## Chrome Development tools[[1]](#footnote-0)

### Цель работы

Познакомиться с инструментами веб-браузера Chrome для отладки и мониторинга клиентской части веб-приложений.

### Задания для выполнения

Запустите браузер Google Chrome. Откройте в браузере любую веб-страницу. Лучше всего, если это будет довольно сложная страница, которую разработали вы сами. Откройте инструменты веб-браузера для разработчиков (клавиша F12). Познакомьтесь с основными возможностями, прочитав методические указания и выполнив действия, описанные в них. Поэкспериментируйте. Прочитайте дополнительную информацию по теме. Можно ли с помощью этого инструмента сделать то, чего вам всегда не хватало при разработке веб-приложений?

### Методические указания

При разработки любого типа программ остро стоит вопрос об отладке программы. Начиная с некоторого (довольно невысокого) уровня сложности кода, весьма неудобно вести разработку не имея возможности понимать, что происходит в процессе работы программ.

Это вдвойне верно для веб-приложений, которые по своей сути состоят из множества компонент: дерево объектов веб-страницы, таблицы стилей, скрипты клиентского кода на JavaScript. Традиционные методы и приемы отладки не работают в таком случае, так как средой выполнения веб-приложения является браузер.

К счастью, все современные браузеры предоставляют широкий набор визуальных графических инструментов, облегчающих разработку, мониторинг и отладку веб-приложений. Умение пользоваться этими инструментами сильно помогает в написании, отладке, оптимизации веб-приложений. Здесь мы рассмотрим работу с подобными инструментами на примере одного из самых распространенных браузеров - Google Chrome.

#### Запуск вкладки

Chrome Development Tools - набор инструментов для отладки веб-приложений созданы как набор специальных вкладок в служебной панели браузера, которая запускается по клавише F12. Той же клавишей можно в любой момент скрыть эту панель.

При запуске панель взаимодействует с открытой в текущей вкладке веб-страницей, и может показывать разнообразную информацию о странице и клиентском коде, обеспечивать взаимодействие с элементами страницы.

При запуске панели вы увидите следующие вкладки:

Elements - обзор дерева элементов веб-страницы;

Console - консоль Javascript;

Sources - перечень всех исходных файлов, участвующих в рендеринге страницы;

Network - мониторинг времени загрузки страницы;

Performance - замеры производительности работы страницы;

Memory - мониторинг использования памяти и профилирование приложения;

Application - общие сведения о приложении, в том числе данные из локальных хранилищ и cookies;

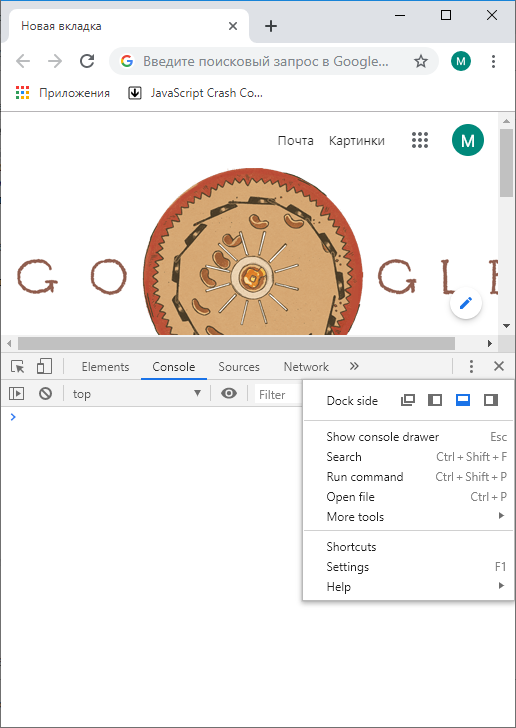
Security - сведения о сертификатах безопасности и шифровании соединения;

Audits - возможность запуска аудитов (комплексных проверок) приложения.

Кроме того, некоторые расширения браузера могут дополнять функциональность этого базового набора. Например, если вы разрабатываете с использованием React, то будет полезно поставить расширение react development tools, которое добавляет две вкладки, специально предназначенные для просмотра информации о компонентах и их состоянии.

#### Изменение положения панели

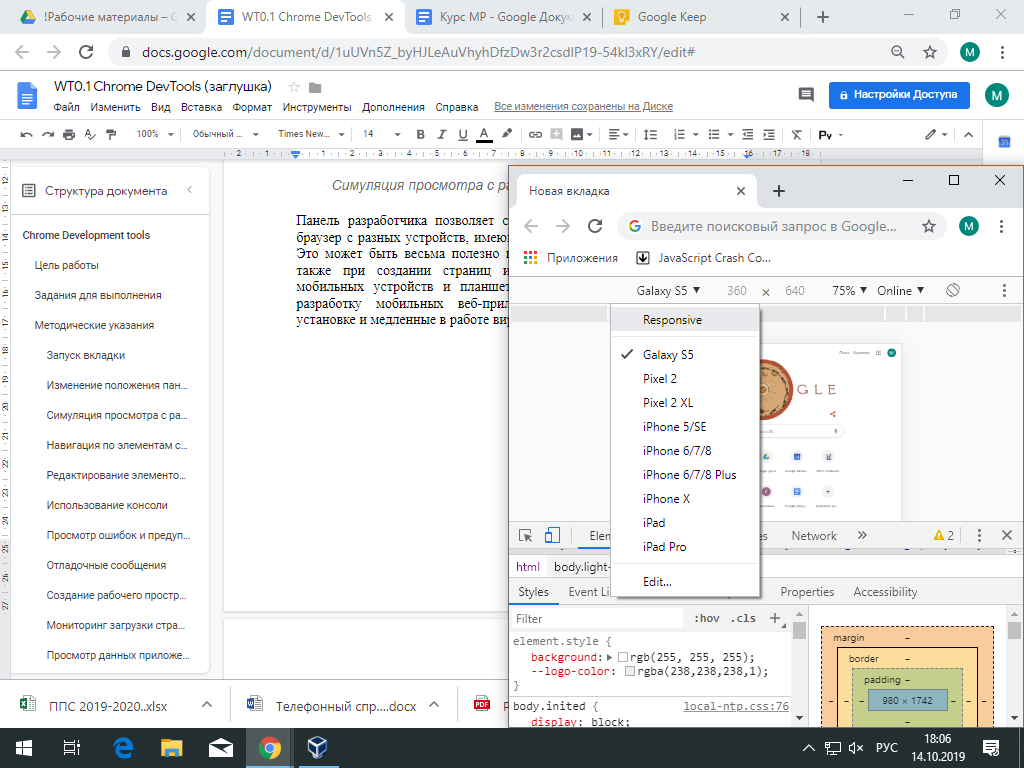
Вы можете использовать консоль как в модальном, так и в независимом режиме. Переключиться между ними, а также пристыковать панель к разным границам окна браузера можно из основного меню панели в правом верхнем углу, рядом с кнопкой закрытия панели:



#### Симуляция просмотра с разных устройств

Панель разработчика позволяет симулировать просмотр загруженной в браузер с разных устройств, имеющих различные характеристики экрана. Это может быть весьма полезно при разработке адаптивного дизайна, а также при создании страниц и приложений, предназначенных для мобильных устройств и планшетов. Это позволяет вести несложную разработку мобильных веб-приложений, не используя сложные в установке и медленные в работе виртуальные мобильные устройства.

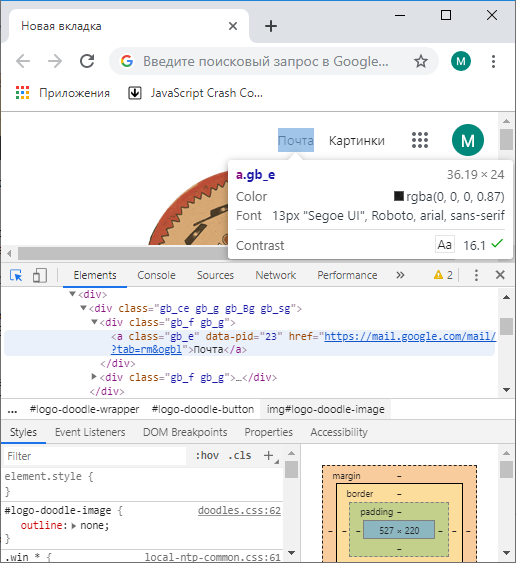
Вызов режима адаптивного просмотра производится кнопкой в левом верхнем углу панели, рядом с кнопкой выделения элементов страницы, о которой мы поговорим позже:



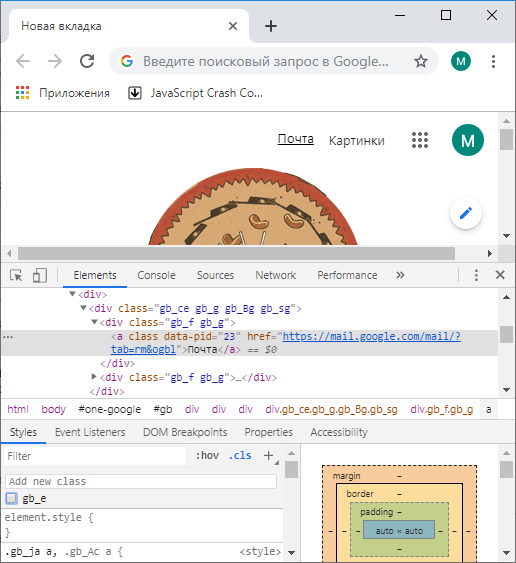
Браузер уже имеет несколько стандартных пресетов, условно соответствующих эффективному размеру экрана разных моделей смартфонов и планшетов. Режим “responsive” позволяет гибко изменять эффективное разрешение экрана по желанию пользователя, таким образом не ограничивая его в стандартных настройках.

#### Навигация по элементам страницы

Вкладка “Elements” позволяет просмотреть код HTML, отображаемый браузером в текущей вкладке. Вы можете в интерактивном режиме выделять элементы в коде и они будут подсвечены на самой странице. Также, используя кнопку “select an element”, которая расположена в левом верхнем углу панели вы можете выделять элементы на самой странице и видеть, какому коду HTML они соответствуют:



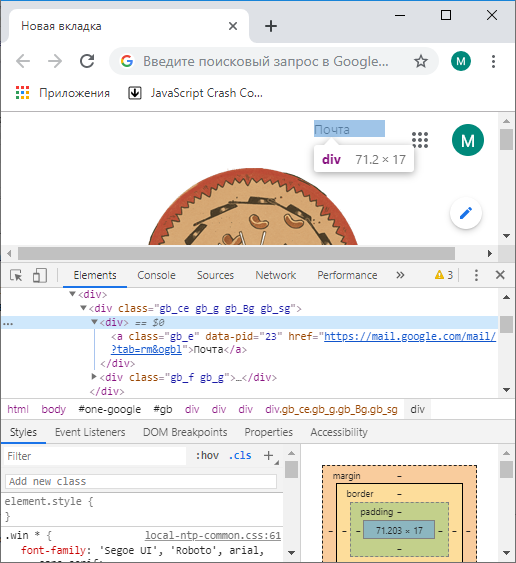
Кроме того, ниже самого кода элементов вы можете видеть его эффективные стили - то есть стили, вычисленные с учетом всех возможных наследований. Здесь же вы можете увидеть все классы и псевдоклассы, присвоенные элементу, а также динамически их включать или отключать. Также можно отключить любое конкретное правило CSS, относящееся к данному элементу. Все изменения в реальном времени отображаются на странице:



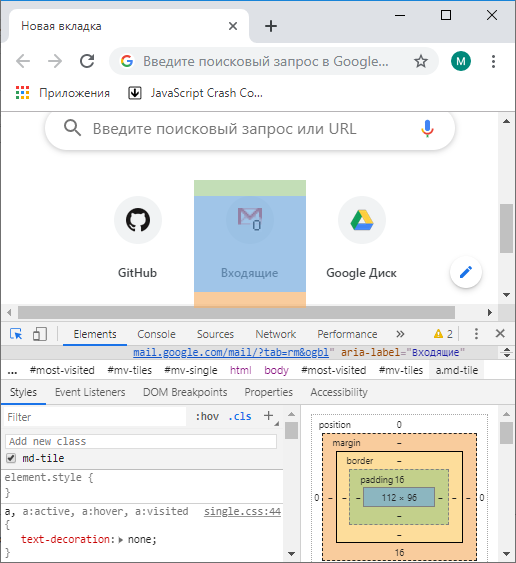
#### Редактирование элементов и стилей

Весь код, который вы видите на этой вкладке можно редактировать. Вы можете изменить значение атрибута тега, удалить или добавить любой атрибут. То же самое применимо и к стилям.

Например, мы можем удалить атрибут класса у элемента, отвечающего за ссылку. После этого, форматирование на странице “поползет”. Это может быть полезно при написании таблиц стилей, так как позволяет вживую увидеть, как то или иное правило, тот или иной класс влияет на отображение элемента на странице.



Обратите внимание, что при выделении элемента на странице, браузер использует цветовое отображение для содержимого элемента (синим), отступов внутри элемента (зеленым) и полей вне элемента (оранжевым).

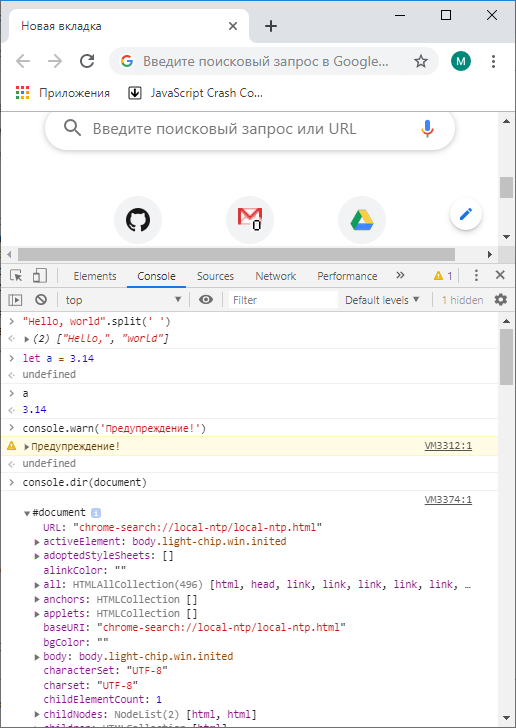


Это помогает правильно выстроить параметры полей и отступов (margin, padding) и толщину границы (border). Для удобства, справа внизу отображается схематичная box model элемента, где также присутствует цветовые обозначения этих параметров, и еще приведены конкретные значения полей, отступов, границ по всем четырем направлениям, и размеры самого элемента.

#### Использование консоли

Наверное, самая часто используемая возможность панели разработчика - javascript консоль. Она работает как любой интерактивный интерпретатор скриптового языка программирования. Многие используют ее только для того, чтобы посмотреть возникшие в процессе работы скриптов на страницы ошибки, однако, она способна на куда большее.

В этой консоли можно выполнять любой код JavaScript, причем о окружении этой консоли загружено дерево элементов страницы и исполнен весь клиентский код. Так, вы, например, можете:



* Выполнять любые инструкции;
* Выводить в консоль обычные сообщения, предупреждения и сообщения об ошибках;
* Создавать и использовать далее переменные;
* Обращаться к переменным, созданным в скриптах на странице;
* Выполнять функции, объявленные в скриптах на странице;
* Работать с объектной моделью документа, загруженного в текущей вкладке браузера;

Также в консоли браузера работает подсветка и автодополнение кода.

Это невероятно мощный инструмент, фактически превращающий обычный браузер в интегрированную среду разработки на JavaScript. Не в последнюю очередь распространение этого языка обязано тому, что для полноценной разработки на нем не обязательно иметь набор специализированных инструментов разработчика, а достаточно одного лишь браузера, который присутствует на любом компьютере.

Консоль настолько полезна, что ее можно вызвать по клавише Escape в любой вкладке панели разработчика.

#### Просмотр ошибок и предупреждений

Чаще всего к консоли обращаются, чтобы узнать, где и почему в программе возникла та или иная ошибка. Все ошибки отображаются специальным образом в консоли. В любом скрипте вы можете образаться к консоли используя следующие методы:

* console.log(“...”) - вывод на консоль простого строкового сообщения.
* console.dir(object) - вывод на консоль списка свойств указанного объекта.
* console.warn(“...”) - отображение в консоли строки в виде предупреждения. Предупреждения выводятся на желтом фоне для наглядности.
* console.error(“...”) - отображение в консоли строки в виде ошибки. Ошибки выводятся на красном фоне для наглядности.

Кроме того, при выводе ошибок и предупреждений, браузер сопровождает сообщение указанием на конкретное место в коде, где это предупреждение или ошибка возникла. Так, вы можете отследить, где в вашем (или не в вашем) файле возникла нештатная ситуация.

Кроме этого у встроеногообъекта консоли существует множество функций, как например, вывод массвов объектов в виде таблиц, группировка сообщений, замеры времени и так далее.

#### Отладочные сообщения

Консоль можно использовать при разработке клиентского кода для отладки путем введения отладочных сообщений. В том месте программы, где вы хотите, вставьте инструкцию console.log() или console.dir().

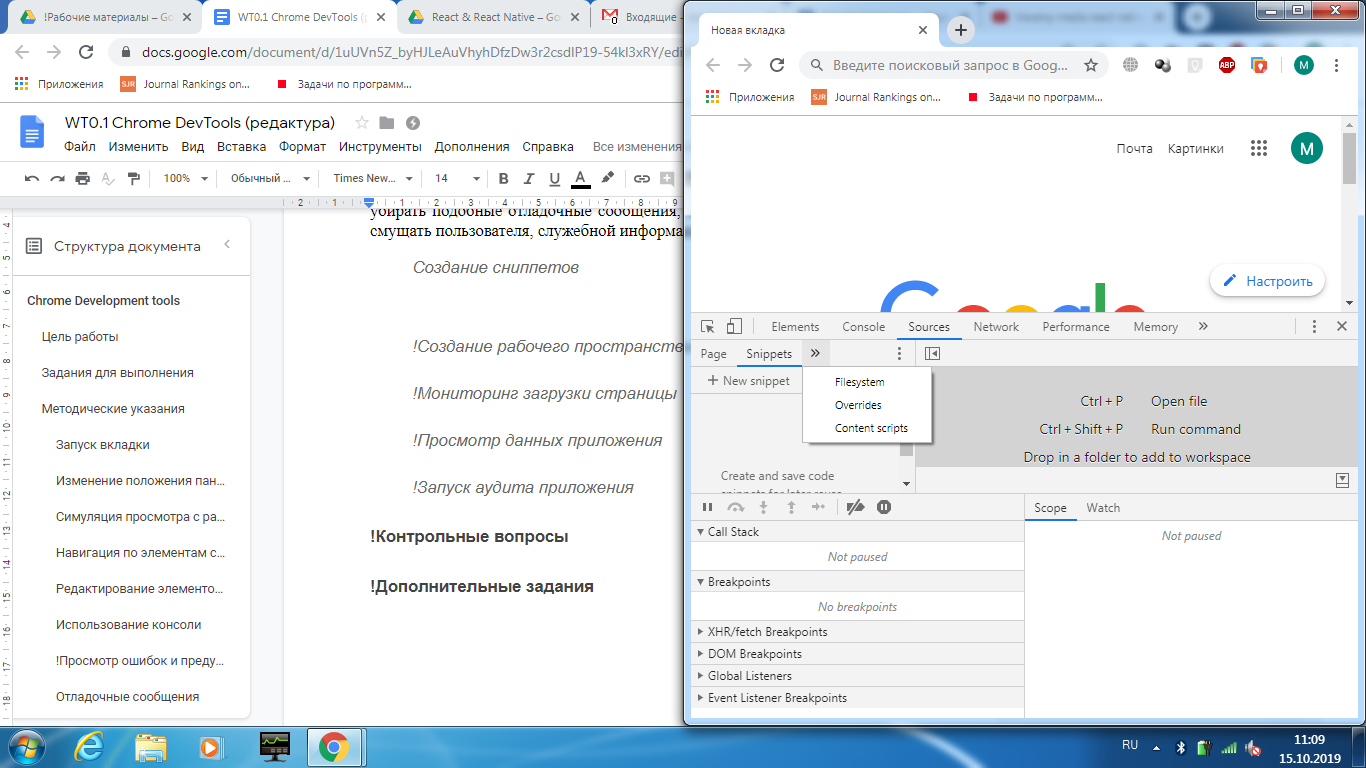
Этот прием можно использовать, например, для:

* просмотра содержимого объектов;
* наблюдения за значением переменных;
* проверки, по какой ветке пошло выполнение программы;
* отслеживания последовательности вызова функций;

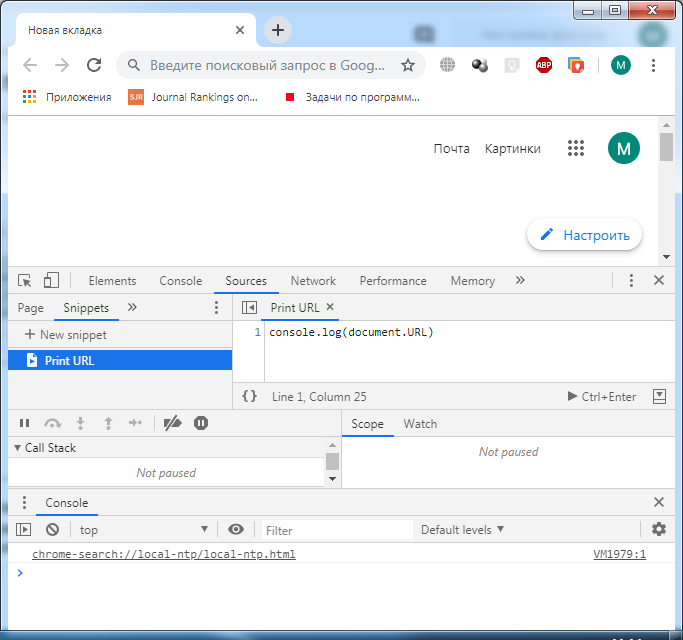
Следует помнить, во-первых, что с ростом сложности приложения этот способ отладки становится все более неудобным, для всего этого существуют специализированные инструменты; во-вторых, при разворачивании приложения для пользователей строго рекомендуется убирать подобные отладочные сообщения, дабы не засорять консоль и не смущать пользователя, служебной информацией.

#### Создание сниппетов

Вкладка “Sources” позволяет, помимо прочего, создавать сниппеты. Сниппеты - это небольшие подпрограммы, которые сохраняются в браузере и пользователь может их выполнить в произвольный момент времени.



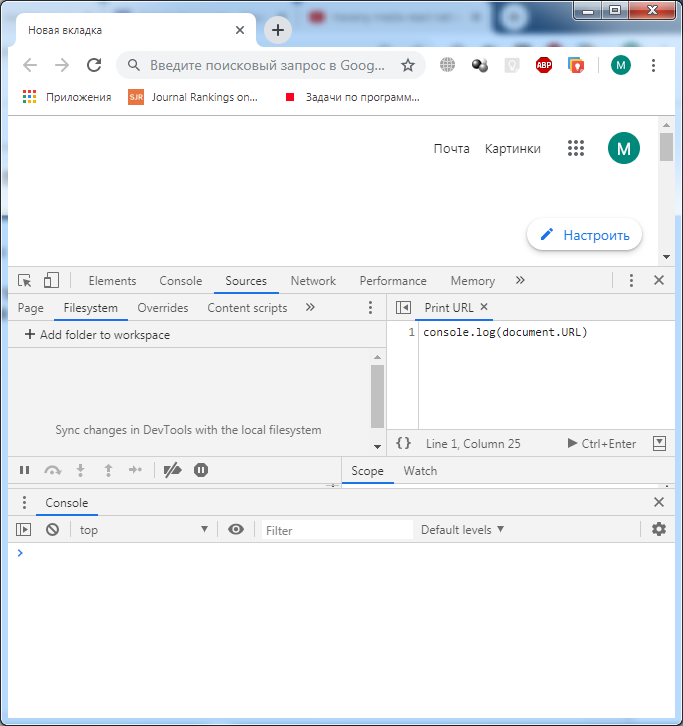
Для создания сниппета зайдите в панель “Snippets”, создайте новый сниппет, введите его имя. В панели справа вы можете ввести код, и сохранить его. При щелчке правой кнопкой на сниппете вы можете его запустить. Например, создадим сниппет, выводящий в консоль адрес открытой веб-страницы:



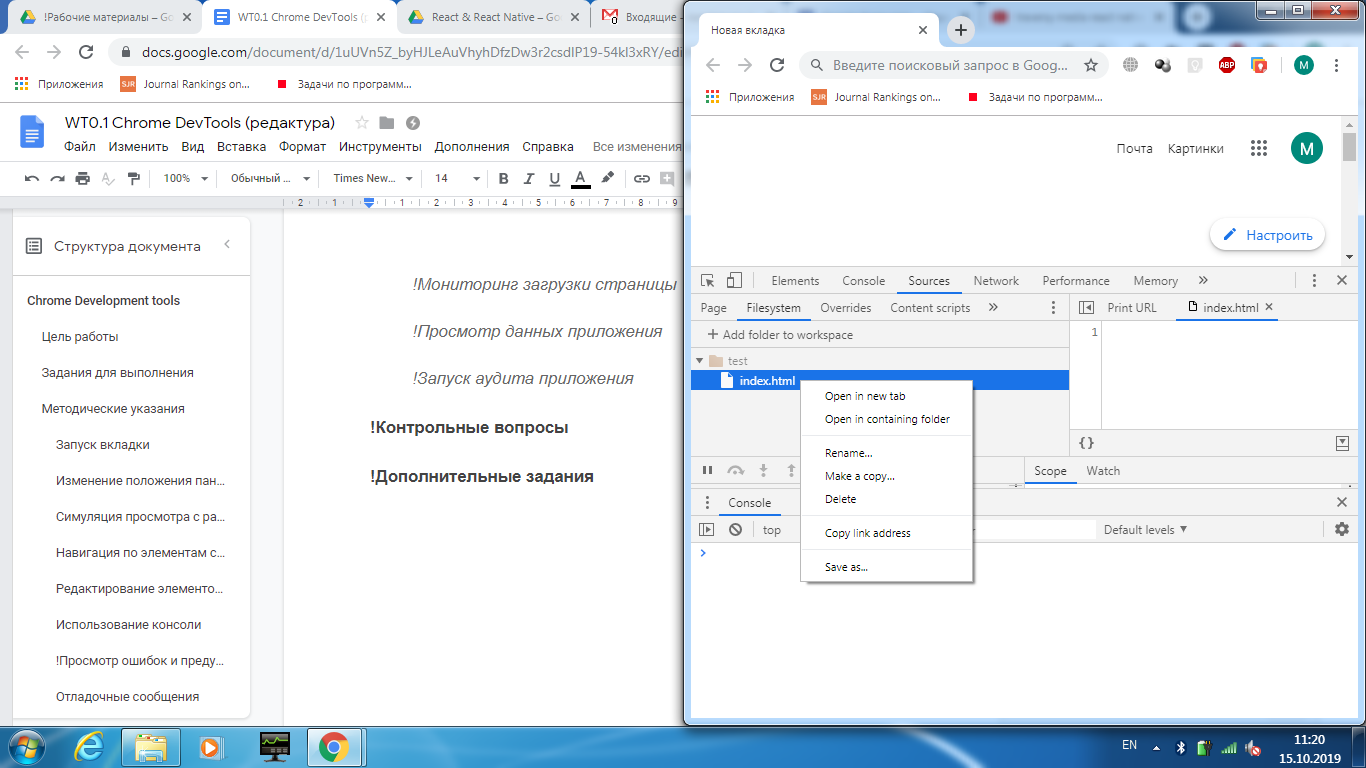
#### Создание рабочего пространства

Как мы описывали выше, вы можете редактировать код любой веб-страницы. Конечно, это только временные изменения, которые происходт с локальной копией кода, открытой в браузере. При перезагрузке страницы весь код вернется в первоначальный вид. И это логично, ведь у вас не может быть возможности редактировать код удаленных файлов.

Панель “Filesystem” на той же вкладке “Sources” позволяет превратить браузер в полнофункциональный тектовый редактор. Достаточно нажать на кнопку “Add folder to workspace”. Браузер запросит у вас разрешения на доступ к локальной файловой системе, и предоставит вам выбор папки.



Так вы можете создать папку, добавить ее к рабочему пространству, создавать и редактировать файлы в этой папке и открывать эти файлы в самом браузере.

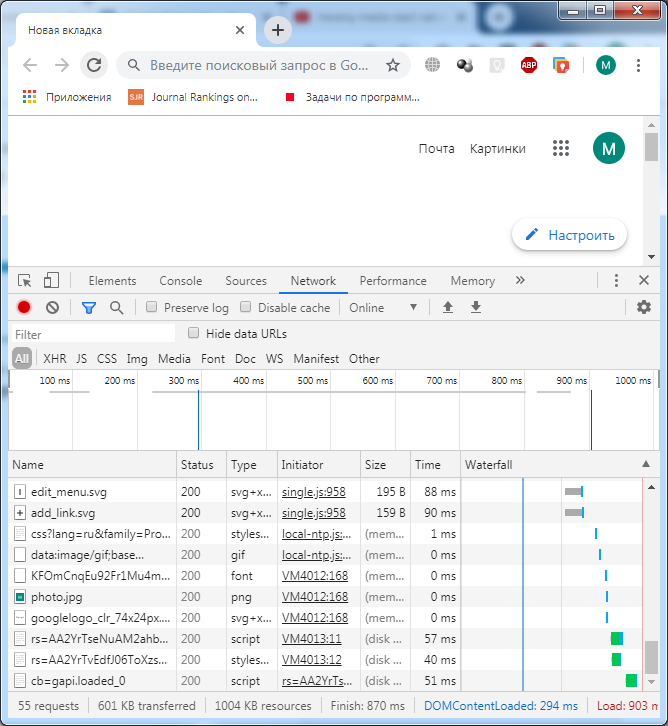


Изменения, произведенные таким образом, будут сохраняться на диск в папках, добавленных в рабочее пространство. Таким образом, вы можете создавать веб-страницы, стили и клиентские скрипты непосредственно в браузере и здесь же мониторить внесенные изменения и производить отладку.

#### Мониторинг загрузки страницы

Вкладка “Network” показывает все файлы, загруженные для отображения данной веб страницы. Обычно их бывает довольно много. На этой вкладке можно их фильтровать по типу. Также здесь показываются все AJAX запросы, открытые веб-сокеты и многая другая информация.

Самая интересная возможность этой вкладки - замер времени загрузки каждого элемента и веб страницы в целом.

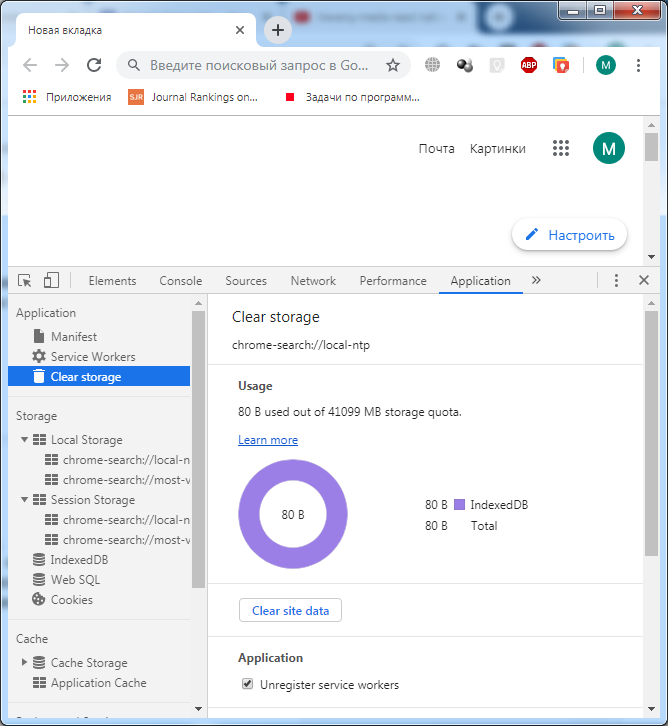


В верхней части панели в графическом виде отображается информация о времени загрузки все веб-страницы. На временной оси показаны две главные точки: время загрузки дерева элементов страницы (и, как следствие, время начала рендеринга страницы) - синим и время окончания загрузки (время, когда страница стала полностью отзывчивой) - красным.

Это очень полезная информация для разработчиков, которые хотят оптимизировать время загрузки своего приложения для улучшения пользовательского опыта.

#### Просмотр данных приложения

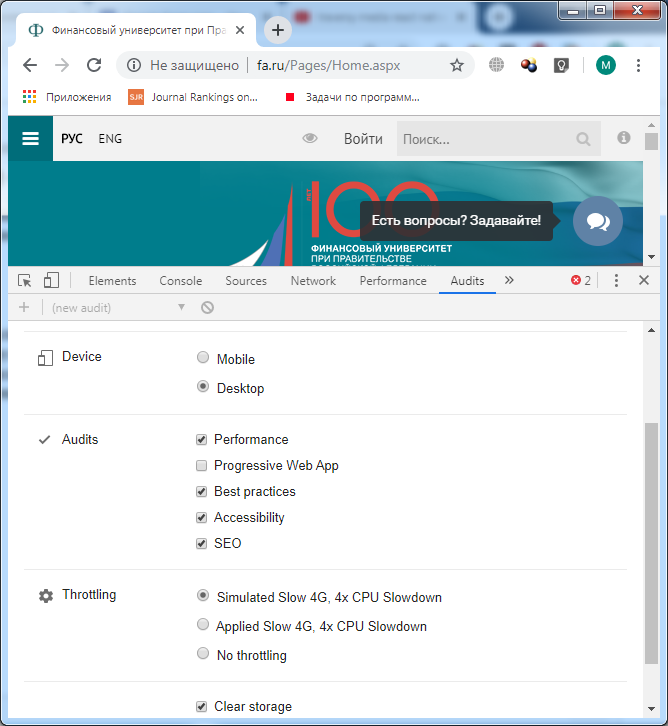
Вкладка “Application” показывает информацию о том, какие данные загруженное приложение хранил в локальном хранилище вашего браузера. Существует несколько разных видов локальных хранилищ - непосредственно локальное хранилище, которое хранит пары имя=значение, хранилище сессии, которое очищается каждый раз при перезапуске браузера, реляционные хранилища, по сути, маленькие выделенные базы данных, традиционные кукисы.



Также здесь можно просмотреть информация об общем потреблении памяти на данные приложения или очистить хранилище данного приложения, если это необходимо.

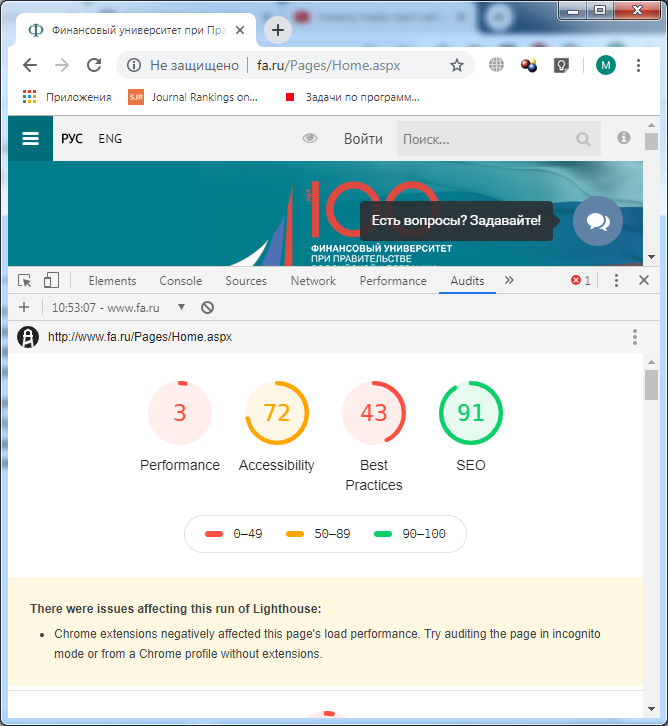
#### Запуск аудита приложения

Вкладка “Audits” позволяет запускать специальные комплексные проверки приложения, разработанные авторами браузера (аудиты), призванные показать сильные и слабые места приложения, дать советы по совершенствованию и оптимизации приложения.



Вы можете настроить параметры проверки, то есть аспекты, на которые вы хотите проверить приложение, выбрать целевое устройство, выбрать предопределенные наборы проверок, симулировать ограниченную скорость подключения и мощность процессора.

При запуске, аудит занимает определенное время, обычно не больше пары минут. После окончания аудита, панель будет показывать результаты проверки, выраженные в процентах по разным комплексным показателям (производительность, доступность, использование рекомендуемых практик, поисковая оптимизация) и рекомендации по улучшению производительности приложения:



### Контрольные вопросы

1. Какими способами можно отлаживать адаптивную верстку сайта?
2. Какие основные инструменты для отладки клиентского JavaScript кода предоставляет браузер?
3. Как веб-приложение может хранить информацию на стороне клиента?

### 

1. <https://www.youtube.com/watch?v=x4q86IjJFag&list=PLillGF-RfqbYeckUaD1z6nviTp31GLTH8&index=30&t=0s> [↑](#footnote-ref-0)