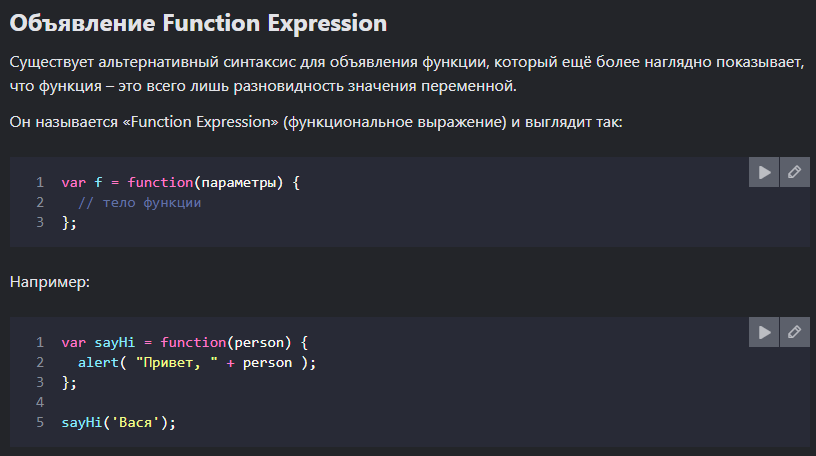
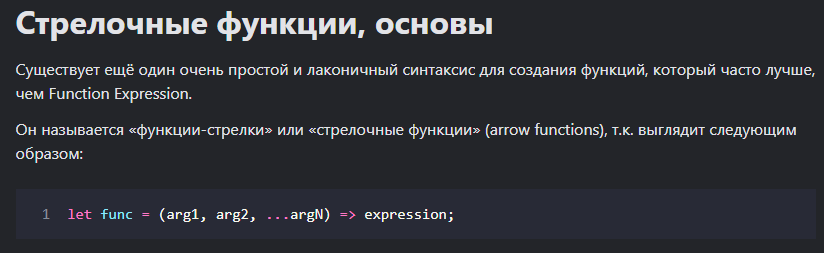
Способы создания функции ОБЬЕКТЫ

JavaScript позволяет создавать функцию различными способами:

* **Function Declaration - Объявление функции**
* **Function Expression - Функциональное выражение**
* 
* **Arrow Function - Функция стрелки**
* 

Анонимная функция

Анонимной в JavaScript называют функцию, с которой не связано никакое имя или, другими словами, у такой функции нет имени. Также можно сказать, что, если после ключевого слова function или перед знаком стрелочной функции => не стоит имя - функция анонимная.

Контект функции

Контекст выполнения

Контекст выполнения (execution context) — это, если говорить упрощённо, концепция, описывающая окружение, в котором производится выполнение кода на JavaScript. Код всегда выполняется внутри некоего контекста.

▍Типы контекстов выполнения

В JavaScript существует три типа контекстов выполнения:

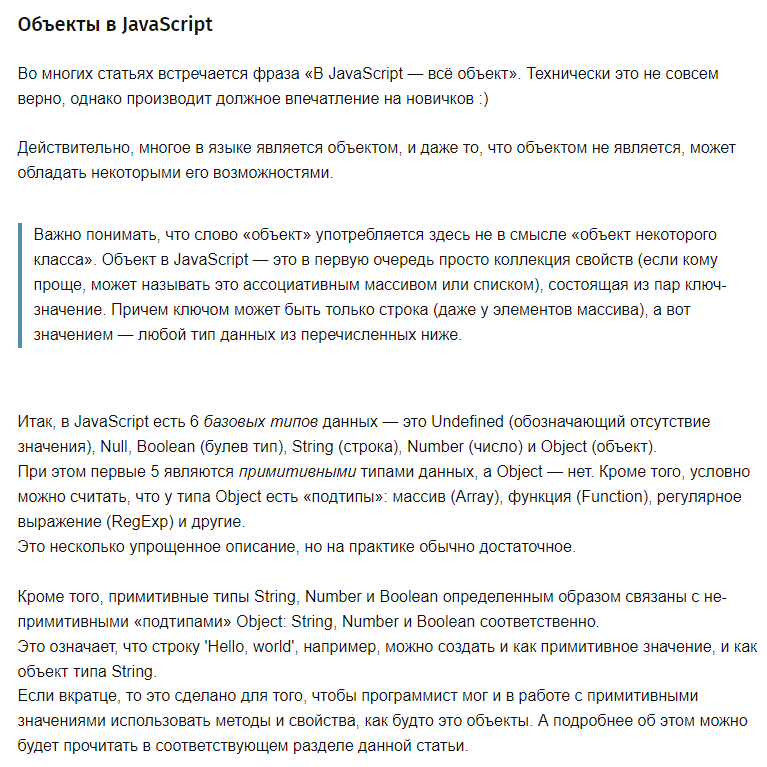
* Глобальный контекст выполнения. Это базовый, используемый по умолчанию контекст выполнения. Если некий код находится не внутри какой-нибудь функции, значит этот код принадлежит глобальному контексту. Глобальный контекст характеризуется наличием глобального объекта, которым, в случае с браузером, является объект window, и тем, что ключевое слово this указывает на этот глобальный объект. В программе может быть лишь один глобальный контекст.
* Контекст выполнения функции. Каждый раз, когда вызывается функция, для неё создаётся новый контекст. Каждая функция имеет собственный контекст выполнения. В программе может одновременно присутствовать множество контекстов выполнения функций. При создании нового контекста выполнения функции он проходит через определённую последовательность шагов, о которой мы поговорим ниже.
* Контекст выполнения функции eval. Код, выполняемый внутри функции eval, также имеет собственный контекст выполнения. Однако функцией eval пользуются очень редко, поэтому здесь мы об этом контексте выполнения говорить не будем. Аргумент функции **eval**() - строка. **eval**() исполняет содержащееся в строке выражение, один или несколько операторов JavaScript.

Возврат значений



Объекты, характеристики, типы обьектов

|  |  |
| --- | --- |
| **Свойство или метод** | **Что делает** |
| length | Число объявленных параметров функции. |
| name | Строка с именем функции (ES2015). |
| prototype | Прототип. Используется при вызове функции как конструктора. Будет прототипом создаваемого объекта. |
| call() и apply() | Методы для косвенного вызова функции. |
| bind() | Создаёт новую функцию, которая при вызове устанавливает в качестве this предоставленное значение. |
| toString() | Метод возвращает исходный код функции в виде строки. |



Операторы цикла (более 3, которые написаны в мануале к лабе)

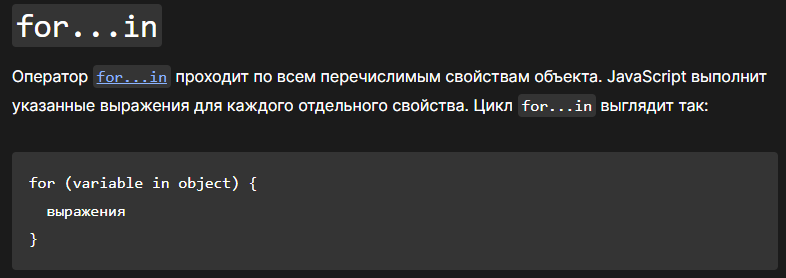
В JavaScript(стандарт ES6) существует 5 типов циклов:

- do...while

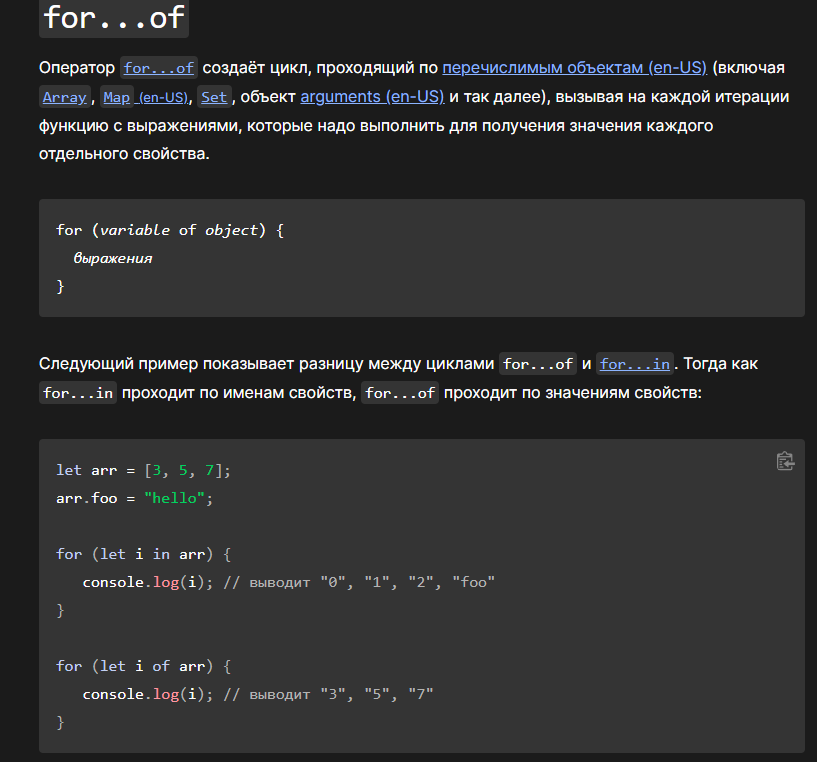
- while

- for

- for..in



- for..of



В JS функции могут вызываться четырьмя способами:

• как функции

• как методы

• как конструкторы

• косвенно, с помощью методов call () и apply ()