Именно в этой области определены такие переменные как window в веб-браузере и global в Node.js.

Вы также можете определять переменные в этой области. Для этого достаточно просто объявить переменные вне блока, функции и модуля. В этом случае они будут находиться в глобальной области видимости

JS. Scope

Переменные объявленные в глобальной области видимости называются глобальными переменными. Такие переменные могут быть доступны в любой точке программы.

Кроме глобальной области видимости в JavaScript имеются ещё локальные. Они, создаются, когда интерпретатор, например, выполняет код блочной конструкции.

Причем такая локальная область видимости называется **областью видимости блока.**

```
// глобальная переменная
let a = 5;
{
  // локальная переменная
  let b = 17;
}
```

JS. Scope

Переменные, объявленные внутри блока с помощью **let и const** имеют область видимости ограниченную этим блоком. Т.е. они привязаны к нему и будет действовать только в его рамках. Переменные, объявленные в локальной области видимости называются локальными.

```
if (true) {
    const message = 'Hello'
}
console.log(message) // ReferenceError: message is not defined
```

```
if (true) {
  // локальная переменная
  let b = 17;
  // выведем значение переменной b в консоль
  console.log(b); // 17
}
console.log(b); // Uncaught ReferenceError: b is not defined
```

Под блоком в JavaScript понимается любой код, который расположен в фигурных скобках { ... }. Блоки используются в конструкциях if, for, while и т.д. Даже тело функции является блоком, т.к. находится между фигурными скобками.

Кроме этого локальные области видимости также создаются вызовами функций и модулями. Они соответственно называются областью видимости функции и областью видимости модуля.

Блок if создал область видимости для переменной. И она доступна только внутри этой области.

JS. Scope

var не имеет блочной области видимости

```
if (true) {
    // область видимости блока if
    var count = 0
    console.log(count) // 0
}
console.log(count) // 0
```

Как мы видели в предыдущих примерах, блок кода создает область видимости для переменных, объявленных с помощью ключевых слов const и let. Однако это не работает для переменных, объявленных с помощью ключевого слова var

JS. Scope

В примере ниже в глобальной области видимости объявляется функция salute с помощью ключевого слова function. Затем эта функция вызывается два раза.

Область видимости функции создаётся для каждого вызова функции. Даже, когда мы вызываем одну и ту же функцию. При этом для каждого вызова создаётся своя отдельная область видимости.

В этом примере будут созданы две локальные области видимости уровня функции.

```
// глобальная область видимости
function salute(welcomeText) {
  console.log(welcomeText);
}

salute('Привет'); // вызов функции salute
salute('Здравствуйте'); // вызов функции salute
```

JS. Scope

Поиск переменной всегда начинается с текущей области видимости.

JS. Scope

Область видимости функции

Функции в JavaScript создают область видимости для всех переменных, независимо от того, с помощью какого ключевого слова они объявлены (var, const или let).

```
function run() {
    // область видимости функции
    var message = 'Беги, Форрест, беги!';
    console.log(message);
}
run(); // 'Беги, Форрест, беги!'
console.log(message); // ReferenceError
```

Функция run() создает область видимости. Переменная message доступна внутри функции, но недоступна снаружи.

JS. Scope

Области видимости могут быть вложенными

Интересной особенностью областей видимости является то, что они могут быть вложены одна в другую.

```
function run() {
    // область видимости функции
    const message = 'Беги, Форрест, беги!';

    if (true) {
        // область видимости блока if
        const friend = 'Бубба';
        console.log(message); // 'Беги, Форрест, беги!'
    }

    console.log(friend); // ReferenceError
}
run();
```

Функция run() создает область видимости, а внутри нее блок if создает еще одну область

JS. Scope

Область видимости, находящаяся внутри другой области, называется внутренней областью видимости. В приведенном примере — это область видимости блока if.

```
function run() {
    // область видимости функции
    const message = 'Беги, Форрест, беги!';

    if (true) {
        // область видимости блока if
        const friend = 'Бубба';
        console.log(message); // 'Беги, Форрест, беги!'
    }

    console.log(friend); // ReferenceError
}
run();
```

Область видимости, содержащая другую область, называется внешней областью видимости. В приведенном примере — это область видимости фукнции run().

JS. Scope

Лексическая область видимости

Ещё один момент, о котором стоит упомянуть - это лексическая область. Лексическая область означает, что дочерняя область имеет доступ к переменным, определенным в родительской области. Дочерние функции лексически связаны с контекстом исполнения их родителей.

JS. Scope

Лексическая область видимости

```
function func1() {
    var num1 = 5;
    const num2 = 10;
    let num3 = 23;
    function func2() {
        console.log(num1); // => 5
        console.log(num2); // => 10
        console.log(num3); // => 23
    }
    func2();
}

func1();
```

JS. Scope

Лексическая область видимости

Лексическая область видимости - это набор правил о том, как и где движок JavaScript может найти переменную. Ключевой характеристикой лексического контекста является то, что он определяется во время написания кода.

Лексическая область - определяется во время написания кода.

Лексическая область заботится о том, где была объявлена функция