УДК 004.9 DOI: 10.30748/soi.2017.150.15

И.Н. Егорова, А.Ю. Худолей

Харьковский национальный университет радиоэлектроники, Харьков

ИССЛЕДОВАНИЕ ВОЗМОЖНОСТЕЙ КОМПОНЕНТНОГО ПОДХОДА ПРИ РАЗРАБОТКЕ ВЕБ-САЙТОВ

В статье проведено исследование веб-компонентов, основанных на независимых стандартах Custom Elements, Shadow DOM, HTML Templates и HTML Imports. Они могут быть использованы как самостоятельные единицы или в совокупности с другими стандартами. Проведен анализ актуальных проблем и пути их решения с помощью веб-компонентов. Даны рекомендации по использованию каждого из стандартов и компонентного подхода в целом при разработке веб-сайтов.

Ключевые слова: *Custom Elements, Shadow DOM, HTML Templates и HTML Imports*.

Введение

Постановка проблемы. Стремительно развивающиеся веб-технологии делают доступными все новые on-line услуги. Сейчас, не выходя из дома, мы можем оплатить коммунальные счета, билеты, забронировать отель, заказать такси и отследить местоположение машины. Мы постоянно ищем в Интернет все новые приложения и веб-сайты, которые сделают нашу жизнь проще, тем самым экономя время. Спрос, как известно, рождает предложение. Растет число компаний, конкурирующих между собой, и соответственно растет конкуренция между их сайтами в Сети. Компании стремятся, с одной стороны, увеличить число своих пользователей, а с другой – минимизировать время, необходимое для внедрения новой функциональности. Зачастую такие задачи являются трудновыполнимыми, поскольку проще бывает написать сайт с нуля, чем его модернизировать. Существует большое количество инструментов, которые позволяют ускорить и оптимизировать процесс создания веб-сайтов, однако, остается еще ряд нерешенных проблем.

Так, слабая семантика выявляется при анализе кода любой страницы. Структура любой страницы состоит по большей части из элементов, что затрудняет определение семантики каждого из них.

Конфликты стилей возникают, как правило, когда в создании веб-сайта участвует несколько разработчиков. Каждый из них вносит изменения в css-файлы, добавляя новые элементы. Такие действия могут привести к изменению и других элементов на странице, которые будут обнаружены позднее. Таким образом, нельзя быть уверенным, что изменение одного элемента не затронет изменение другого.

Отсутствие шаблонов затрудняет разработку веб-сайтов. Зачастую бывает необходимо отобразить ряд элементов с одинаковой структурой, но различным контентом. В этом случае при отсутствии инструментов для работы с шаблонами создается скрытый блок, который будет дублироваться и наполняться необходимой информацией. Несмотря на то, что блок скрыт, в DOM дереве он присутствует, а, следовательно, увеличивается время загрузки страницы.

Отсутствие единых правил написания отдельных элементов и пользовательских компонентов также затрудняет создание веб-сайтов. Различные библиотеки и фреймворки диктуют разработчикам свои правила и условия. Все вышеперечисленные проблемы решает новый комплекс стандартов – веб-компоненты.

Анализ последних исследований и публикаций. Веб-компоненты разработаны Консорциумом Всемирной паутины (W3C). Они представляют собой набор технических требований, которые позволяют создавать новые настраиваемые, многоразовые, инкапсулированные теги HTML для использования при создании веб-страниц и веб-приложений. Пользовательские компоненты и виджеты, основанные на стандартах веб-компонентов, будут работать в современных браузерах и могут использоваться с любой библиотекой JavaScript или средой, которая работает с HTML. Веб-компоненты основаны на существующих веб-стандартах. Функции поддержки веб-компонентов в настоящее время добавляются к спецификациям HTML и DOM, позволяя веб-разработчикам легко расширять HTML с помощью новых элементов с инкапсулированным стилем и пользовательским поведением [5]. Веб-компоненты упоминаются с 2011 года и находятся в непрерывном развитии. Постоянные изменения в спецификации затрудняют нахождение актуальной информации. Что касается поддержки браузерами, то на данный момент она реализована не полностью. На рис. 1–2

показана поддержка стандартов, входящих в веб-компоненты в процентном отношении [1].

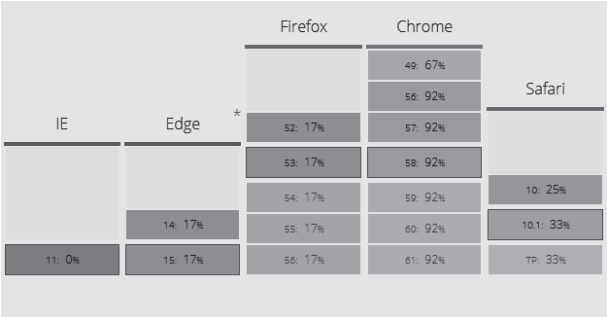

Рис. 1. Поддержка браузерами веб-компонентов (часть 1)

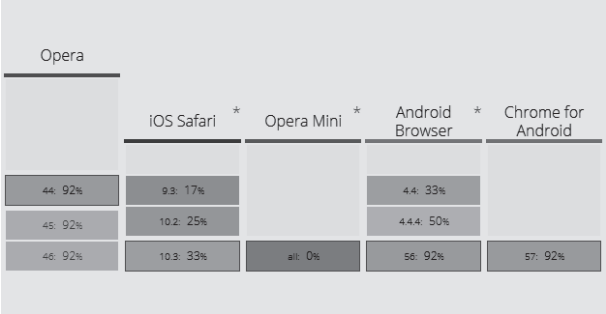

Рис. 2. Поддержка браузерами веб-компонентов (часть 2)

Не все браузеры в полной мере поддерживают этот набор стандартов, но их процент постоянно растет.

Цель статьи. Исследование новых возможностей компонентного подхода при разработке веб-сайтов.

Изложение основного материала

Веб-компоненты основываются на 4 стандартах: Custom Elements, Shadow DOM, HTML Templates и HTML Imports.

Стандарт Custom Elements позволяет создавать и использовать новые теги HTML, для которых необходимо задавать свое поведение и стили. Этот стандарт позволяет расширять уже существующие элементы, настраивая или изменяя их функции. Спецификация Custom Elements имеет 2 редакции v0 и v1. В редакции v1 появился глобальный объект customElements с тремя методами:

– CustomElements.define (name, Class [, options]) позволяет зарегистрировать новый настраиваемый элемент, предоставляя его имя, класс и, в качестве необязательного параметра, объект с ключевым словом extends, которое описывает его поведение;

– CustomElements.whenDefined (name) возвращает Promise, который будет вызываться после определения элемента. Promise решает, когда задан настраиваемый элемент. Если элемент уже определен, то немедленно разрешает его и отклоняет, если имя тега не является допустимым именем специального элемента;

– CustomElements.get (name), возвращает заданный конструктор Class для указанного имени.

Следующее изменение заключается в том, что компоненты определяются классами.

Наиболее значимым является изменение названий методов. Так, attachCallback теперь называется connectedCallback, detachedCallback называется disconnectedCallback, createdCallback теперь является конструктором, а attributeChangedCallback срабатывает только в том случае, если атрибут был определен через открытый статический массив observAttributes [2].

Для создания собственного элемента <my-element> необходимо его определить следующим образом:

class MyElement extends HTMLElement { ... }
customElements.define("my-element", MyElement);

Стандарт Shadow DOM решает проблему конфликтов стилей, позволяя инкапсулировать стилизацию и внутреннюю разметку компонента, скрывая детали реализации будущего компонента от конечного пользователя. Shadow DOM также имеет 2 редакции спецификации. Shadow DOM в редакции v0 допускал только открытый режим, который позволял получить доступ к элементам при помощи JavaScript. В редакции v1 был добавлен еще и закрытый режим, который ограничивает доступ к элементам. Одной из особенностей Shadow DOM является его связь с Light DOM, что позволяет непосредственно из Shadow DOM обратиться ко всем дочерним узлам элемента, которые находятся вне этого веб-компонента и отобразить их в любом его месте [7].

В этом стандарте также добавлен метод Element.attachShadow() вместо createShadowRoot(). Работа метода, который присоединяет Shadow DOM к указанному элементу и возвращает ссылку на его ShadowRoot осуществляется следующим образом:

```
<my-element> </my-element>
class MyElement extends HTMLElement {
  connectedCallback() {
    const shadow = this.attachShadow({mode: 'open'});
    shadow.innerHTML = `
      <style>  …  </style>
          <div><h2>Hello world!</h2></div> `;
  }
}
customElements.define("my-element", MyElement);
```

Стандарт HTML Templates решает проблему отсутствия шаблонов, определяя шаблон для нового элемента. Этот шаблон позволяет хранить некоторую разметку для страницы, которую впоследствии можно клонировать и повторно использовать. Изна-

чально шаблон скрыт и не является частью DOM, он проверяется на валидность парсером, но только при его инициализации содержимое шаблона выводится. HTML Templates имеет самую стабильную спецификацию. Для создания шаблона необходимо контент обернуть в тег <template></template>.

Использование шаблона возможно только после его активации. Предоставляется возможность сделать полную копию шаблона при помощи importNode и вывести его содержимое [4].

Шаблон может быть размещен в любом из контейнеров <head>, <body>, <frameset>, <colgroup>, а также в другом документе.

Стандарт HTML Imports позволяет импортировать одни документы в другие при помощи элемента <link rel=”import”>. В источнике src необходимо указать путь к html-документу. При этом импортируемый файл должен либо находиться на том же домене, либо передаваться со специальным http-заголовком с другого домена. Стандарт поддерживает вложенные импорты, т.е. один документ может импортироваться в другой, который в свою очередь импортируется в третий и т.д [3; 6]. Стили и скрипты выполняются в основном документе. Следует отметить, что при повторном импорте одного и того же файла используется уже импортированный файл, что избавляет от дублирования информации.

Выводы

Проведенное в работе исследование позволило определить основные возможности веб-компонентов. Несмотря на то, что браузеры не в полной мере поддерживают этот комплекс стандартов, целесообразность их использования при разработке веб-сайтов очевидна. Применение веб-компонентов позволяет решить ряд значительных проблем и сделать приложения более гибкими и легко расширяемыми без каких-либо сторонних фреймворков и библиотек.

Список литературы

1. *Can I Use Web Component [Электронный ресурс]. – [Электронные данные]. – Режим доступа: http://caniuse.com/#search=web%20component. – Загл. с экрана.*
2. *Custom Elements [Электронный ресурс]. – [Электронные данные]. – Режим доступа: https://www.w3.org/TR/2016/WD-custom-elements-20161013/. – Загл. с экрана*
3. *HTML Imports [Электронный ресурс]. –- [Электронные данные]. – Режим доступа: https://www.w3.org/TR/2016/WD-html-imports-20160225/. – Загл. с экрана.*
4. *HTML Templates [Электронный ресурс]. – [Электронные данные]. – Режим доступа: https://www.w3.org/TR/2014/NOTE-html-templates-20140318/. – Загл. с экрана*
5. *Introduction [Электронный ресурс]. – [Электронные данные]. – Режим доступа: https://www.webcomponents.org/introduction#custom-elements. – Загл. с экрана.*
6. *Jarrod Overson, Jason Strimpel, Developing Web Components / Jarrod Overson, Jason Strimpel, O’Reilly Media, Inc., 2015. – 252 p.*
7. *Shadow DOM [Электронный ресурс]. – [Электронные данные]. – Режим доступа: https://www.w3.org/TR/2017/WD-shadow-dom-20170213/. – Загл. с экрана.*

Поступила в редколлегию 10.05.2017

Рецензент: д-р техн. наук проф. А.М. Синотин, Харьковский национальный университет радиоэлектроники, Харьков.

ДОСЛІДЖЕННЯ МОЖЛИВОСТЕЙ КОМПОНЕНТНОГО ПІДХОДУ ПРИ РОЗРОБЦІ ВЕБ-САЙТІВ

І.М. Єгорова, А.Ю. Худолей

У статті проведено дослідження веб-компонентів, заснованих на незалежних стандартах Custom Elements, Shadow DOM, HTML Templates та HTML Imports. Вони можуть бути використані як самостійні одиниці або у сукупності із іншими стандартами. Проведено аналіз актуальних проблем та шляхи їх вирішення за допомогою веб-компонентів. Надані рекомендації щодо використання кожного із стандартів та компонентного підходу в цілому під час розробки веб-сайтів.

Ключові слова: *Custom Elements, Shadow DOM, HTML Templates і HTML Imports*.

RESEARCH FOR COMPONENT APPROACH IN WEBSITES DEVELOPMENT

I. Iegorova, A. Khudoley

This paper contains details of research for web-components based on independent standards Custom Elements, Shadow DOM, HTML Templates and HTML Imports. Components are used as self-contained units or in conjunction with other standards. The paper describes conducted analysis for actual problems and solutions with web-components. Recommendations are given for usage of each standard and component approach in general in websites development.

Keywords: *Custom Elements, Shadow DOM, HTML Templates and HTML Imports*.