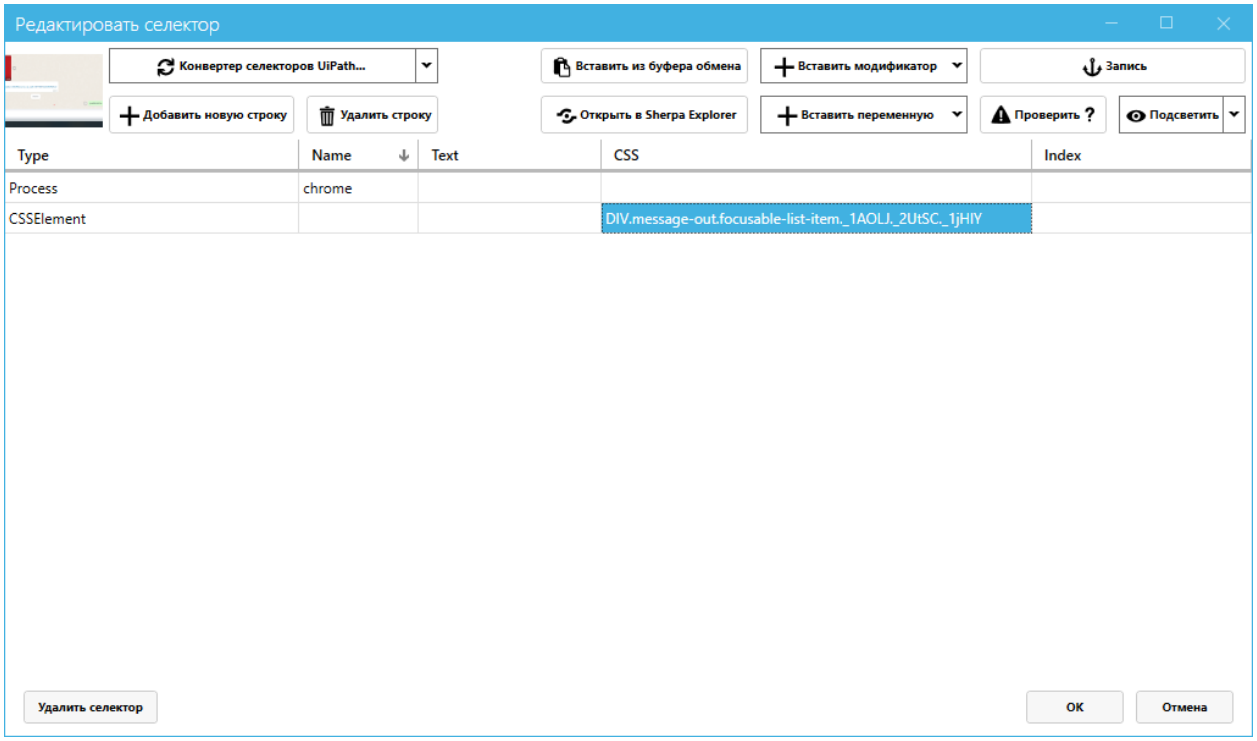
Руководство по веб-селекторам для профессионалов

1. Селекторы для веб-страниц в Sherpa RPA

В данном документе мы не рассматриваем базовые способы записи и использования селекторов в **Sherpa RPA** с помощью кнопки и панели **Запись**. Если вы не знакомы с базовыми приемами записи селекторов – обратитесь к «Руководству пользователя Sherpa RPA» и обучающим видео на нашем официальном YouTube-канале. Также в данном документе мы не рассматриваем десктоп-селекторы. Этот документ рассказывает о способах подбора веб-селекторов в сложных случаях.

Для поиска элементов на веб-странице в блогах из палитры «Автоматизация браузеров» могут использоваться CSS или XPath селекторы. В **Sherpa RPA** инструмент записи действий пользователя (рекордер) по умолчанию записывает CSS селекторы.



Если вы хотите вручную использовать XPath-селектор вместо CSS-селектора, допишите к селектору в колонке CSS в окне **Редактировать селектор** префикс «xpath:»

Если вы задались вопросом, какой тип селекторов лучше – CSS или XPath, то на самом деле в большинстве случаев нет никакой разницы. Почти любой CSS селектор может быть преобразован в XPath, и наоборот, это просто два разных способа записи. CSS селекторы для одних и тех же элементов как правило короче чем XPath, и их удобнее читать, поэтому в **Sherpa RPA** по умолчанию используются именно они. XPath-нотация обладает дополнительным функционалом для работы с текстовым содержимым элементов, однако в **Sherpa RPA** для этой цели существует отдельная колонка в окне селекторов – **Text**, которая даёт те же самые возможности CSS селекторам, и более удобным способом.

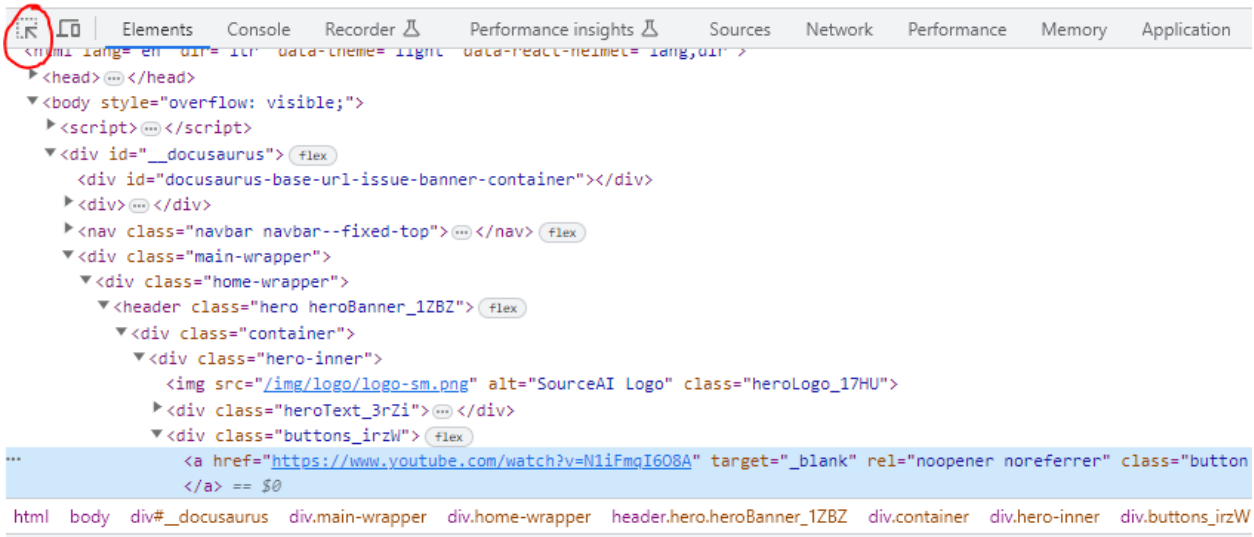
Перечень возможных префиксов перечислен в выпадающем меню кнопки **Вставить модификатор** в окне **Редактировать селектор**. В частности, доступен префикс «regex:», который позволяет в

текстовых колонках селектора (например, в колонке **Text**) использовать регулярные выражения. Также в текстовых колонках селектора доступны подстановочные символы ***** (замена любой последовательности символов) и **?** (замена ровно одного символа).

2. Ручной поиск CSS селекторов в браузере

В случае, если инструмент записи действий пользователя (кнопка **Запись**) записал не оптимальный селектор, вы можете подобрать альтернативный селектор вручную, ориентируясь на исходный код нужного элемента управления в веб-странице, который можно увидеть с помощью **Инструментов разработчика** вашего браузера. Например, в **Google Chrome** в основном меню вы можете выбрать **More Tools -> Developer Tools** или нажать **Ctrl+Shift+I**. В появившейся панели **Developer Tools** вы увидите текущий исходный код веб-страницы на вкладке **Elements**. Вы также можете быстро открыть этот инструмент и перейти к коду нужного вам элемента управления, если на веб-странице кликните правой кнопкой мыши и выберите пункт контекстного меню **Inspect**. Код, соответствующий нужному элементу, будет подсвечен на вкладке **Elements** в **Developer Tools**.

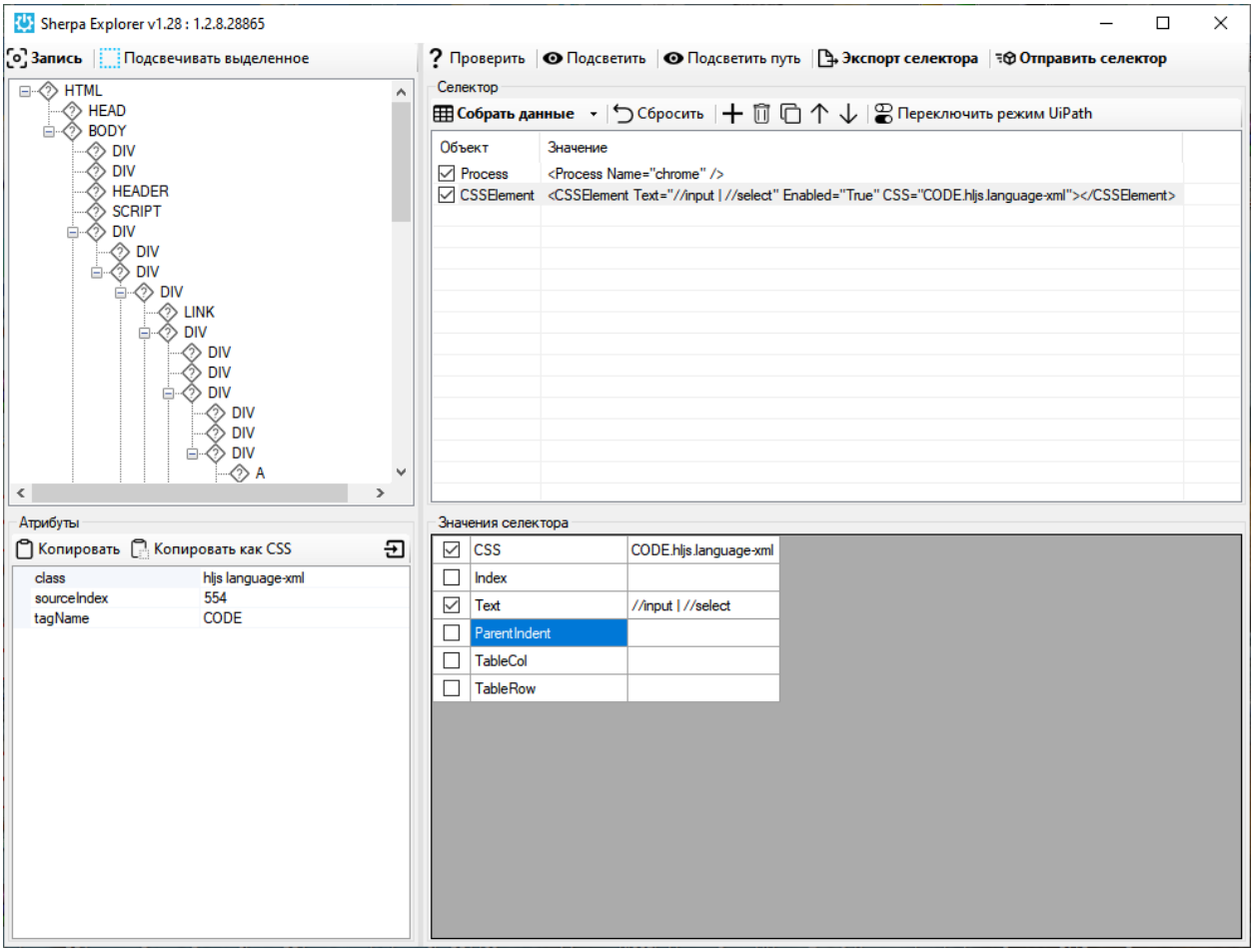
Некоторые веб-страницы блокируют отображение стандартного контекстного меню браузера и вместо него по правому клику показывают собственное меню. Также в некоторых случаях клик по элементу меняет состояние элемента так, что у элемента пропадает нужный нам атрибут. Например, вы хотите записать селектор непрочитанного сообщения в мессенджере, но, когда вы кликаете на него правой кнопкой мыши, сообщение становится прочитанным, и код этого элемента в **Inspect** меняется. Чтобы увидеть код нужного элемента в этих двух ситуациях – используйте кнопку **Select Element** - в **Google Chrome** это самая первая кнопка на панели **Elements**:



В других браузерах (Edge, Firefox и т.д.) инструмент **Developer Tools** выглядит и работает похожим образом, но конкретные названия пунктов меню и комбинации клавиш для его вызова могут отличаться.

Важно понимать, что современные веб-страницы являются динамическими структурами, и они часто меняют сами себя (названия и само наличие и отсутствие тех или иных элементов управления, их порядок и иерархию, их внешний вид и атрибуты) уже после того как были загружены с сервера в кэш вашего веб-браузера. Кроме того, страницы часто загружаются по частям, с помощью фреймов или динамического изменения **DOM-дер**ева элементов (DOM расшифровывается как «Document Object Model» – то есть модель веб-страницы, представленная в виде иерархического дерева тегов, именно такое дерево вы видите в Developer Tools браузера), в том числе с помощью таких технологий как **Ajax**. Чтобы увидеть исходный HTML-код страницы в том виде, как он был первоначально получен браузером с сервера, нужно выбрать пункт **View page source** в контекстном меню страницы браузера. Однако этот код для целей автоматизации почти бесполезен, так как с момента загрузки страницы браузер, следуя заложенным в страницу скриптам, уже успел много раз её изменить. **Developer Tools** браузера показывает вам динамический, обновляемый в реальном времени слепок того, как DOM-дерево страницы выглядит прямо сейчас, поэтому для ручного подбора селекторов следует пользоваться именно им.

3. Ручной поиск CSS-селекторов с помощью Sherpa Explorer



Ещё один мощный инструмент ручного подбора и тестирования селекторов доступен в самом Sherpa Designer и вызывается по кнопке Sherpa Explorer. В нём, как и в обычном редакторе

селекторов вы можете записать любой элемент интерактивно, а также исследовать иерархическое дерево любого Desktop и веб-приложения, его доступные атрибуты, подобрать и протестировать нужное сочетание атрибутов. Подобранный и проверенный селектор можно отправить в редактор селекторов самого Дизайнера с помощью нажатия на соответствующую кнопку.

4. Как понять, что селектор не оптимальный?

Для одного и того же элемента веб-интерфейса можно подобрать много разных селекторов, все из которых будут в той или иной степени рабочими. Например, в веб-клиенте WhatsApp селектор

#main > footer > div._2lSWV._3cjY2.copyable-area > div > span:nth-child(2) > div > div._1VZX7 > div._2xy_p._3XKXx > button

и селектор

button[aria-label="Отправить"]

будут обозначать один и тот же элемент управления – кнопку «Отправить». И они оба будут работать в моменте. Первый селектор опирается на длинный путь от самого начала (корневого элемента) веб-страницы до конкретного элемента управления «вниз» по дереву иерархии элементов управления. Второй селектор опирается на тип тега элемента управления и один характерный атрибут этого тега. Несмотря на то, что в момент записи оба селектора работают корректно, первый селектор с гораздо большей вероятностью «сломается», то есть перестанет работать, при очередном обновлении веб-приложения или веб-сайта. Например, по пути селектора появятся или наоборот исчезнут промежуточные элементы управления, и, следовательно, изменится сам путь от корневого элемента до нужного нам.

Также, если в селекторе присутствуют псевдослучайные комбинации букв и цифр (такие как 1VZX7 или 3XKXx в примере выше), это также является признаком потенциально нестабильного селектора. Например, эта комбинация букв и цифр может меняться при каждом заходе пользователя на сайт, и для другого пользователя она будет уже другой. Проверить это можно, разлогинившись из сайта, закрыв браузер, открыв его и снова перейдя на нужную страницу, либо просто зайдя на эту страницу с другого компьютера или браузера, и проверив – те же буквы и цифры вы видите в этом месте селектора, или уже другие. Также может помочь (но не всегда) комбинация **Ctrl+F5**, которая вызывает полную перезагрузку страницы. Но даже если после всех этих действий комбинация букв и цифр не поменялась, она может поменяться после следующего билда (обновления) веб-сайта или веб-сервиса, т.к. современные фреймворки веб-сайтов (такие как Angular, React и т.д.) могут генерировать эти названия классов случайным образом при каждой пересборке страницы.

Таким образом, чтобы робот в продуктиве работал надежно, и разработчику не приходилось слишком часто исправлять селекторы в блоках автоматизации браузеров, очень важно на этапе разработки подобрать как можно более стабильный селектор. Не всегда, но очень часто самый стабильный селектор – это самый короткий из всех возможных для данного элемента.

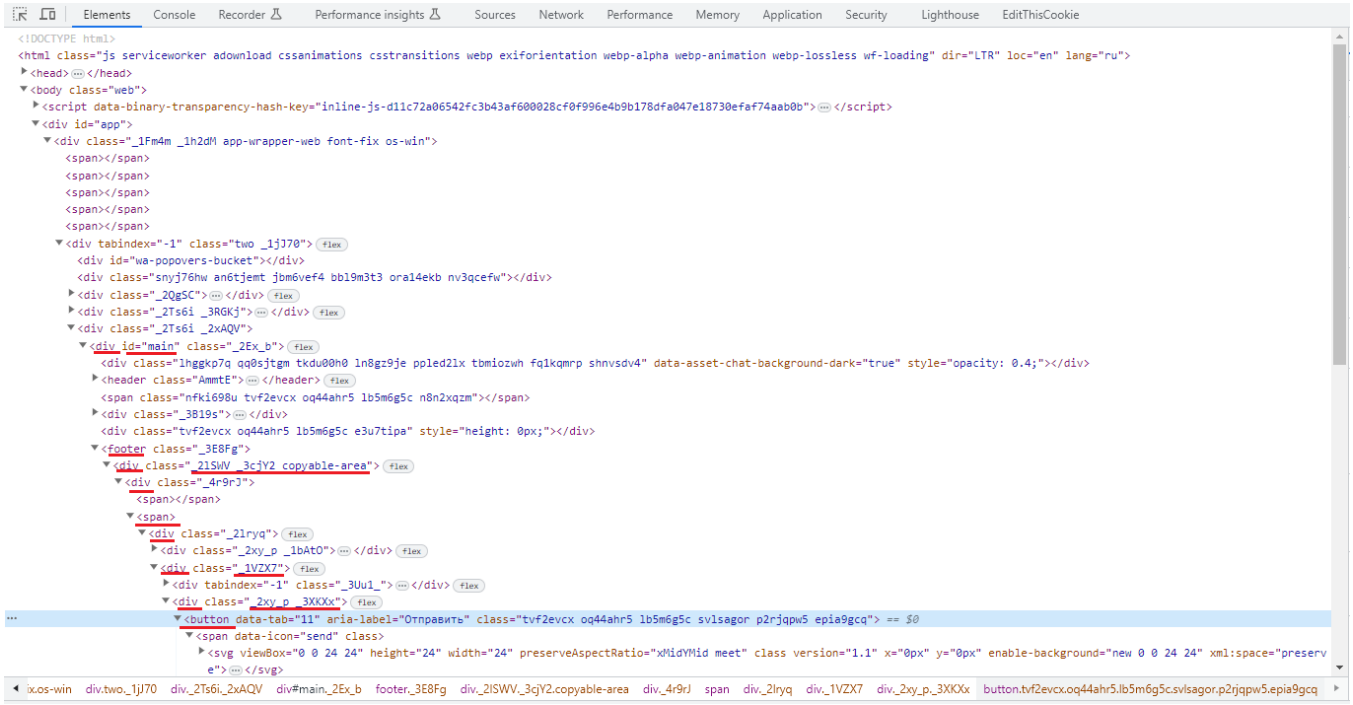
Если инструмент рекордера не выдаёт вам достаточно стабильный селектор, его нужно подобрать или отредактировать вручную. Для этого нужно знать нотацию CSS-селекторов.

5. Нотация CSS-селекторов на примерах

Давайте рассмотрим длинный селектор из WhatsApp, который мы записали выше:

#main > footer > div._2lSWV._3cjY2.copyable-area > div > span:nth-child(2) > div > div._1VZX7 > div._2xy_p._3XKXx > button

И сразу же посмотрим на участок кода в самой веб-странице, которая соответствует этому элементу:



Для вашего удобства, мы подчеркнули красным все участки кода и идентификаторы, которые входят в этот селектор, чтобы вы могли сопоставить их в селекторе и в HTML-коде страницы. Начнем прослеживать селектор с самого начала и спустимся вниз по иерархии вложенных тегов веб-страницы до нужного нам элемента управления. Запись **#main** означает, что нам нужно найти элемент, у которого атрибут **id** имеет значение **main**. В отличие от имен классов, которые могут повторяться много раз, **id** элементов встречаются на странице только по одному разу. В данном случае эта запись нашла нам тег ** **, при этом сам тег **div** или его класс никакой роли не играли. Символ **>** означает, что мы должны спуститься вниз по иерархии, к дочерним элементам найденного тега, и последующую часть селектора искать среди них.

Далее следует название тега **footer**, здесь мы нашли соответствующий тег просто по его названию, без всяких дополнительных уточнений. Далее видим ещё один символ **>**, значит спускаемся ещё ниже по иерархии элементов. Следующая запись **div._2lSWV._3cjY2.copyable-area** означает, что мы должны найти тег **div**, среди классов которого должны быть одновременно **_2lSWV**, **._3cjY2** и **copyable-area**. Классов у самого искомого элемента может быть и больше, мы указываем только тот необходимый минимум, который нам нужен. Таким образом мы находим тег ** **, который подходит под этот селектор. Обратите внимание, что в селекторе имена классов соединены через точку (которая в данном случае означает логический оператор **И**),

а в тексте атрибута **class** в самом теге эти же имена классов перечислены через пробел – это важное отличие записи имен классов в атрибутах тегов и в селекторах.

Спускаемся ещё ниже, находим следующий элемент просто по названию тега **div**. Т.к. тегов **div** на этом уровне иерархии много, будет взят первый попавшийся. Спускаемся ещё ниже. Запись **span:nth-child(2)** говорит нам о том, что мы должны найти тег **span**, и далее перейти во второй по счёту дочерний элемент этого тега. Затем ещё раз находим элемент **div** просто по имени элемента. Затем находим тег ** ** который соответствует селектору **div._1VZX7** – по названию тега и имени класса в этом теге. Затем находим элемент ** ** по селектору **div._2xy_p._3XKXx**, ну а в конце находим дочерний тег **button** просто по названию самого тега. Это была достаточно длинная цепочка, и по этому пути многое может пойти не так, если сайт будет обновляться. Зато на этом примере мы посмотрели, какие разные варианты нотации используются в CSS селекторах.

Сравните с коротким селектором **button[aria-label="Отправить"]**, который выделяет тот же элемент управления, и который мы предложили как альтернативу. Почему мы вручную составили именно такой селектор? Сначала мы посмотрели на финальный тег, который нам нужен:

<button data-tab="11" aria-label="Отправить" class="tvf2evcx oq44ahr5 lb5m6g5c svlsagor p2rjqpw5 epia9gcq">

Имена классов нам сразу не нравятся – их много, и они выглядят как произвольные сочетания букв и цифр, а это означает что в новой версии веб-страницы они почти наверняка изменятся. Название тега **button** выглядит перспективно. Не так уж много кнопок на этой странице. Но тем не менее само по себе название **button** может быть неуникальным – на странице могут быть и другие кнопки. Конечно мы можем использовать модификатор вроде **button:nth-child(1)**, который укажет нам на точный порядковый номер однотипного элемента на странице. Или то же самое можно сделать, занеся число 1 в колонку **Index** справа от колонки **CSS** в окошке редактора селекторов Sherpa RPA – это даст тот же эффект.

Но кто даст гарантию, что при следующем обновлении страницы порядок или количество кнопок на странице не изменятся? Текст **«Отправить»** в атрибуте **aria-label** выглядит достаточно уникальным – это тот текст, которым будет замещен элемент управления в «экранных читалках» и других accessibility-инструментах для лиц с ограниченными возможностями. Очень маловероятно что разработчики сайта будут менять это название в следующих версиях веб-приложения, поэтому его-то мы и возьмем за характерный отличительный атрибут. Чтобы указать, что нам нужен только такой тег button, у которого атрибут **aria-label** имеет значение **«Отправить»,** используем квадратные скобки - **button[aria-label="Отправить"]**. Мы получили достаточно надежный селектор, который не стыдно использовать в продуктиве нашего робота.

6. Справочник по нотации CSS и XPath селекторов

Ниже мы перечислим популярные типы CSS селекторов и соответствующие им селекторы XPath. Обратите внимание, что некоторые устаревшие версии браузеров могут не поддерживать отдельные типы селекторов, описанные ниже. Также не забывайте, что все перечисленные ниже элементы селекторы можно легко комбинировать друг с другом в рамках одного большого селектора.

6.1 Значение ID элемента

Идентификатор элемента в CSS определяется с помощью **#**, а в XPath с помощью **[@id='example']**. Идентификаторы должны быть уникальными в пределах DOM-дерева страницы.

Примеры:

CSS: #example

XPath: //div[@id='example']

6.2 Название тега

В предыдущем примере мы использовали запись **//div** в варианте для XPath. Это название тега, например, **input** для текстового поля или кнопки, **img** для изображения или **a** для ссылки. Можно выделять элемент просто по названию его тега.

Примеры:

CSS: input

XPath: //input

6.3 Название класса

В HTML коде названия классов пишутся внутри значения атрибута **class**, и, если классов у атрибута несколько, они разделяются (или точнее объединяются) пробелом. Но в селекторах имена классов указываются немного по-другому – в CSS названия классов перечисляются через точку, либо отделяются точкой от других частей того же элемента селектора, например, от названия тега. В XPath классы указываются в квадратных скобках в атрибуте **@class**.

Примеры:

CSS: div.example

XPath: //div[@class='example']

6.4 Значение атрибута

У HTML-тегов может быть множество атрибутов, и можно находить теги просто по названиям и значениям этих атрибутов. В примерах ниже мы найдем тег **input** по атрибуту **name** и его значению **«username»**:

CSS: input[name='username']

XPATH: //input[@name='username']

Также есть возможность искать элементы по значениям сразу нескольких атрибутов.

Примеры:

CSS: input[name='login'][type='submit']

XPATH: //input[@name='login'and @type='submit']

6.5 Частичное значение атрибута

Если значение атрибута, с помощью которого вы хотите найти элемент, частично изменяется в процессе работы сайта, вы можете использовать для поиска только ту часть значения, которая остаётся неизменной. Такая возможность есть только в CSS-селекторах:

Поиск элемента по префиксу значения атрибута:

CSS: a[id^='id_prefix_']

В данном случае мы найдем гиперссылку, у которой атрибут ID начинается на **«id_prefix_»**.

Поиск элемента по суффиксу значения атрибута:

CSS: a[id$='_id_sufix']

В данном случае мы найдем гиперссылку, у которой атрибут **ID** заканчивается на **«_id_sufix»**.

Поиск элемента по подстроке в значении атрибута:

CSS: a[id*='id_pattern']

В данном случае мы найдем гиперссылку, у которой атрибут **ID** содержит подстроку **«id_pattern»**.

6.6 Объединение результатов нескольких селекторов

В примере с WhatsApp выше все исходящие сообщения можно найти с помощью селектора **div. message-out**, а все входящие сообщения с помощью селектора **div. message-in**. А что делать, если мы в одном списке хотим получить или перебрать и входящие, и исходящие сообщения, причем в том порядке, в котором они идут в чате? В CSS для этих случаев можно легко объединить списки элементов, полученные разными селекторами с помощью запятой. В XPath для этих же целей используется символ вертикальной черты.

Пример:

CSS: div.message-out, div.message-in

XPath: //div[@class=' message '] | //div[@class= 'message-in']

6.7 Непосредственный дочерний элемент

Страницы HTML имеют структуру, подобную XML, с дочерними элементами, вложенными в родительские. Прямой дочерний элемент определяется в CSS с помощью символа **>**, а в XPath с помощью символа **/**.

Примеры:

CSS: div > a

XPath: //div/a

6.8 Дочерний элемент ниже по иерархии

Описание цепочки всех непосредственных дочерних элементов, во-первых, утомительно, а во-вторых приводит к ненадежным селекторам. Если элемент может находиться внутри другого или внутри одного из его дочерних элементов, он определяется в CSS просто с помощью пробела, а в XPath с помощью **//**.

Примеры:

CSS: div a

XPath: //div//a

6.9 Следующий потомок

Этот тип селектора полезен, чтобы перебирать однотипные элементы на одном уровне иерархии, например, строки таблиц, элементы нумерованных или ненумерованных списков. Следующий потомок – это следующий элемент однородного списка элементов на том же уровне иерархии что и текущий.

Представьте себе, что у вас в форме идут подряд два тега **input**, вот таких:

<input type = "text" class = "form-control" id = "username" name = "username" placeholder = "username" required autofocus></br>
<input type = "password" class = "form-control" id = "password" name = "password" placeholder = "password" required>

По какой-то причине вы не можете адресоваться ко второму **input** напрямую, но у вас есть устойчивый селектор для первого **input**. В такой ситуации, для выделения второго **input** в CSS вам нужно будет записать вот так:

#username + input

А в XPath вот так:

//input[@id='username']/following-sibling:input[1]

6.10 Псевдоклассы для однородных элементов

Существует несколько псевдоклассов, которые позволяют выделить нужный по счёту элемент просто по его порядковому номеру на нужном уровне иерархии (**nth-child**), либо же по порядковому номеру и типу элемента (**nth-of-type**).

Селектор ниже выделит четвертый по счёту тег **LI** на заданном уровне иерархии:

CSS: li:nth-of-type(4)

Если же мы хотим выделить четвертый по счёту тег, являющийся потомком тега **LI**, безотносительно типа этого тега, можно сделать вот так:

CSS: li:nth-child(4)

В XPath есть возможность указывать нужный по счёту дочерний элемент относительно текущего элемента просто приписав порядковый номер в квадратных скобках:

//li[4]