Левшин Николай Сергеевич
аспирант
ФГБОУ ВО «Донской государственный
технический университет»
г. Ростов-на-Дону, Ростовская область

ПРИМЕНЕНИЕ ГОТОВЫХ РЕШЕНИЙ В ПРОЕКТИРОВАНИИ ВЕБ-ИНТЕРФЕЙСОВ: ОСОБЕННОСТИ, ПРОБЛЕМЫ И ПЕРСПЕКТИВЫ

***Аннотация**: как отмечает автор, в реалиях современного рынка веб-разработки владельцы электронных ресурсов могут и не подозревать о том, что на принадлежащих им проектах присутствует код, ухудшающий ключевые показатели работы их интерфейсов (скорость загрузки страниц, безопасность для посетителей, позиции в результатах поисковой выдачи). Наиболее распространенной причиной появления такого рода проблем становится использование, так называемых, «готовых решений», т.е. кода, разработанного в рамках других проектов.*

***Ключевые слова**: веб-интерфейс, веб-проект, проектирование веб-интерфейсов, application programming interface.*

Большая часть проектов в сети Интернет использует однотипные решения для организации вывода информационного наполнения: вывод статей в виде списка новостей, гостевая книга, система комментариев, форум и другие. Поэтому, в большинстве случаев человек, занимающийся проектированием веб-интерфейса, может позволить себе не создавать код с нуля, а воспользоваться готовыми решениями, написанными другими разработчиками.

Если речь идет о сайте, функционирующем на основе одной из популярных «систем управления контентом» (далее – CMS, от англ. «content management system»), то существует вероятность, что код наиболее востребованных у пользователей функций уже реализован в этой системе её разработчиками.

Когда какая-то популярная CMS изначально не поддерживает ту или иную функцию, для нее, как правило, можно найти различные надстройки от сторонних разработчиков, призванные облегчить создание мультиязычных проектов на

Content is licensed under the Creative Commons Attribution 4.0 license (CC-BY 4.0)

ее основе. Такие готовые решения, в русскоязычном сегменте веб-разработки, традиционно называют плагинами (от англ. plug in – подключать) или расширениями (т.к. они расширяют базовые возможности системы).

В зависимости от преследуемых их авторами целей, эти готовые решения могут оказаться как платными, так и бесплатными. И те и другие, при этом, имеют одинаковый набор недостатков. И, ключевым из этих недостатков является возможность предумышленного внедрения вредоносного кода. С помощью такого кода злоумышленник может получить полный или частичный контроль над ресурсом и использовать его в самых разнообразных целях, наиболее распространенные из которых:

– шантаж владельцев сайта угрозами нарушения работоспособности сайта;

– перенаправление посетителей сайта на мошеннические ресурсы (может привести как к ухудшению позиций «зараженного» ресурса в поисковых системах, так и его полному исключению из их индекса и блокировке «зараженного» ресурса всеми популярными браузерами).

Другую проблему использования сторонних расширений может представлять несоблюдение их разработчиками требований логики и архитектуры тех систем, для которых они их разрабатывают. В качестве наглядного примера здесь можно привести пример из практики автора, имевший место в 2013 году, когда у одной небольшой компании возникла необходимость переноса корпоративного сайта-визитки с одного сервера на другой.

Сайт был построен на основе CMS «Joomla 2.5», для нормальной работы которой необходимо, чтобы сервером поддерживались функции языка PHP, появившиеся начиная с версии 5.2.4. Однако, после переноса сайта на другой сервер, с поддержкой более поздней версии PHP (5.4), большая часть страниц этого сайта просто перестала отображаться.

В процессе поиска источника проблемы было установлено, что перестали отображаться страницы, обрабатываемые сторонним расширением, предназначенным для вывода блока социальных сетей и онлайн-закладок. В коде этого ре-

Содержимое доступно по лицензии Creative Commons Attribution 4.0 license (CC-BY 4.0)

шения использовалась функция передачи переменных по ссылке, поддержка которой была окончательно прекращена в языке PHP версии 5.4. Что и приводило к возникновению проблемы при обработке страниц сайта использующих это расширение, несмотря на то, что сама CMS на новом сервере продолжала работать без сбоев [2, с. 161].

Здесь важно отметить тот факт, что установка расширения не требовала каких-либо познаний в программировании. Поэтому менеджер, занимавшийся информационным наполнением сайта, смог произвести ее самостоятельно. А вот последующее выявление и устранение проблемы, вызванной использованием этого расширения, повлекло за собой дополнительные издержки в виде оплаты услуг программиста.

Еще одной проблемой, которую, несомненно, следует отнести к недостаткам использования расширений для CMS, является то, что готовые решения этого типа неизбежно добавляют в процесс загрузки каждой страницы сайта ряд лишних операций. Например, операцию по указанию места на интернет-странце, в котором будет отображаться интегрируемый расширением переключатель локализаций. Таким образом, использование расширений может создавать дополнительную аппаратную нагрузку на сервер и приводить к увеличению времени загрузки страниц электронного ресурса.

Также следует упомянуть о существовании универсальных готовых решений, которые, в теории, могут быть интегрированы в любую CMS. В качестве примера такого решения можно привести проект «SLI» (аббревиатура от англ. Site Language Injection – прим. пер. как «языковая инъекция для сайта») [1, с. 222].

Но, помимо того, что решения такого типа в равной степени подвержены всем вышеперечисленным недостаткам, нужно учитывать и то, что их интеграция потребует более существенных затрат и усилий, нежели установка и настройка расширения, изначально разрабатывавшегося под конкретную систему.

Content is licensed under the Creative Commons Attribution 4.0 license (CC-BY 4.0)

Нельзя не упомянуть и о решениях, распространяемых по принципу арендных бизнес-моделей, например, SaaS (англ. Software as a Service – программное обеспечение как услуга). В качестве примера такого готового решения, применимого для организации веб-проектов, можно привести проект «CartEnergy» [1, с. 223], предлагающий SaaS-платформу для электронной коммерции.

Бизнес-модель SaaS подразумевает, что обслуживание, доработка и обновление программной составляющей является обязанностью стороны предоставляющей услугу. На практике это приводит к тому, что, отдав предпочтение решению арендного типа, владельцу приходится довольствоваться поддержкой лишь функций, которые уже входят в состав готового решения, не имея никакой возможности доработать их под нужды своего проекта или мотивировать к этому владельца SaaS-платформы.

Еще один тип готовых решений, которому следует уделить внимание, это, так называемые, интерфейсы прикладного программирования (далее – API, от англ. «application programming interface»). Это особый класс интерфейсов, ориентированных на внедрение в другие проекты. Основная идея использования API заключается в использовании готовых функций, библиотек и элементов оформления, которые по мере необходимости подгружаются непосредственно с сервера разработчика того или иного API. Другими словами, использование API позволяет веб-разработчикам «не изобретать велосипед» всякий раз, когда возникает необходимость в реализации на сайтах таких традиционно однотипных элементов как расположение офиса компании на карте города [3, с. 191] или кнопки социальных сетей.

Например, на сегодняшний день, каждый более-менее известный сервис социальных коммуникаций или онлайн-закладок предлагает веб-разработчикам собственный «фирменный» API, для установки на сайт, как минимум, одной версии кнопки, позволяющей отслеживать предпочтения пользователей этого сервиса и транслировать заинтересовавшую их информацию на страницы этого сервиса. При этом, предоставление API для внедрения кнопок может быть выгодно социальным сервисам сразу по целому ряду причин:

Содержимое доступно по лицензии Creative Commons Attribution 4.0 license (CC-BY 4.0)

– повышение узнаваемости бренда;

– обеспечение притока на страницы сервиса информационного наполнения, превращающего социальные сервисы в ленты новостей и, тем самым, делающего их более интересными для пользователей;

– возможность получения информации о предпочтениях конкретного пользователя, которую можно использовать при подборе содержимого рекламных блоков, отображаемых для этого пользователя на страницах сервиса.

Типичный код использования API для внедрения такой «фирменной» кнопки социального сервиса традиционно включает задачу на загрузку файла с основным кодом отображения кнопки. В ходе выполнения этой задачи происходит целый ряд разнообразных процессов, ключевыми из которых являются:

– поиск в сети Интернет указанного ресурса, содержащего код кнопки;

– загрузка файла с основным кодом и вспомогательных файлов (чаще всего, изображение копки и дополнительный код ее визуального оформления);

– проверка посетителя на предмет авторизации в социальном сервисе.

В зависимости от типа кнопки возможен дополнительный обмен данными с социальным сервисом, например, для получения количества нажатий кнопки, отображаемого на интегрированном в кнопку счетчике. А, если посетитель авторизован в социальной сети – передача в эту социальную сеть информации о посещенной странице, получение изображений и списка друзей из анкеты пользователя и перестройка визуального оформления кнопки в соответствии с полученными данными (например, формирование всплывающего окна с изображениями пользователей нажавших кнопку).

Все вышеперечисленные процессы занимают определенное время и могут существенно замедлять загрузку интернет-страниц, особенно в тех случаях, когда речь идет о единовременном подключении нескольких кнопок от различных социальных сервисов. Это, может вызвать как рост недовольства со стороны посетителей сайта, так и пессимизацию позиций сайта в результатах поисковой выдачи.

Content is licensed under the Creative Commons Attribution 4.0 license (CC-BY 4.0)

Подводя общий итог вышесказанного, можно констатировать, что готовые решения еще долгие годы будет применяться разработчиками в целях экономии ресурсозатрат. И очень важно, чтобы эти разработчики понимали, что от их выбора и качества внедрения готовых решений зависит будущее их проектов.

Список литературы

1. Левшин Н.С. Актуальные проблемы проектирования и продвижения веб-проектов, ориентированных на представителей нескольких языковых групп [Текст] // Интернет-маркетинг. – 2016. – №4. – С. 216–230.

2. Левшин Н.С. Особенности реализации кнопок социальных сетей и онлайн-закладок при проектировании веб-интерфейсов [Текст] // Интернет-маркетинг. – 2015. – №3. – С. 142–153.

3. Себрант А. API – строительный материал для современного сайта [Текст] // Интернет-маркетинг. – №3. – 2010. – С. 190–194.

Содержимое доступно по лицензии Creative Commons Attribution 4.0 license (CC-BY 4.0)