**Как работает JS: обзор движка, механизмов времени выполнения, стека вызовов**

JavaScript — однопоточный язык, он использует очередь функций обратного вызова. Это означает, что у него один стек вызовов. Таким образом, в некий момент времени он может выполнять лишь какую-то одну задачу.

Движок V8 состоит из двух основных компонентов:

* Куча (Memory Heap) — то место, где происходит выделение памяти.
* Стек вызовов (Call Stack) — то место, куда в процессе выполнения кода попадают так называемые стековые кадры.

Помимо движка есть и другие компоненты, к примеру Web API, который предоставляет нам браузер — средства для работы с DOM, инструменты для выполнения AJAX-запросов, нечто вроде функции setTimeout, и многое другое.

Стек вызовов — это структура данных, которая, говоря упрощённо, записывает сведения о месте в программе, где мы находимся. Если мы переходим в функцию, мы помещаем запись о ней в верхнюю часть стека. Когда мы из функции возвращаемся, мы вытаскиваем из стека самый верхний элемент и оказываемся там, откуда вызывали эту функцию. Это — всё, что умеет стек.

Если будет достигнут максимальный размер стека, возникнет так называемое переполнение стека. Произойти такое может довольно просто, например, при необдуманном использовании рекурсии.

Модель выполнения кода в однопоточном режиме облегчает жизнь разработчика. Ему не нужно принимать во внимание сложные схемы взаимодействия программных механизмов, вроде возможности взаимной блокировки потоков, которые возникают в многопоточных окружениях.  
  
Однако, и у исполнения кода в однопоточном режиме тоже есть определённые ограничения. Учитывая то, что у JavaScript имеется один стек вызовов, поговорим о том, что происходит, когда программа «тормозит».

Каждая запись в стеке вызовов называется **стековым кадром**.  
  
На механизме анализа стековых кадров основана информация о стеке вызовов, трассировка стека, выдаваемая при возникновении исключения. Трассировка стека представляет собой состояние стека в момент исключения.

 Если браузер начинает заниматься обработкой тяжёлых задач, он может на достаточно долгое время перестать реагировать на какие-либо воздействия. Большинство браузеров в подобной ситуации выдают ошибку, спрашивая пользователя о том, хочет ли он завершить выполнение сценария и закрыть страницу. Решение этой проблемы заключается в использовании асинхронных функций обратного вызова.

**Браузер != Браузерный движок**

Браузер объединяет периферию и предоставляет функционал позволяющий манипулировать движком, его поведением, предоставляет дополнительные сервисы.

## **Кто создает браузеры и движки**

Разработкой собственного браузера занимается множество компаний: Google, Mozilla, Apple, Microsoft, Opera Software, Яндекс, Ростелеком, Vivaldi Technologies, **много их**…

Каждый добавляет какие-то свои "фишки", сервисы в создаваемый браузер. К примеру, Яндекс интегрирует свои сервисы, поиск.

С браузерными движками дела обстоят кардинально иначе. Вот список движков, их разработчиков и обладателей:

* Blink: Google
* Gecko: Mozilla
* WebKit: Apple
* EdgeHTML: Microsoft

Это основные обладатели и разработчики движков. В разработке им помогают сторонние компании на определенных условиях.

Например, рассмотрим самый популярный проект от компании Google — Chromium. Данный проект содержит в себе движок Blink. Создавать данный проект Google помогают такие компании как Intel, Facebook, IBM, LG Electronics, NVIDIA, Yandex. Полный список можно посмотреть [тут](https://chromium.googlesource.com/chromium/src.git/+/master/AUTHORS).  
Условия на которых компании помогают создавать Blink/Chromium описаны в разделе [Legal stuff](https://www.chromium.org/developers/contributing-code" \l "TOC-Legal-stuff) на сайте проекта Chromium. Если кратко, то всё что вы создаёте принадлежит (не эксклюзивно) Google.

## **Лицензии браузерных движков**

Любой открытый код содержит в себе лицензию с описанием условий по использованию данного кода, и браузерные движки не исключение.

[**Blink**](https://chromium.googlesource.com/chromium/src.git/+/master/third_party/WebKit)

Открытый исходный код, ответвление движка WebKit. Как и в WebKit заявлено четыре лицензии:

* [The 3-Clause BSD License](https://opensource.org/licenses/BSD-3-Clause)
* [The 2-Clause BSD License](https://opensource.org/licenses/bsd-license.php)
* [GNU Lesser General Public License Version 2, June 1991](https://www.gnu.org/licenses/lgpl-2.0.html)
* [GNU Lesser General Public License Version 2.1, February 1999](https://www.gnu.org/licenses/lgpl-2.1.html)

Самый популярный браузерный движок. Он входит в состав проекта Chromium. Именно на основе Chromium создают собственные браузеры.

[**WebKit**](https://webkit.org/getting-the-code/)

Открытый исходный код. Заявлено четыре лицензии:

* [The 3-Clause BSD License](https://opensource.org/licenses/BSD-3-Clause)
* [The 2-Clause BSD License](https://opensource.org/licenses/bsd-license.php)
* [GNU Lesser General Public License Version 2, June 1991](https://www.gnu.org/licenses/lgpl-2.0.html)
* [GNU Lesser General Public License Version 2.1, February 1999](https://www.gnu.org/licenses/lgpl-2.1.html)

Развивается компанией Apple и используется в собственном браузере компании — Safari. Ранее многие компании использовали в своих разработках WebKit, но после покинули проект и переключились на Blink от Google.

[**EdgeHTML**](https://developer.microsoft.com/en-us/microsoft-edge/)

Закрытый исходный код. Проприетарная лицензия.

## **Риски**

Использование стороннего движка порождает очевидные риски:

1. Закрытие исходного кода
2. Выход ключевых компаний из разработки
3. Смена лицензий на код

## **Адаптация движков**

Существует проблема адаптации стороннего движка под нужды собственного браузера. На адаптацию тратится не мало ресурсов. При этом нужно понимать, что браузерные движки часто обновляются, что создает систематические проблемы при интеграции в собственный браузер.

**Собственный движок**

Прежде всего стоит ответить на главный вопрос — зачем создавать свой браузерный движок? А точнее, кому не надо создавать свой движок?

**Свой браузерный движок не стоит создавать если:**

Нет задачи/стремления захватить рынок.

* Браузер нужен лишь номинально. Сейчас это модно, делают даже школьники.
* Создаваемый движок не будет отличаться от существующих: скорость, надёжность, адаптивность.
* Идёт освоение бюджета.

**Стоит ли заниматься разработкой движка для "заработка", для создания стартапа?**

Данный вопрос скорее к сфере его применения. Мне видится несколько направлений:

Собственно, создание полноценного браузера. Сотрудничество с поисковыми системами, различными сервисами. Создание особых версий для государственных структур.

1. Рынок IoT (интернет-вещей), а так же адаптация для телевизоров, приставок, портативных устройств.
2. Сервисы на отдельных компонентах браузера. Например, умный анализ контента сайта на риски: запрещенные комментарии, посты от пользователей и так далее. Технический анализ сайтов.
3. Продажа движка. Самое простое, написать под компанию.