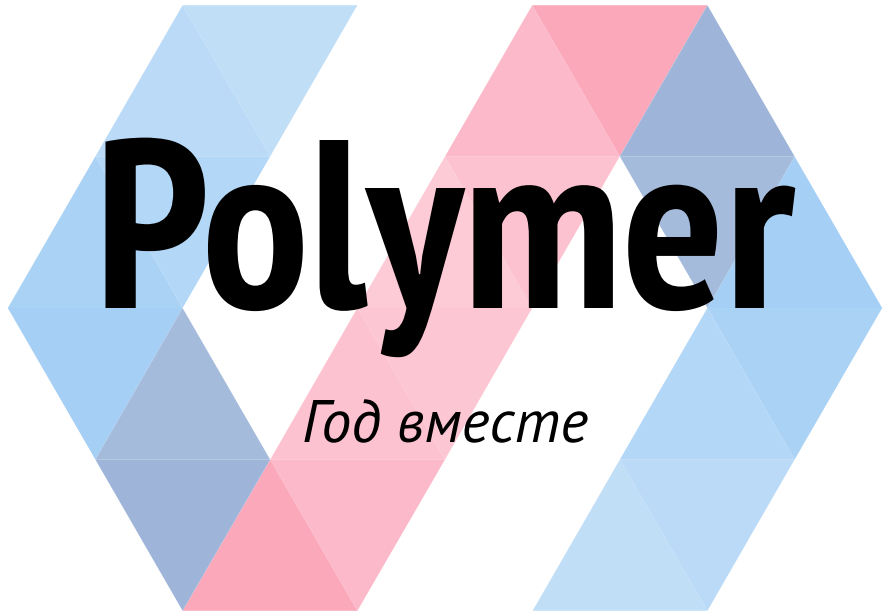



arturparkhisenko

2016-11-26, Kyiv

Веб-стандарты

livestream

# Наша миссия

Мы обеспечиваем онлайн-трансляции

---

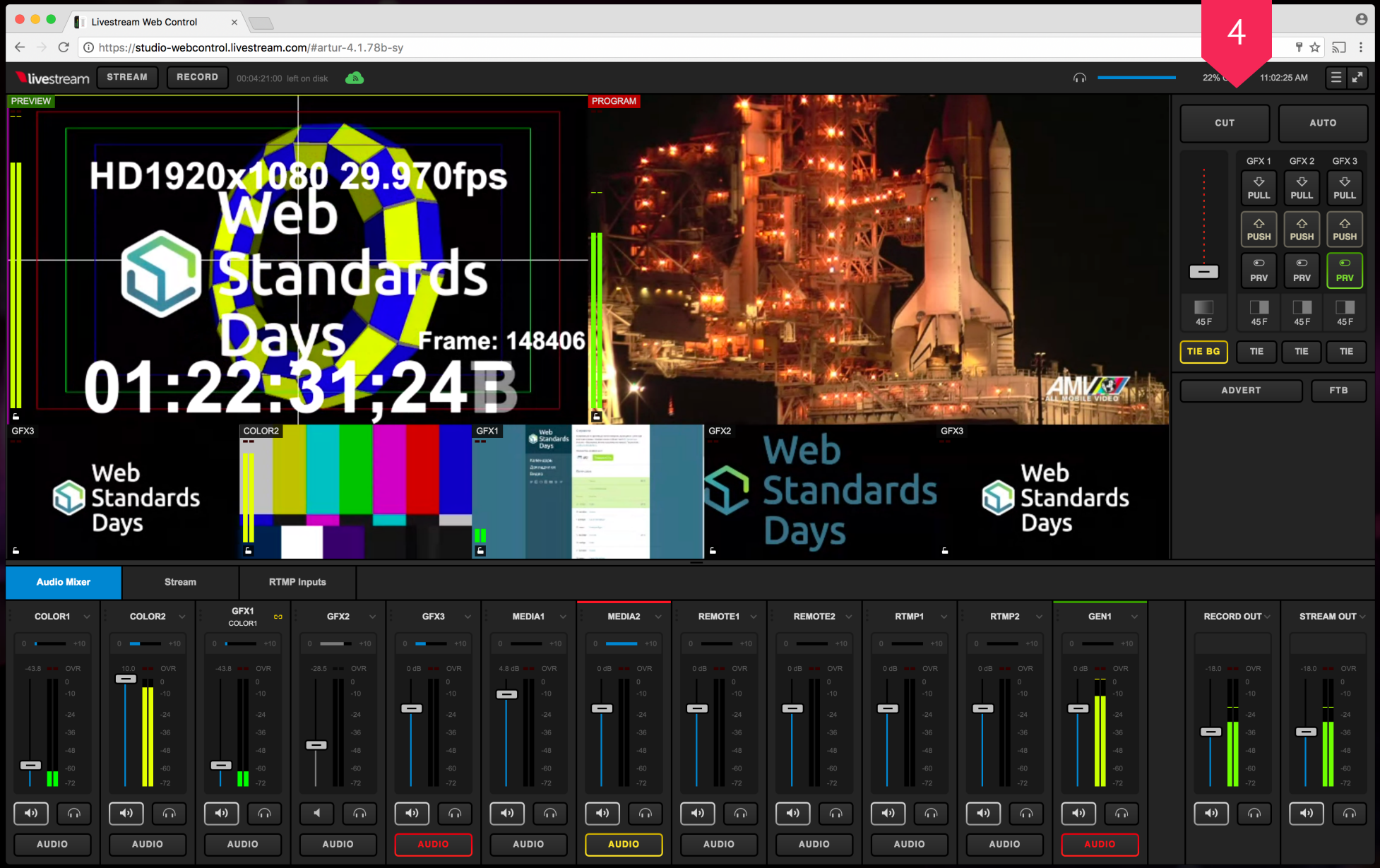
Мы сближаем тебя с окружающими, превращая каждое событие в бесценный опыт

livestream

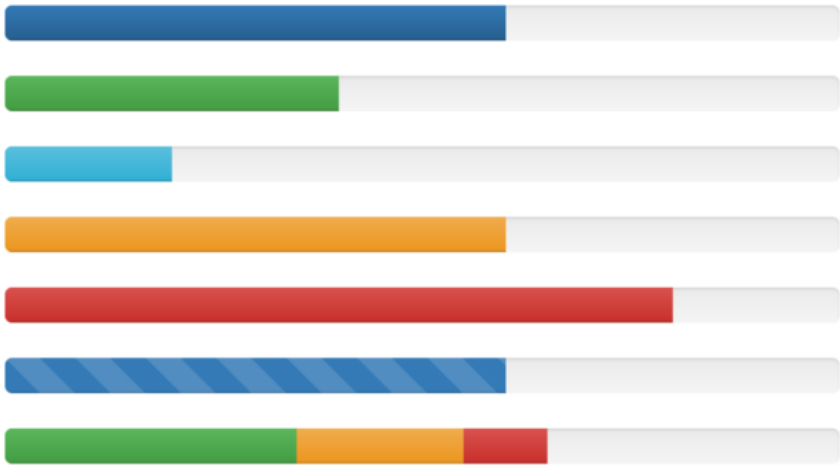
# Проект Livestream Studio Webcontrol

Часть Livestream Studio, позволяющая управлять приложением с помощью веб-браузера из любой точки мира

Используя WebRTC


Livestream Web Control
https://studio-webcontrol.livestream.com/#artur-4.1.78b-sy
4
livestream
STREAM
RECORD
00:04:21:00 left on disk
11:02:25 AM
PREVIEW
HD1920x1080 29.970fps
Web Standards Days
Frame: 148406
01:22:31;24
PROGRAM
CUT
AUTO
GFX 1
GFX 2
GFX 3
PULL
PUSH
PRV
45 F
TIE BG
TIE
ADVERT
FTB
GFX3
COLOR2
GFX1
GFX2
GFX3
Audio Mixer
Stream
RTMP Inputs
COLOR1
COLOR2
GFX1
COLOR1
GFX2
GFX3
MEDIA1
MEDIA2
REMOTE1
REMOTE2
RTMP1
RTMP2
GEN1
RECORD OUT
STREAM OUT
AUDIO

## Progress bars



## Alerts

**Well done!** You successfully read this important alert message.

**Heads up!** This alert needs your attention, but it's not super important.

**Warning!** Best check yo self, you're not looking too good.

**Oh snap!** Change a few things up and try submitting again.

## Dropdown menus

- Action
- Another action
- Something else here
- Separated link

## Input type number

1

## Please sign in

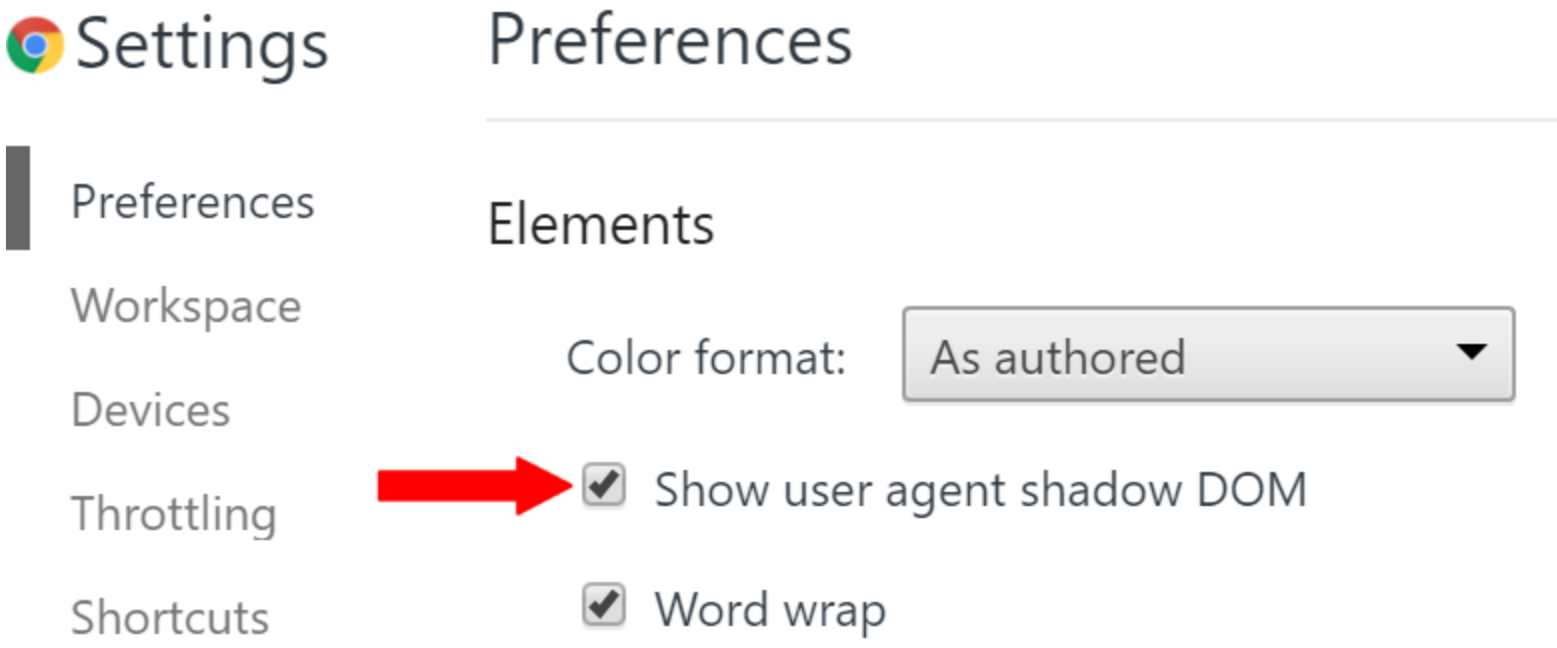
Email address

Password

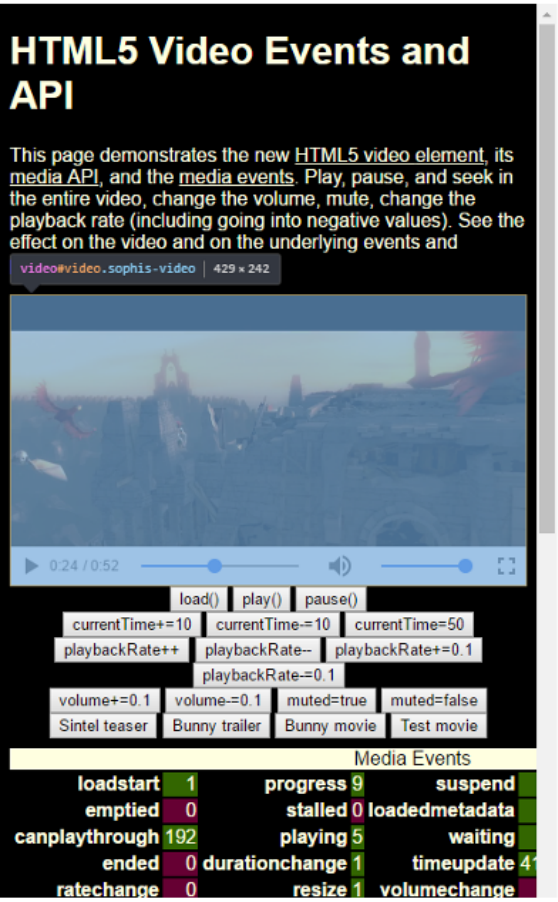
Remember me

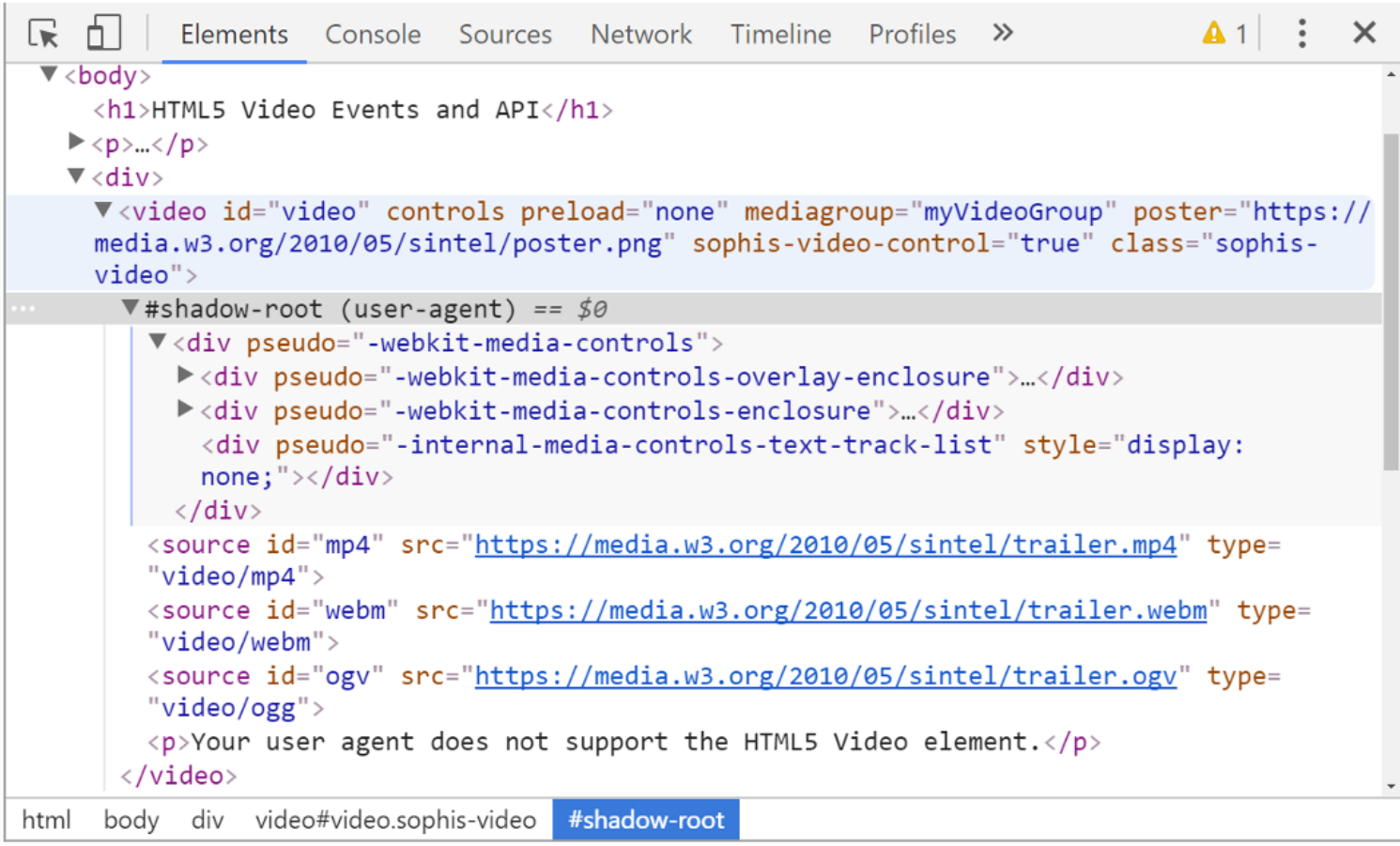
Sign in

# Как увидеть внутренности веб компонентов?



# Под "капотом" у тега <video>



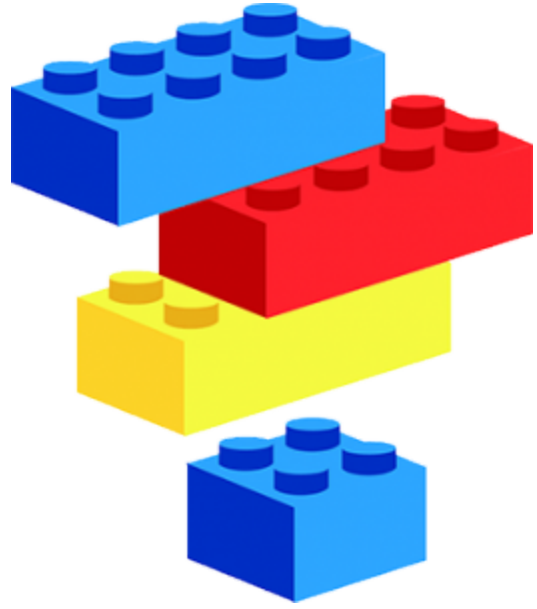



# Веб-компоненты

Это возможность создавать пользовательские, инкапсулируемые, переиспользуемые компоненты, использующие веб-платформу.

В основе веб-компонентов лежат эти спецификации:

- **Shadow Dom** - Теневой DOM
- **CSS Scoping** - Стилизация Теневого DOM
- **HTML Imports** - Импорт HTML
- **HTML Templates** - Шаблоны
- **Custom Elements** - Пользовательские элементы



# Кросс браузерные Веб-Компоненты уже здесь👍

**Addy Osmani**
@addyosmani

Custom Elements are enabled by default in today's Safari Tech Preview:
webkit.org/blog/7071/rele … 🔥

| RETWEETS | LIKES |
| --- | --- |
| 83 | 154 |

8:24 PM - 9 Nov 2016

# Поддержка браузерами

| | Chrome | Opera | Safari | Firefox | Edge |
|---|---|---|---|---|---|
| **Template** | ✅ | ✅ | ✅ | ✅ | ✅ |
| **Shadow DOM v1** | ✅ | ✅ | ✅ | 🤍 | 🤍 |
| **Custom Elements v1** | ✅ | ✅ | 🤍 | 🤍 | 🤍 |
| **HTML Imports** | ✅ | ✅ | ❗ | ❗ | ❗ |

Полифил `webcomponents.js` добавляет нам поддержку:

IE 10, IE 11+, Safari 7+, Chrome (Android), Safari (iOS 7.1+)

# Проект и библиотека Polymer

Используйте веб-платформу, чтобы сделать веб лучше

**Проект Polymer** - это набор библиотек и утилит для веб-разработчиков.

**Библиотека Polymer** создана для того, чтобы упростить работу с нативными API веб-платформы.

# Polymer утилиты

**Polymer App Toolbox** - это коллекция готовых веб-компонентов, утилит и шаблонов для создания *Прогрессивных Веб Приложений* с библиотекой Polymer (роутинг, локализация, хранилище, лэйауты, polygit-cdn).

**Отдельные утилиты:** polyserve, polylint, Web component tester, polybuild (vulcanize, crisper, polyclean).

# Polymer утилиты

**Polymer-cli** - консольная утилита, которая включает в себя инструменты сборки, генерации, линтер, сервер для разработки и платформу для тестирования.

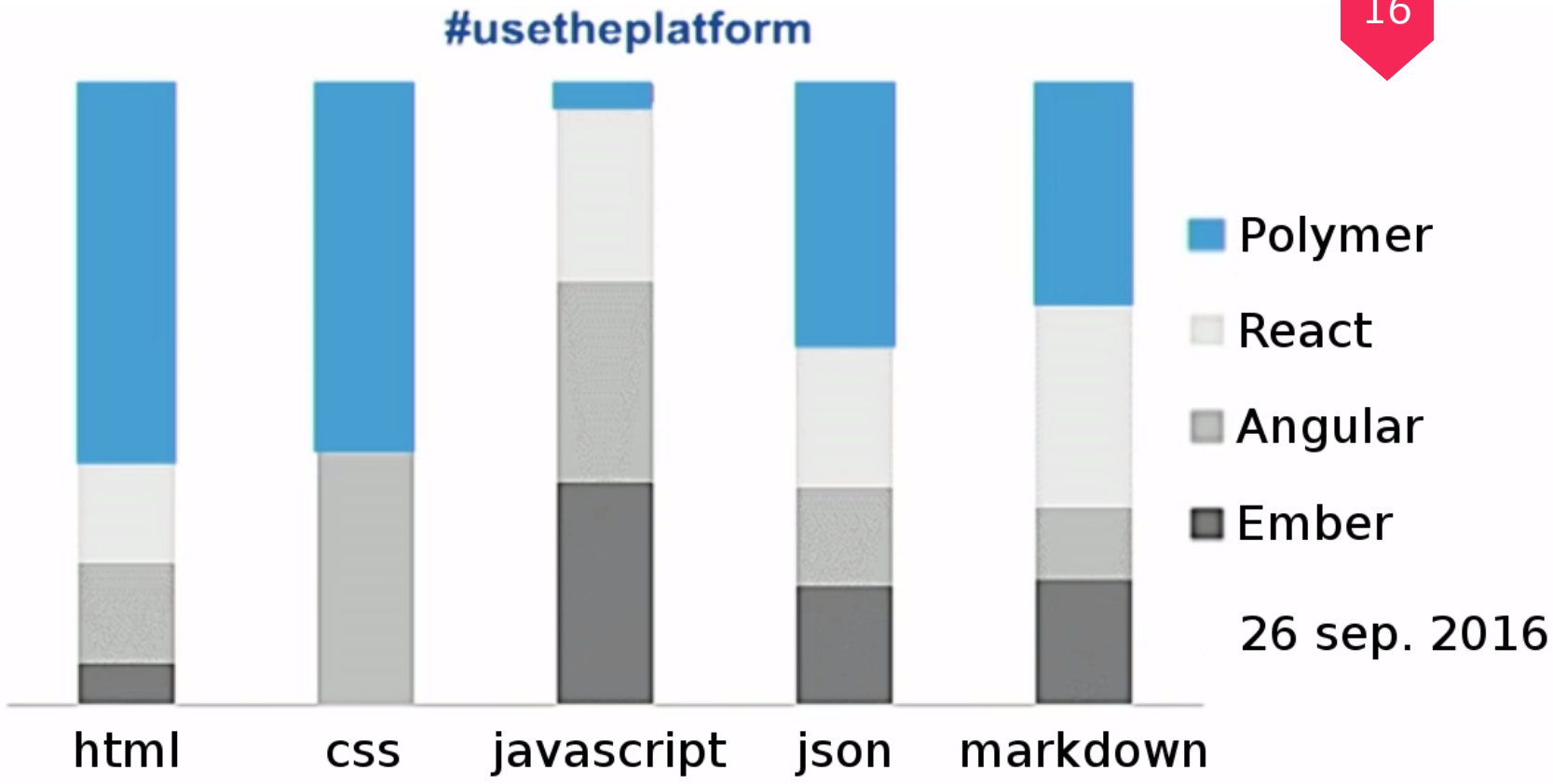
# Кто использует Polymer?

ZEPLIN

NET-A-PORTER

# Polymer подход

- Декларативный путь передачи данных
- Синхронизация атрибутов и свойств
- Меньше шаблонного кода




Polymer GitHub Lines of Code

**4x more HTML, CSS, JSON, MARKDOWN**

10x less JAVASCRIPT

Source: youtu.be/8ZTFEhPBJEE

# Сравнение с конкурентами

Polymer > всех по размеру и доступен только в Bower (v2 будет в yarn).

SkateJS

Incremental DOM, JS импорт вместо HTML импорта, живой проект

X-tag и Bosonic

Являются заброшенными :/

и т.д. не используют Web Components APIs,

но можно использовать компоненты в них, и их внутри компонентов.

# Слабые стороны Polymer 1.X (ES5)

- Низкая производительность, из-за полифилов
- "Сырые" утилиты
- Нет способа подключения внешних файлов стилей
- Много запросов без бандлинга (http2 поможет)
- Observers вызывают событие при изменении значения с `undefined` на начальное

# Слабые стороны Polymer 1.X (ES5)

- Сложности управления состоянием
- Нет *server-side* рендеринга (связано с Shadow Dom)
- Computed свойства могут просто не работать
- Проблемы с элементами `dom-if`, `dom-repeat`

еще...

# Год спустя

- Столько новых спецификаций! 😉
- Сторонние компоненты не пригодились, не подошли или не нашлись
- Не все фичи необходимо использовать и не всегда получится
- Очень разные виды элементов (страницы и макеты, их части)

# Год спустя

- Управление состоянием с помощью паттерна Медиатор и Pub/Sub
- Подход через использование атрибутов, а не вызов публичных методов
- Не используйте ShadyDom, если возможно
- Рефакторинг 0.5 —> 1.0 —> 1.7 (есть утилиты)

# Пример: простой Polymer компонент

```
01. <dom-module id="element-name">
02.   <template>
03.     <style>/* CSS rules for your element */</style>
04.     <!-- local DOM for your element -->
05.   </template>
06.   <script> Polymer({ is: "element-name" }); </script>
07. </dom-module>
```

Много примеров для Polymer 1.X

# Пример: компонент <wsd-logo>

```
01. <wsd-logo color="dark" width="128px"></wsd-logo>
02. <wsd-logo color="light" width="128px"></wsd-logo>
03. <wsd-logo color="mono" width="128px"></wsd-logo>
```

# Пример: связывание атрибутов и свойств

```
01. <template>
02.   <!-- Атрибуты -->
03.   <my-element selected$="[[value]]"></my-element>
04.   <!-- Свойства -->
05.   <my-element selected="{{value}}"></my-element>
06. </template>
```

# Пример: CSS Scoping, важные селекторы

```
<style>
  :host {} /* Текущий элемент */
  :host() {} /* Текущий элемент, соответствие селектору */
  :host-context() {} /* Родитель, соответствие селектору */
  ::slotted() {} /* Вставленный контент, старый ::content */
  >>> {} /* Игнорирует границы Shadow Dom,
    старые /deep/ и ::shadow */
</style>
```

# Пример: дополнительная изоляция с помощью CSS Containment

CSS свойство, позволяющее разработчикам ограничить область применения стилей, компоновок и отрисовок для браузера.

Влияет на производительность, используйте аккуратно.

```
01. contain: strict; /* === layout style paint size */
02. contain: content; /* === layout style paint */
```

habr пост о CSS Containment

# Эволюция Polymer

- Polymer 0.5 как эксперимент
- Polymer 1.0+ продакшн версия

## Polymer 2.0 нативный веб

- Поддержка веб-компонентов версии 1.X
- Плавный переход на новую версию (гибридный режим)
- Улучшенная совместимость с нативным вебом
- Маленький размер ~12кб (Polymer.Element)

# Принцип FIRST при разработке компонентов

При создании компонентов не забывайте про эти 5 пунктов:

1. **Focused** - легкое API, делает одну вещь и делает ее хорошо
2. **Independent** - не имеет зависимостей
3. **Reusable** - больше гибкости, меньше кода, композиция
4. **Small** - проще поддерживать и документировать, меньше ответственности
5. **Testable** - легче тестировать, более точное определение компонента, приводящего к ошибке

addyosmani.com/first

# WebComponent bundle analyzer

# Каталоги компонентов

- elements.polymer-project.org

  и новая унифицированная бета версия
- customelements.io
- expandjs.com
- vaadin.com

# Ссылки

- What's New in Shadow DOM v1 (by examples)
- Custom Elements articles: developers.google.com, webkit.org
- Google codelabs, from Polymer Summit 2016
- webcomponents/gold-standard
- Server-side Web Components
- learn.javascript.ru/webcomponents
- Useful GitHub Projects: PolymerTS, polymer-redux, ReactiveElements, WCBA

# Вопросы?

Artur Parkhisenko

www arturparkhisenko.github.io

twitter @lifeasecond

Ссылка на презентацию: arturparkhisenko.github.io/wsd-polymer-2016/