

JS: Functions

НАШИ ПРАВИЛА



Включенная камера



Вопросы по поднятой руке



Не перебиваем друг друга



Все вопросы, не связанные с тематикой курса (орг-вопросы и т. д.), должны быть направлены куратору



Подготовьте свое рабочее окружение для возможной демонстрации экрана (закройте лишние соцсети и прочие приложения)

Повторим;)

Какие особенности массива в JS?

Что делают методы push, shift, unshift, pop?

Как вызвать определенный элемент массива

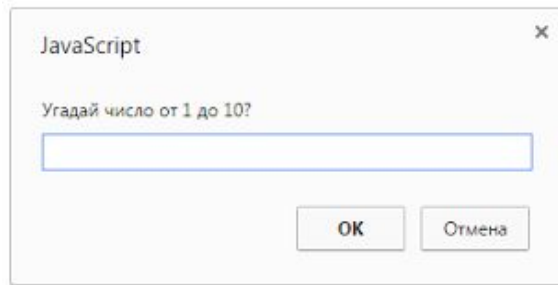
ЦЕЛЬ

Изучить функции в JS. Область видимости переменных

ПЛАН ЗАНЯТИЯ

- Functions: introduction

Встроенные функции. Взаимодействие с пользователем



A screenshot of a JavaScript alert dialog box. The title bar at the top left says "JavaScript" and there is a close button (X) at the top right. The main text inside the dialog asks "Угадай число от 1 до 10?". Below the text is a single-line text input field. At the bottom of the dialog are two buttons: "ОК" on the left and "Отмена" on the right.

console.log()

Чтобы вывести что-то на консоль из нашего кода, существует функция `console.log`.

Обычный пользователь сайта не увидит такой вывод, так как он в консоли. Чтобы увидеть его, либо откройте консольную панель инструментов разработчика, либо нажмите Esc, находясь в другой панели: это откроет консоль внизу.

```
let message = "Привет, мир!";  
console.log(message);
```

alert()

- alert используется для вывода всплывающего диалогового окна с сообщением.
- Принимает один параметр - текст сообщения.

Она показывает сообщение и ждёт, пока пользователь нажмёт кнопку «OK».

```
1 alert("Hello");
```


prompt()

- `prompt` используется для вывода всплывающего диалогового окна с полем для ввода текста.
- Принимает два параметра: текст сообщения и необязательное значение по умолчанию для поля ввода.

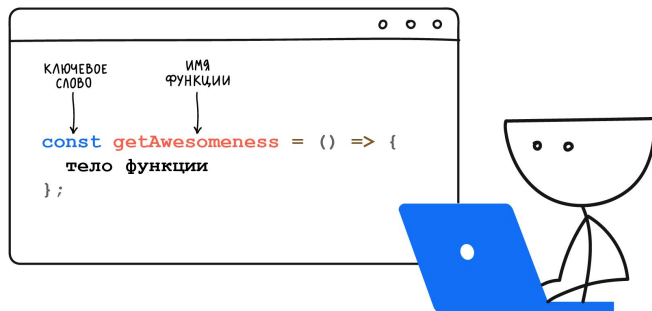
```
1 let age = prompt('Сколько тебе лет?', 100);  
2
```

confirm()

Функция `confirm` отображает модальное окно с текстом вопроса `question` и двумя кнопками: ОК и Отмена.

```
1 let isBoss = confirm("Ты здесь главный?");  
2  
3 alert( isBoss ); // true, если нажата ОК
```

Функции. Основы



Зачастую нам надо повторять одно и то же действие во многих частях программы.

Например, необходимо красиво вывести сообщение при приветствии посетителя, при выходе посетителя с сайта, ещё где-нибудь.

Чтобы не повторять один и тот же код во многих местах, придуманы функции. Функции являются основными «**строительными блоками**» программы.

Объявление функции

Для создания функций мы можем использовать объявление функции.

```
1 function showMessage() {  
2   alert( 'Всем привет!' );  
3 }
```

Вначале идёт ключевое слово `function`, после него имя функции, затем список параметров в круглых скобках через запятую - такое объявление функции называется **Function Declaration**

```
1 function имя(параметры) {  
2   ...тело...  
3 }
```

Вызов функции

Наша новая функция может быть вызвана по своему имени: `showMessage()`.

```
1 function showMessage() {  
2     alert( 'Всем привет!' );  
3 }  
4  
5 showMessage();  
6 showMessage();
```

Локальные переменные

Переменные, объявленные внутри функции, видны только внутри этой функции.

```
1 function showMessage() {  
2     let message = "Привет, я JavaScript!"; // локальная переменная  
3  
4     alert( message );  
5 }  
6  
7 showMessage(); // Привет, я JavaScript!  
8  
9 alert( message ); // <-- будет ошибка, т.к. переменная видна только внутри функции
```

Внешние переменные

У функции есть доступ к внешним переменным, например:

```
1 let userName = 'Вася';  
2  
3 function showMessage() {  
4   let message = 'Привет, ' + userName;  
5   alert(message);  
6 }  
7  
8 showMessage(); // Привет, Вася
```

Если одноимённая переменная объявляется внутри функции, тогда она перекрывает внешнюю.

Параметры

Мы можем передать внутрь функции любую информацию, используя параметры. В нижеприведённом примере функции передаются два параметра: `from` и `text`.

```
1 function showMessage(from, text) { // параметры: from, text
2   alert(from + ': ' + text);
3 }
4
5 showMessage('Аня', 'Привет!'); // Аня: Привет! (*)
6 showMessage('Аня', "Как дела?"); // Аня: Как дела? (**)
```

Возврат значения

Функция может вернуть результат, который будет передан в вызвавший её код.

Простейшим примером может служить функция сложения двух чисел:

```
1 function sum(a, b) {  
2   return a + b;  
3 }  
4  
5 let result = sum(1, 2);  
6 alert( result ); // 3
```

Выбор имени функции

Функция – это действие. Поэтому имя функции обычно является глаголом. Оно должно быть кратким, точным и описывать действие функции, чтобы программист, который будет читать код, получил верное представление о том, что делает функция.

Например, функции, начинающиеся с "show" обычно что-то показывают.

Функции, начинающиеся с...

- "**get...**" – возвращают значение,

- "**calc...**" – что-то вычисляют,

- "**create...**" – что-то создают,

- "**check...**" – что-то проверяют и возвращают логическое значение, и т.д.

Выбор имени функции

Примеры имен

```
1 showMessage(..)    // показывает сообщение
2 getAge(..)         // возвращает возраст (получая его каким-то образом)
3 calcSum(..)        // вычисляет сумму и возвращает результат
4 createForm(..)     // создаёт форму (и обычно возвращает её)
5 checkPermission(..) // проверяет доступ, возвращая true/false
```

Параметры по умолчанию

Если при вызове функции аргумент не был указан, то его значением становится undefined.

Если мы хотим задать параметру text значение по умолчанию, мы должны указать его после =:

```
1 function showMessage(from, text = "текст не добавлен") {  
2     alert( from + ": " + text );  
3 }  
4  
5 showMessage("Аня"); // Аня: текст не добавлен
```

Теперь, если параметр text не указан, его значением будет "текст не добавлен"



Ваша новая IT-профессия – Ваш новый уровень жизни

Программирование с нуля в
немецкой школе AIT TR GmbH