

Секреты Огнелиса

Александр Майоров

Сегодня каждый опытный пользователь ПК, и Интернета в частности, знает о браузере Mozilla Firefox, в простонародье называемом Огнелисом. Но не каждый знает обо всех его секретах и возможностях. Он, конечно, не идеален, и это легко понять, зная его историю создания.

История браузера началась в 1998 году, когда фирма Netscape поняла, что ее продукт не может конкурировать с детищем компании Microsoft – Internet Explorer, который был встроенным в новейшую ОС того времени, Windows 98. В компании Netscape решили, что единственный верный выход из сложившейся ситуации – написать новый бесплатный браузер, при этом используя опыт свободного программного обеспечения. Было решено выложить ис-

ходные тексты в Сеть для свободного доступа. К 2000 году, фактически из ничего, родилось чудовище под названием Mozilla. Браузер получился неповоротливый, медленный, изобилующий ошибками. Монстр под именем Netscape 6.0, собранный на основе одной из тестовых версий Mozilla, был ужасен на фоне быстрых Opera 5.x и Internet Explorer 5.0. Он загружался по полминуты, а стабильность оставляла желать лучшего. Версия 6.2 положения не исправила, а 7.0 и 7.2 хоть

и стали конкурентоспособными, но репутация Netscape уже была подорвана окончательно.

В 2002 году дела у Mozilla.org пошли на поправку. Вышла «единичка» – очень стабильный релиз, на который тут же обратили внимание локализаторы. В России за дело взялись специалисты из ALT Linux, и вскоре мы увидели версию Mozilla 1.0 с переведенным на русский язык меню, настройками и закладками на отечественные ресурсы. Летом того же года

вышел браузер Phoenix 0.1 (нынешний Firefox) – сильно облегченная версия Mozilla 1.0 с возможностью конфигурирования кнопок на панели. Затем появился миниатюрный K-Meleon – на движке от Mozilla, но с внешностью от текущей версии IE на вашей машине. Переломный момент наступил спустя год, когда mozilla.org стала независимым проектом. Давление со стороны AOL, под чьим крылом оказался проект, исчезло. Более того, AOL выделила 2 млн. долларов на его дальнейшее развитие. Программисты стали работать исключительно на пользователей, и за лето-осень 2003-го все программы от Mozilla.org изменились в лучшую сторону.

Браузер Firefox родился летом 2002 года. Это сейчас он так называется, а сначала долго страдал от неудачно выбранного имени. От Phoenix пришлось отказаться из-за существования одноименной BIOS. Следующее имя было Firebird. Но оно также не прижилось из-за претензий со стороны разработчиков одноименной базы данных. Сегодня все юридические споры улажены.

На этом я закончу рассказ об истории создания браузера. Те, кому мало данной информации, могут заглянуть в энциклопедию <http://ru.wikipedia.org/wiki/Firefox>, ну и поискать в Интернете.

Firefox обрел свою аудиторию благодаря колоссальной возможности изменения любых настроек браузера и завидной расширяемости, которая недоступна ни Opera, ни Internet Explorer. Сегодня среди огромной армии поклонников Firefox лишь малая часть знает о том, чем этот браузер так примечателен и как его правильно использовать. Остальные же просто используют голый движок Mozilla, лишь подозревая, что его функциональность можно каким-то образом расширить. Но, прежде чем говорить о плагинах, надо разобраться со стандартными возможностями браузера, ибо даже с ними знакомы не все.

Эта статья не призвана раскрыть все секреты Огнелиса, так как это невозможно сделать в рамках даже целого журнала. Я постарался отобрать самые полезные и часто применяемые возможности. Вторая половина статьи посвящена полностью поисковым плагинам.

Горячие клавиши

Чтобы открыть новую вкладку, надо дважды кликнуть на свободном месте панели вкладок. Это также можно сделать и с помощью горячих клавиш <Ctrl+T>.

Однажды закладок становится много, и в один прекрасный момент мы начинаем их упорядочивать. Многие для того, чтобы их рассортировать, используют окно управления закладками. Потому как, если просто пытаться перемещать их между директориями, ничего не получится. Однако функции drag'n'drop в Firefox реализованы. Для того чтобы их использовать, нужно удерживать клавишу <Shift>. При этом мы спокойно можем перемещать закладки из раздела в раздел мышкой.

Чтобы отыскать нужную информацию на странице, можно открыть окно поиска комбинацией клавиш <Ctrl+F>.

Но помимо окна поиска в Firefox есть панель быстрого поиска. Чтобы открыть ее, надо нажать клавишу со знаком слеш «/». При этом будет открыта панель быстрого поиска.

Также найти слово можно, набирая его прямо в окне браузера, при условии, что курсор не стоит в каком-нибудь поле ввода (работает, правда, только в ОС семейства Windows). Панель быстрого поиска в таком случае откроется автоматически. Слово тут же выделится желтым цветом.

Следующее совпадение будет найдено по нажатию <Ctrl+G> или <F3>.

Как правило, в браузере всегда открыто много вкладок. Чтобы быстро переключиться на нужную вкладку, достаточно нажать комбинацию <Ctrl+Tab>, тогда вкладки будут переключаться по порядку, слева направо.

Чтобы осуществить быстрый доступ к конкретной открытой вкладке, можно нажать <Ctrl+{число}>, где {число} – это номер вкладки. То есть <Ctrl+4> откроет четвертую по счету вкладку.

Чтобы открыть ссылку в новой вкладке, надо щелкнуть по ней средней кнопкой мыши либо держать <Ctrl> и кликнуть левой кнопкой мыши по ссылке.

Если вы читаете какую-то on-line документацию и у вас заняты руки, то, чтобы пролистать страницу вниз, необязательно пользоваться мышью. Достаточно нажать пробел.

Все ссылки, которые мы посещаем, сохраняются в истории посещений. Это удобно, так как при наборе URL браузер предлагает подходящие варианты во время ввода адреса. Очень удобная возможность. Но что делать, если скопилось много ненужных URL либо среди них присутствуют нежелательные ссылки, которые не должны увидеть другие (например, это компьютер на работе, и к нему имеют свободный доступ коллеги). Что тогда делать?

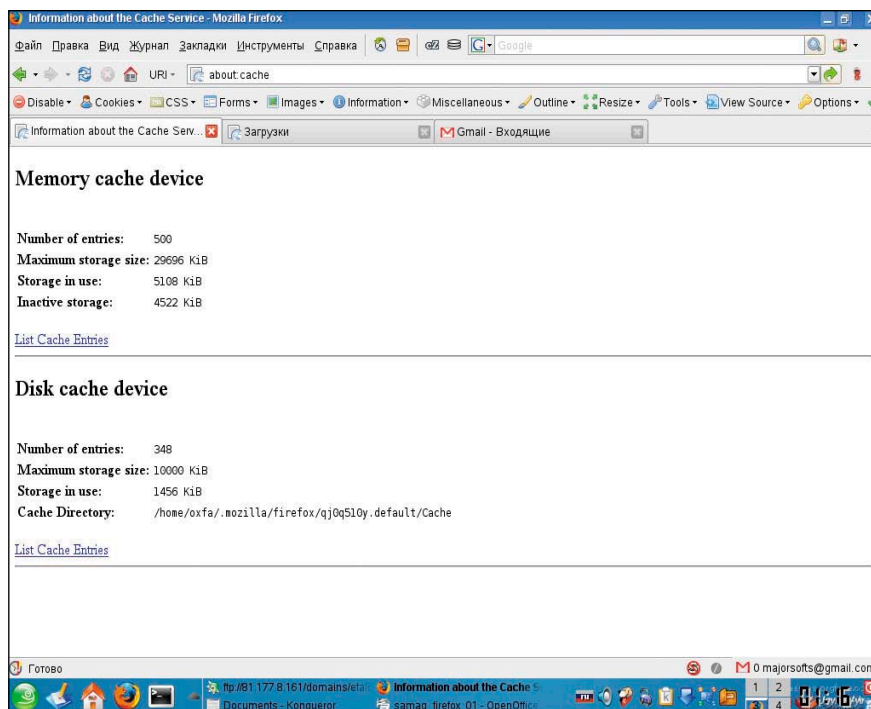


Рисунок 1. Информация о содержимом кэша

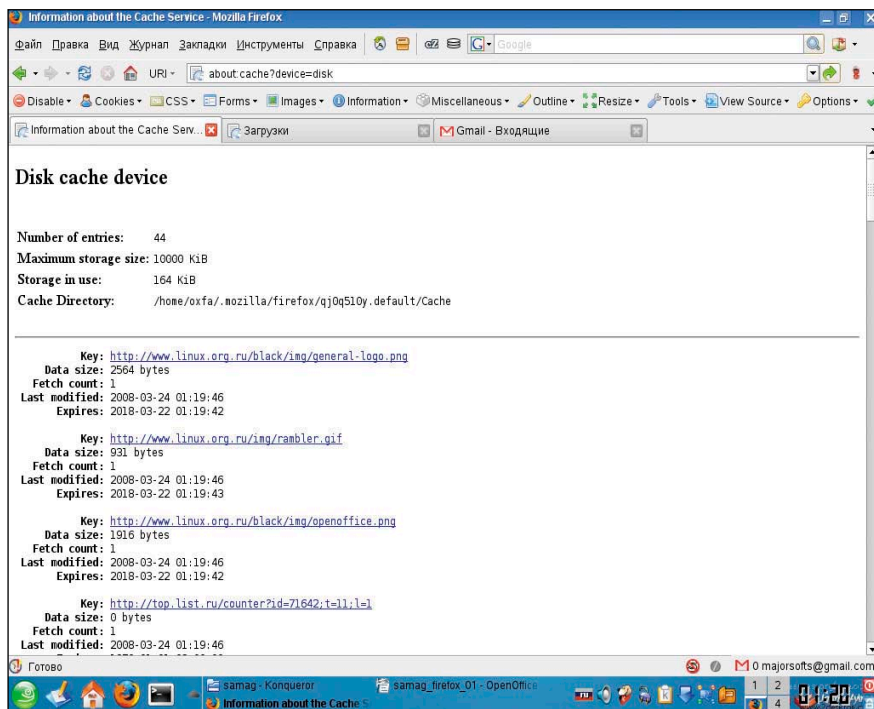


Рисунок 2. Информация о содержимом дискового кэша

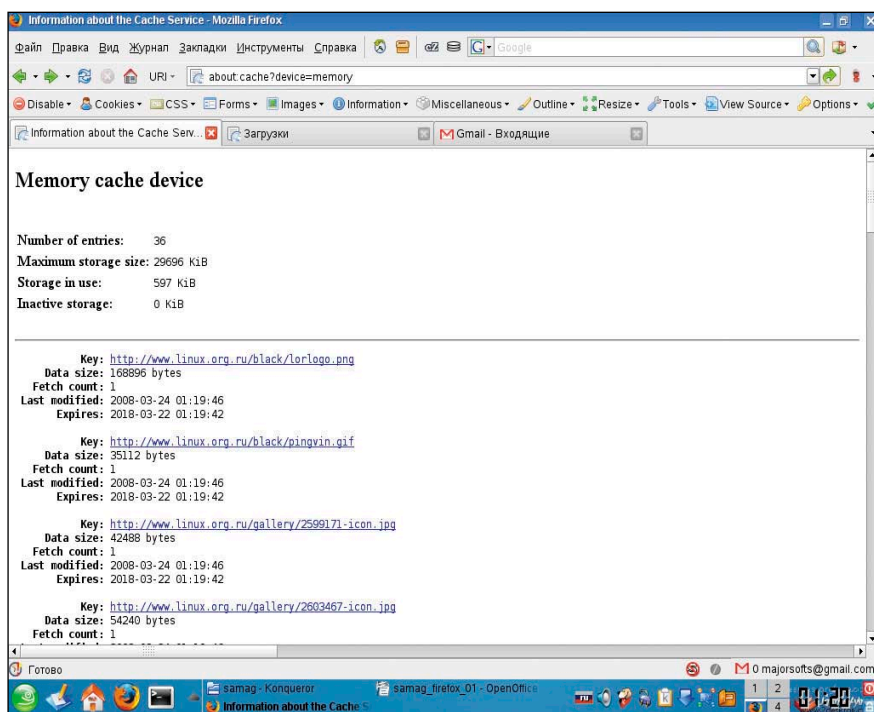


Рисунок 3. Информация о содержимом памяти процесса

Можно удалить всю историю посещений («Инструменты → Удалить личные данные»). Но если там много хороших ссылок, а удалить надо только одну или какие-то выбранные? Для этого достаточно начать вводить в адресе нужный URL, и, после того как Огнелис предложит вам варианты в выпадающем списке, курсором переместитесь к нужной ссылке и нажмите <Delete>. Данная ссылка будет удалена.

Этот совет уместен также для выпадающих списков автозаполнения. Со временем он может вырасти. Его выборочная очистка так же производится, как и очистка истории посещений. Кстати, если вы не хотите копаться в настройках браузера, а быстро очистить список (при условии, что он небольшой), то можете просто зажать кнопку <Delete>. Список будет очищен моментально.

Поисковые сокращения и алиасы

Для быстрого обращения к наиболее часто используемым закладкам можно назначить ключевые слова (так называемые алиасы). Для этого следует проделать следующее: нажмите правой кнопкой мыши на закладке, зайдите в ее свойства. Там будет поле «Краткое имя». В этом поле введите ключевое слово. Теперь достаточно в адресной строке набрать ваше ключевое слово, и Огнелис автоматически перейдет на нужный сайт. Для примера: создаем закладку на сайт <http://www.samag.ru>. В свойствах закладки в поле «Краткое имя» вводим «sm». Теперь, набрав в адресной строке «sm» и нажав <Enter>, Firefox автоматически перейдет на сайт www.samag.ru.

Для осуществления поиска мы часто пользуемся поисковой формой, которая позволяет нам обращаться к поисковым механизмам напрямую. Но не всегда среди этих форм есть поисковый плагин для нужной вам поисковой системы. Их легко собрать самому, но об этом чуть позже. Сейчас я расскажу, как можно сделать быстрый поиск без создания каких-либо дополнительных плагинов. Запустить поиск можно сразу из строки ввода адреса, указав специфичный префикс перед запросом.

Например:

google журнал системный администратор

приведет в Google, на страницу поиска, где будет результат запроса «журнал системный администратор».

В стандартную поставку Firefox входит несколько таких заготовок. Но что делать, если хочется иметь свой поисковый запрос на свой любимый поисковик? Все очень просто. Сейчас я покажу, как добавить свои собственные сокращения.

По сути дела, поисковые сокращения задаются в свойстве закладки, как уже описывалось выше, когда рассматривали создание псевдонимов для закладок. Набрав это имя в строке поиска, осуществляется немедленный переход на ссылку, сохраненную в закладке. Но как же передать параметры в закладку? На самом деле все очень просто. Если в этой

строке ввести %s, то вместо него будет подставлена строка, следующая за коротким именем в строке ввода. Давайте для примера добавим Яндекс в качестве поискового сокращения Firefox.

Для начала надо получить ссылку для закладки. Заходим на <http://www.ya.ru> и вводим что-нибудь в поиске, например «samag». Далее берем полученный адрес: «<http://www.yandex.ru/yandsearch?text=samag&rpt=rad>» и заносим его в закладки. Открываем свойства закладки. Находим в адресе ссылки строку «samag» и заменяем ее на %s. Вводим краткое имя, например «yandex». Сохраняем изменения.

Теперь попробуйте в адресной строке ввести:

```
yandex системный администратор
```

Мы попадаем на страницу результатов yandex.ru.

Схемы, поддерживаемые Firefox

Каждый браузер работает с так называемыми схемами, определенными стандартом. Например, обработка ссылок происходит через схему http, работа с FTP через схему ftp. Так же есть схемы: javascript, data, mailto и прочие. Их очень много, все описать я здесь не смогу.

Некоторые схемы добавляются сторонними приложениями (например, торрент-клиенты имеют свои схемы). Я расскажу только про самые интересные. Начну со схемы data.

data: url – это определенная стандартом RFC 2397 схема, которая позволяет включать небольшие элементы данных в строку URL, как если бы они были ссылкой на внешний ресурс. Она гораздо проще альтернативных методов включения, таких как MIME с cid: или mid:. Согласно букве RFC «data: url» – это фактически URL (url – унифицированный указатель ресурса), хотя реально он ни на что не указывает.

Формат этой схемы:

```
data:[<mediatype>][;encoding],<data>
```

<mediatype> – спецификация типа носителей данных (с дополнительными параметрами; см. MIME).

Появление «base64» означает, что данные закодированы в base64. Без «base64» данные (как последовательность байтов) представляются с использованием кодировки ASCII в диапазоне безопасных символов URL и используя стандартное %xx шестнадцатеричное кодирование URL для символов вне этого диапазона.

Если <mediatype> опущен, значение по умолчанию – text/plain; charset=US-ASCII. Для краткости можно опустить «text/plain», оставив параметр charset. Схема data: url не поддерживает относительные формы URL.

Небольшой пример:

```
data:text/html;base64,dGVzdCE=
```

Вставив эту запись в строку адреса, вы увидите на странице «test!».

Кстати, это можно использовать для быстрого раскоди-

рования данных. Допустим, вам попала строка в base64-кодировке и вам надо ее расшифровать. Можно прибегнуть к помощи сторонних приложений, либо, используя язык программирования (тот же PHP, например), написать расшифровщик. А можно просто вставить эту строку в браузер и сразу получить результат.

Данная схема позволяет кодировать не только текстовую информацию, но и графическую. Например, взгляните на приведенный кусок текста:

```

```

Достаточно вставить его в HTML-страницу либо скопировать ссылку в строку адреса, и вы увидите картинку. Таким образом можно вставлять не только картинки и тексты, но даже аудио-файлы и прочее.

Полный тест и описание возможностей работы Firefox со схемой «data:url» можно посмотреть на сайте mozilla.org по адресу: <http://www.mozilla.org/quality/networking/testing/datatests.html>.

Схема «file:///» позволяет использовать Огнелиса как проводник по директориям на компьютере.

Схема «resource:///» открывает каталог, в котором находятся все файлы Firefox.

Интересная схема «view-source:». Она позволяет просматривать исходный код документа.

Например:

```
view-source:http://www.samag.ru
```

Кстати, интересная мысль создавать закладки сразу на сайт, используя данную схему.

Следующая схема – это about. О ней мы поговорим более подробно, так как она представляет наибольший интерес из всех перечисленных.

About расскажет все

Данная схема позволяет управлять настройками браузера и получать системную информацию. Например, about:cache – информация о содержимом кэша браузера (см. **рис. 1**).

Firefox расскажет, что он сохранил в файловый кэш, если набрать в адресной строке (см. **рис. 2**):

```
about:cache?device=disk
```

Вообще любые страницы из кэша можно серфить, просто выбрав в меню «Файл → Работать автономно».

А если вы хотите узнать, что в настоящий момент находится в памяти процесса, то надо ввести в адресной строке (см. **рис. 3**):

```
about:cache?device=memory
```

Если набрать:

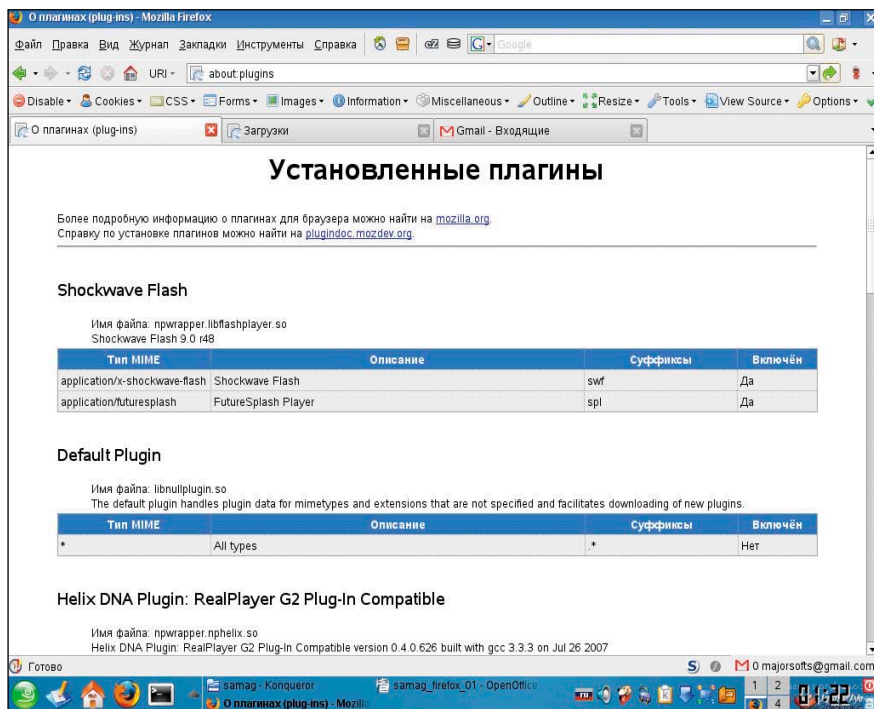


Рисунок 4. Информация об установленных плагилах

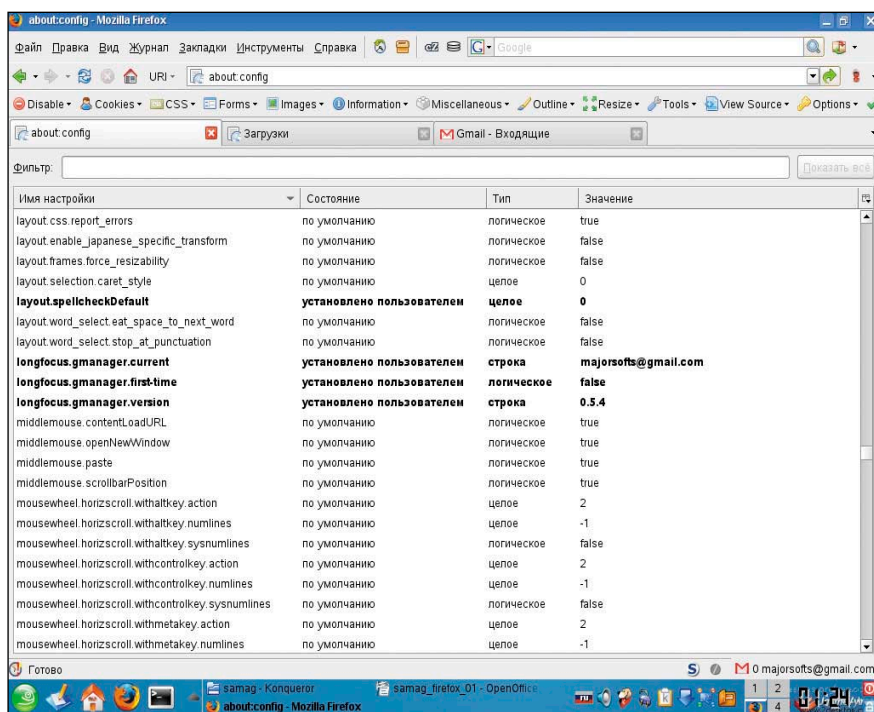


Рисунок 5. Системный реестр браузера

- **about:** – браузер расскажет о своей версии;
- **about:blank** – откроет пустую страницу;
- **about:buildconfig** – информация о том, с какими опциями скомпилирована Mozilla;
- **about:mozilla** – цитата из Книги Мозиллы;
- **about:plugins** – информация обо всех установленных плагилах (см. рис. 4);

- **about:config** – самая полезная ссылка, доступ к локальному реестру браузера, позволяющая настраивать почти все, что только есть в Firefox.

Тонкая настройка производительности

Вы считаете, что Firefox слишком прожорлив? На самом деле это не так. А если вас все-таки что-то не устраи-

вает в поведении браузера, то вы можете сделать тонкую настройку под себя. Для этого надо внести коррективы в некоторые настройки, которые скрыты от глаз обычного пользователя. Они находятся в системном реестре, который становится доступным, если набрать в адресной строке: **about:config** (см. рис. 5).

Браузер может работать намного быстрее, если изменить параметр **browser.sessionhistory.max_entries**, с помощью которого задается максимальное количество сайтов, загружаемых Firefox в свою память. Такие сайты моментально открываются прямо из памяти, если мы переходим на них вновь (например, нажимаем кнопку «Назад» или «Вперед»). По умолчанию этот параметр равен 50.

Только представьте себе. Пятьдесят страниц постоянно висят в памяти без особой необходимости. Если у вас ограничена память на компьютере, просто уменьшите этот параметр до пяти – десяти и наслаждайтесь результатом.

Если вы считаете, что Огнелис занимает много места в ОЗУ, то вы можете ограничить максимальное количество памяти, которое Огнелис может использовать. Для этого надо изменить параметр **browser.cache.memory.capacity**. Оптимальную величину этого параметра нужно будет подобрать самому.

Когда вы сворачиваете браузер, процесс по-прежнему занимает ресурсы системы. А зачем процессу занимать десятки и даже сотни мегабайт оперативки, если окно свернуто? Действительно, незачем. Это дело можно поправить с помощью ключа **config.trim_on_minimize**. Если выставить этот параметр в логическое **true**, то процесс свернутого браузера можно засвопить на жесткий диск. После этого в оперативке останется около 8-10 Мб.

В списке параметров «**about:config**» некоторые важные опции по умолчанию отсутствуют, в том числе и названный параметр. Поэтому придется его создавать самостоятельно.

Для этого надо кликнуть правой кнопкой мыши по списку параметров. В меню выбрать пункт «Создать → Логическое». Задайте имя названного параметра и укажите **true**.

Создание поисковых плагинов

Выше мы уже упоминали про поисковую форму и поисковые плагины. Теперь рассмотрим, как их делать самому. Принцип создания до безобразия прост, и не составляет особого труда написать свой, для своего поисковика. Допустим, вы захотели сделать поисковый плагин для своего сайта. Файлы этих плагинов размещаются в директории `~/firefox/searchplugins`. Они представляют из себя обычный XML-файл. Огнелис поддерживает 2 формата поисковых плагинов. Это файлы с расширением `src` и `xml`.

Плагины типа `src` были в 1-й версии браузера, но так же поддерживаются во 2-й. Это не просто XML-формат, а свой внутренний стандарт, поддерживаемый Firefox. Поэтому не стоит удивляться, что он не является валидным XML-файлом.

Для примера создадим плагин для поисковой системы Яндекс. У меня в стандартной поставке такого плагина не было. Создаем файл `yandex.src` следующего содержания:

```
# Так задаются комментарии, если вы хотите записать
# какую-то информацию, например, имя автора и как с вами
# связаться
#
# Раздел описания поискового механизма
<SEARCH
  version = "7.1"
  name="Yandex"
  action="http://www.yandex.ru/yandsearch"
  method="GET"
  description="Yandex search engine special for SaMag.ru">
  <input name="text" user>
  <input name="rpt" value="rad">
</search>

# Раздел описания автоматического обновления плагина
#
<browser
  Update="http://samag.ru/majorsoft/yandex.src"
  updateIcon="http://samag.ru/majorsoft/yandex.png"
  updateCheckDays="14"
>
```

Теперь немного комментариев к данному коду. Файл делится на два раздела — «SEARCH» и «BROWSER». Раздел «SEARCH» отвечает за процесс поиска и обработки запроса, а раздел «BROWSER» содержит параметры автоматического обновления. Пояснять подробно смысл каждой строчки не вижу целесообразным. Тут и так все понятно.

Скажу только, что пользовательский текст из формы ввода будет добавляться в переменную, в которой объявлена вместо статического значения `value` динамическая переменная `user`. Эта строка в коде выделена красным цветом.

Для того чтобы присвоить иконку данному плагину, достаточно положить файл с точно таким же именем. В данном случае `yandex.png`. Формат иконки может быть: `png`, `gif`. Размер иконки ограничен: 16x16 пикселей. После перезапуска браузера вы сможете выбрать ваш поисковый плагин и пользоваться им (см. рис. 6).

Во 2-й версии Огнелиса появился новый формат поисковых расширений. Он подчиняется всем правилам XML. Сейчас я вам покажу пример. Создадим поисковую форму для журнала «Системный администратор». Все в той же директории `searchplugins` создаем файл `samag.xml`. Код данного файла:

```
<!-- Так задается комментарий в файлах такого типа -->
<SearchPlugin xmlns="http://www.mozilla.org/2006/
  browser/search/">
  <ShortName>SaMag</ShortName>
  <Description>System administrator search
    form</Description>
  <InputEncoding>windows-1251</InputEncoding>
  <Image width="16" height="16">
    <!-- Место для иконки --></Image>
  <Url type="text/html" method="GET"
    template="http://www.google.ru/custom">
    <Param name="q" value="{searchTerms}" />
    <Param name="ie" value="windows-1251" />
    <Param name="oe" value="windows-1251" />
    <Param name="sitesearch" value="www.samag.ru" />
    <Param name="complete" value="1" />
    <Param name="hl" value="ru" />
    <Param name="inlang" value="ru" />
    <Param name="newwindow" value="1" />
    <Param name="client" value="pub-1582709506696353" />
    <Param name="cof" value="FORID%3A%3BGL%3A%3BS%
      3Ahttp%3A%2F%2Fwww.samag.ru%3BL%3Ahttp%
      3A%2F%2Fwww.samag.ru%2Fimg%2Flogob.gif%
      3BLH%3A50%3BLW%3A22%3BLBGC%3AFFFFFF%
      3BLC%3A%230000ff%3BVL%3A%23663399%
      3BGFNT%3A%230000ff%3BGIMP%3A%230000ff%
      3BDIV%3A%23336699%3B" />
    <Param name="domains" value="www.samag.ru" />
  </Url>
  <SearchForm>http://www.google.ru/custom</SearchForm>
</SearchPlugin>
```

Поисковая форма журнала организована при помощи сервиса google. В отличие от предыдущего варианта здесь все переменные задаются в формате `{varName}`. Как видите, поисковый запрос подставляется в параметр «q» из переменной `{searchTerms}`. Вообще для своих нужд можно использовать любую системную переменную. Например, вы хотите собирать статистику о версии браузера и использовать язык.

```
<Param name="browser" value="{moz:distributionID}" />
<Param name="language" value="{moz:locale}" />
```

Вы обратили внимание на комментарий «Место для иконки»? В данном файле иконку можно указать через `url`, используя одну из поддерживаемых схем браузера.

Если вы хотите загружать иконку с сервера, то можно написать:

```
<Image width="16" height="16">
  http://samag.ru/samag.png</Image>
```

Если хотите указать иконку на диске, можете воспользоваться схемой «file:///» или «resource:///», что предпочтительнее.

Но зачем куда-то класть иконку, тем самым плодить файлы? Мы уже рассматривали выше схему `data`. Возьмем нашу иконку и закодируем форматом `base64`. Я, например, для этого использовал PHP.

После чего мы просто вставляем нашу иконку прямо в файл:

```
<Image width="16" height="16">data:image/x-icon;base64,
iVBORw0KGgoAAAANSUUhEUgAAABAAAA
...
AAwDafE3pE1ITxQAAAABJRUS5ErkJggg==
</Image>
```

теперь наш плагин содержит все в себе. Перезапускаем браузер и пользуемся нашим новым поисковым плагином.

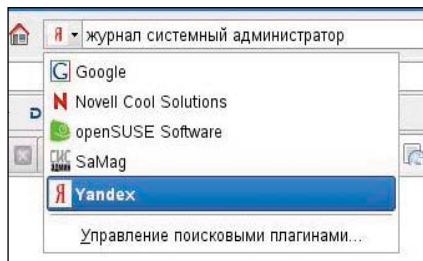


Рисунок 6. Наш плагин для поиска Яндекса



Рисунок 7. Работа Ajax в нашей поисковой форме

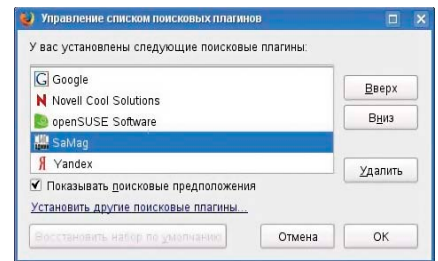


Рисунок 8. Мастер управления поисковыми расширениями

Вроде бы на этом можно было остановиться, но я обещал показать, как можно оживить поисковый плагин. Сейчас уже никого не удивишь выпадающим списком предполагаемого поискового запроса, который реализуется при помощи технологии Ajax.

На том же поисковом сервисе Google реализована такая возможность. А как было бы здорово реализовать ее в своем поисковом плагине.

Представим себе, что у вас есть свой сайт, на котором много информации и у вас есть свой поисковый механизм. Вы пишете свой поисковый плагин и отдаете пользователям. Ваш плагин может предлагать пользователю, по мере набора текста, варианты искомого запроса, на основе данных, имеющихся в вашей базе. Делается все очень просто.

Плагин будет обмениваться с вашим скриптом на сервере посредством формата JSON (англ. JavaScript Object Notation) – это текстовый формат обмена данными, основанный на JavaScript. Подробнее о данном формате можете прочитать здесь: <http://ru.wikipedia.org/wiki/JSON>.

Я не буду рассказывать, как писать скрипт. Это может сделать любой программист на любом языке. Опишу вкратце примерный алгоритм работы.

Ваш скрипт принимает некий параметр query и делает поиск по базе. На основе этого поиска формируется список предполагаемых слов. На выход скрипт «выплевывает» этот список в формате JSON.

Пример строки запроса:

```
http://your_host/firefox_search_json.php?query=системный
```

Пример того, что может вывести в данном случае ваш скрипт:

```
["системный", "системный администратор", "системный блок", "системный анализ", "системный аналитик", "системный интегратор"]
```

С серверной частью разобрались. Теперь рассмотрим клиентскую часть. Добавляем в код нашего плагина следующую строку:

```
<Url type="application/x-suggestions+json" method="GET" template="http://your_host/firefox_search_json.php?query={searchTerms}"/>
```

Собственно, все. Теперь вы можете набрать пару символов и задержаться на секунду. Поисковая форма предложит вам варианты для запроса. В браузере они называются «предположения».

Кстати, эта опция настраиваемая. Вы можете отключить «предположения». Для этого щелкните на пиктограмме те-

кущего плагина. В выпадающем списке в самом низу будет пункт «Управление поисковыми плагинами». Щелкаете по нему и попадете в редактор поисковых форм.

Идем дальше.

Апгрейдем свой поисковый сервис

Мы разобрали подробно процесс создания поисковых расширений, их возможности и преимущества. И все вроде бы неплохо, и можно было бы уже закругляться. Но я не смогу удержаться, чтобы не рассказать вам, как можно распространять ваши поисковые плагины.

Представим себе, что у вас очень хороший форум с массой полезной информации. Ваш ресурс посещает большое число людей, и вы решили сделать для своих постоянных посетителей удобную поисковую форму для вашего форума. Вы выкладываете его в виде файла, пишете инструкцию по установке, и на этом как бы все.

Но не каждый пользователь захочет ставить такой плагин руками. На самом деле ваш сайт может сообщать браузеру, что у него есть поисковый сервис и он готов предложить расширение. Чтобы Огнелис смог узнать о вашем плагине, надо добавить в заголовок страницы следующую информацию:

```
<head id="pageHead" profile="http://a9.com/-/spec/ opensearch/1.1/">
<title>Мой поисковый сервис</title>
<link href="http://your mega service ru/search.xml" title="SaMag search" rel="search" type="application/opensearchdescription+xml"/>
</head>
```

Здесь мы сообщаем браузеру, что мы можем предложить поисковую форму. Формат такого плагина описывается профайлом формата Opensearch. Далее мы указали ссылку на наш файл с описанием поисковой формы. Сам файл search.xml представлен ниже:

```
<?xml version="1.0" encoding="UTF-8"?>
<OpenSearchDescription xmlns="http://a9.com/-/spec/ opensearch/1.1/">
<Url type="text/html" template="http://your mega service ru/?s={searchTerms}"/>
<ShortName>Найдишка</ShortName>
<LongName>Супер поиск по моему форуму</LongName>
<Description>Описалово...</Description>
<Image height="16" width="16">
<!-- Место для иконки --></Image>
<Developer>Александр Майоров</Developer>
<Contact>majorsoft@majorsoft.ru</Contact>
<Attribution>&copy; 2008, Majorsoft.ru, All Rights Reserved.</Attribution>
<OutputEncoding>UTF-8</OutputEncoding>
<InputEncoding>UTF-8</InputEncoding>
</OpenSearchDescription>
```

Я не знаю, что тут добавить, так как этот плагин точно такой же, как мы описывали выше. Разве что здесь обязательно надо указать заголовок XML-файла. Все остальное, что было описано выше, применимо к данному расширению. Теперь, когда пользователь заходит на ваш портал, браузер опознает ваш поисковый плагин. Предложение установить расширение будет светиться в выпадающем списке, где находятся ваши поисковые формы.

Кстати, возле пункта меню «Добавить "Название вашего поисковика"» будет иконка сайта (favicon.ico), а не плагина, так что не удивляйтесь.

Есть еще один способ добавить поисковую форму через Jscript. Напишем небольшую функцию.

```
<script language="javascript" type="text/javascript">
function AddFFSearchPlugin()
{
    //Префикс, чтобы сократить запись
    var $= "http://www.majorsoft.ru/MajorsoftSearch";
    if (
        (typeof window.sidebar.addSearchEngine == "function") &&
        (typeof window.sidebar == "object"))
    ){
        // Ссылка на описание поисковой формы
        var file = $+ ".src";
        var icon = $+ ".png"; // Ссылка на иконку

        window.sidebar.addSearchEngine(file,icon, "MajorSoft search","FFPlugin");
    }
    else alert("Ваш браузер не поддерживает данный формат!");
}
</script>
```

Теперь достаточно вызвать эту функцию и браузер предложит установить плагин.

```
<input onclick="AddFFSearchPlugin()" type="button" value="Установить плагин"/>
```

В данном примере показано, как установить плагины формата SRC. Если вы хотите распространять расширения формата XML, то они добавляются точно так же. Разницы никакой нет. Если вы «зашили» иконку внутрь файла поисковой формы, то просто оставьте переменную icon пустой.

Думаю, хватит. Теперь вы знаете почти все о поисковых расширениях браузера Firefox. По крайней мере все необходимое.

Добавить в избранное или не добавит?

Напоследок расскажу о наболевшем. Вы часто встречали сайты, на которых предлагалось добавить страничку в закладки одним кликом по ссылке. Но почему-то все сайты, как один, знают только один способ добавления и только под один браузер. Да, да, это всем известная строчка на JavaScript, чтобы добавить ресурс в избранное Internet Explorer.

```
<a href="#" onClick="window.external.AddFavorite('http://www.samag.ru/', 'Страничка') ">В закладки</a>
```

Что за дискриминация? Более того, такой ужас встречается даже на известных посещаемых ресурсах. И если пользователь зашел браузером Firefox или Opera, то эта ссылка все равно висит на сайте и, естественно, не работает под эти браузеры. Это не только некрасиво, но портит юзаби-

лити, так как сайт становится не кроссбраузерным автоматически.

Добавить в закладки страницу через ссылку можно и Firefox, и Opera. И способ один и тот же как для первого так и для второго. А главное, нам не понадобится JavaScript! Нам нужно всего лишь сформировать правильную ссылку:

```
<a href="http://www.samag.ru/" rel="sidebar" title="Страничка" target="general">В закладки</a>;
```

По нажатию данной ссылки будет предложено добавить страницу в закладки. Все очень просто. Отдавать каждому браузеру свой формат ссылки не составляет труда, и это может сделать любой хоть мало-мальски знакомый с программированием человек.

Заключение

На этом можно поставить точку. Очень много осталось за рамками статьи, но описать все не представляется возможным в рамках одной статьи. По крайней мере вы узнали немало интересных фактов о браузере, а также некоторые интересные возможности, которые можете применить у себя на сайте. Теперь вы знаете, где и что можно подкрутить в браузере, чтобы настроить его под себя.

Благодаря таким расширениям, как WebDeveloper и FireBug, этот браузер стал стандартным инструментом почти каждого веб-разработчика. Большое количество расширений обусловлено тем, что их пишут энтузиасты. Это и хорошо, и плохо одновременно.

Хорошо, так как существует огромное количество плагинов почти под все нужды самых требовательных пользователей.

Плохо, так как плагин пишется по желанию, и однажды у автора может пропасть желание его обновлять и поддерживать.

Хотя если плагин стоящий, то всегда найдутся последователи, которые его переписут и возьмут под свою опеку. А расширяемость у Огнелиса действительно колоссальная. Некоторые дополнения тянут на отдельные самостоятельные программы.

Программистам стоит посмотреть, например, на расширение SQLite Manager. Это целое самостоятельное приложение, написанное на XUL (XUL – язык разметки для создания пользовательских интерфейсов на основе XML, на нем как раз и написаны все расширения Firefox) и использующее движок Mozilla. С помощью его можно визуально редактировать и управлять базой SQLite. На сайте <https://addons.mozilla.org/ru/firefox/browse/type:1> вы найдете любое дополнение по вашему вкусу. Это всевозможные почтовые менеджеры, оффлайн-браузеры, даунлоад-менеджеры и еще масса полезных вещей, которые превратят ваш браузер в централизованную рабочую среду, заменяющую множество сторонних приложений. Собственно, на этом все. 🌐

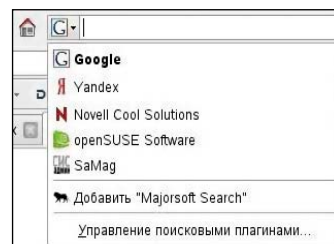


Рисунок 9. Добавление поискового расширения с сайта