

Функциональный javascript

Урок 1





Кадочников Алексей

Frontend-разработчик

- ☀ Веб-разработчик со стажем более 9 лет
- ☀ Преподаватель GeekBrains с 2015 года
- ☀ Автор курсов по Frontend на портале Geekbrains
- ☀ Работал в таких компаниях как VK и Wizard-C







План курса





Что будет на уроке сегодня

-  Spread, rest operator
-  Чистые функции, иммутабельность
-  Замыкания
-  Рекурсия



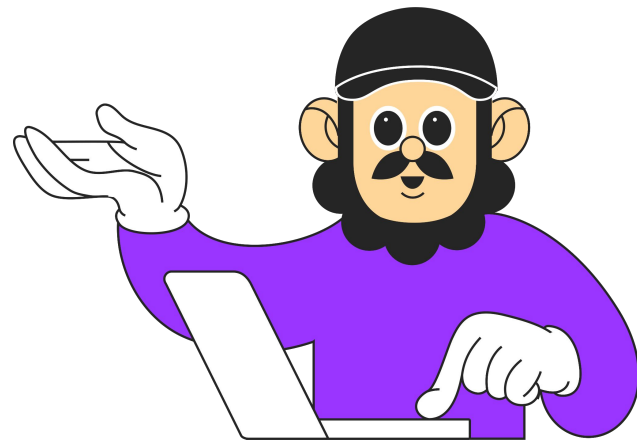
Spread, rest operator





Spread, rest operator

Со стандартом ES2015 нам стали доступны очень полезные инструменты для работы с массивами - это spread и rest операторы, а также деструктуризация.





Spread operator

Spread (от английского расширить) - оператор расширения, или по другому распространения данных из массива в атомарные элементы. Т.е. мы можем взять массив, и вытащить все его элементы как отдельные переменные



Rest operator

Rest оператор (от английского остальные, оставшиеся) - позволяет собрать оставшиеся аргументы функции в массив. Звучит немного странно, но этот оператор позволяет не перечислять все аргументы функции, как отдельные переменные, а получить их скопом в один массив.





Чистые функции,
иммутабельность





Чистые функции

Чистые функции - это такие функции которые при вызове с одними и теми же параметрами всегда возвращают одинаковое значение, при этом такие функции оперируют данными только из полученных аргументов и никак не взаимодействуют с глобальными для них переменными.



Замыкания





Замыкания

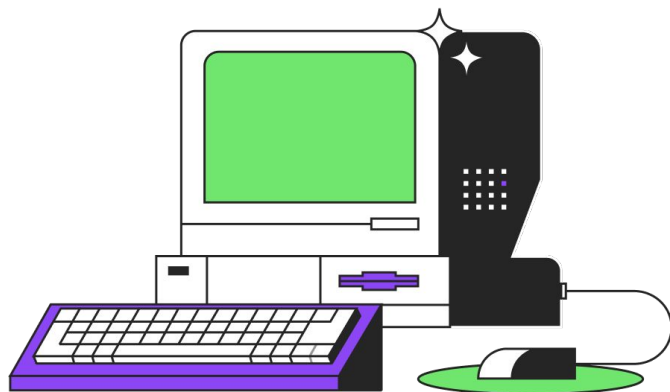
Замыкание — это термин для механизма сохранения данных. Мы замыкаем данные внутри функции таким образом, чтобы к этим данным можно было обратиться и изменить их внутри этой функции, но при этом они были недоступны снаружи.





Основной подход к созданию замыканий:

- создать функцию;
- внутри неё объявить переменные, которые мы хотим в ней замкнуть: спрятать, сохранить и использовать в дальнейшем;
- вернуть из неё другую функцию, которая уже выполняет какое-то конкретное действие и может использовать замкнутые (спрятанные) данные.





Недостатки замыканий

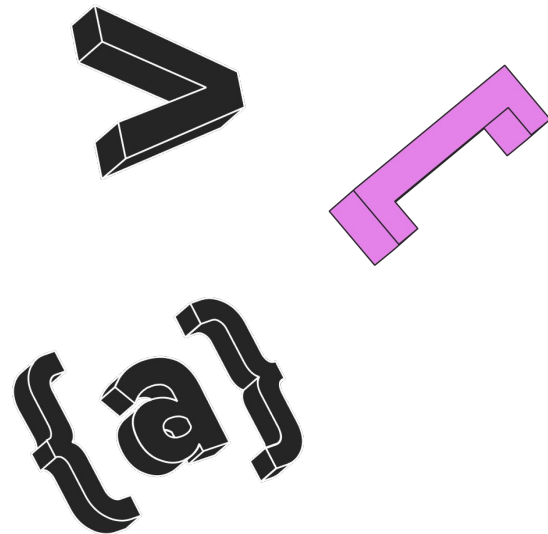
Так как все функции — это замыкания, то при вызове функции создаётся лексическое окружение, которое занимает место в памяти компьютера. В коде часто бывают моменты, когда какая-то функция начинает вызываться чрезмерно много раз. Чаще всего это происходит при работе с рендерингом элементов и их обновлении.

Для функции создаётся чрезмерно много лексических окружений, которые начинают быстро съедать память, — это называется утечкой памяти во время исполнения программы.



Лексический контекст

Замыкания работают благодаря такому механизму, как лексическое окружение. Именно лексическое окружение (или лексический контекст) позволяет хранить все эти замкнутые данные и обращаться к ним при вызове функции, а также позволяет функции иметь доступ к внешним данным.





Лексический контекст

Лексический контекст или лексическое окружение — это механизм в JavaScript, который позволяет функции во время её вызова получать доступ к переменным, константам и всему, что ей нужно. Каждый раз при вызове функции создаётся что-то вроде объекта словаря, который записывает все значения переменных и констант внутри функции, а также тех переменных и констант вне функции, к которым она обращается.



Рекурсия





Рекурсия

Рекурсия — подход к вычислению, который позволяет сделать сложный расчёт итеративным.









Рекурсия


Рекурсия — определение, описание, изображение какого-либо объекта или процесса внутри самого этого объекта или процесса, то есть ситуация, когда объект является частью самого себя. Термин «рекурсия» используется в различных специальных областях знаний — от лингвистики до логики, но наиболее широкое применение находит в математике и информатике





Итоги урока

-  Рассмотрели spread, rest operator
-  Познакомились с чистыми функциями, иммутабельностью
-  Узнали что такое замыкания
-  Узнали про рекурсию

Спасибо 
за внимание

