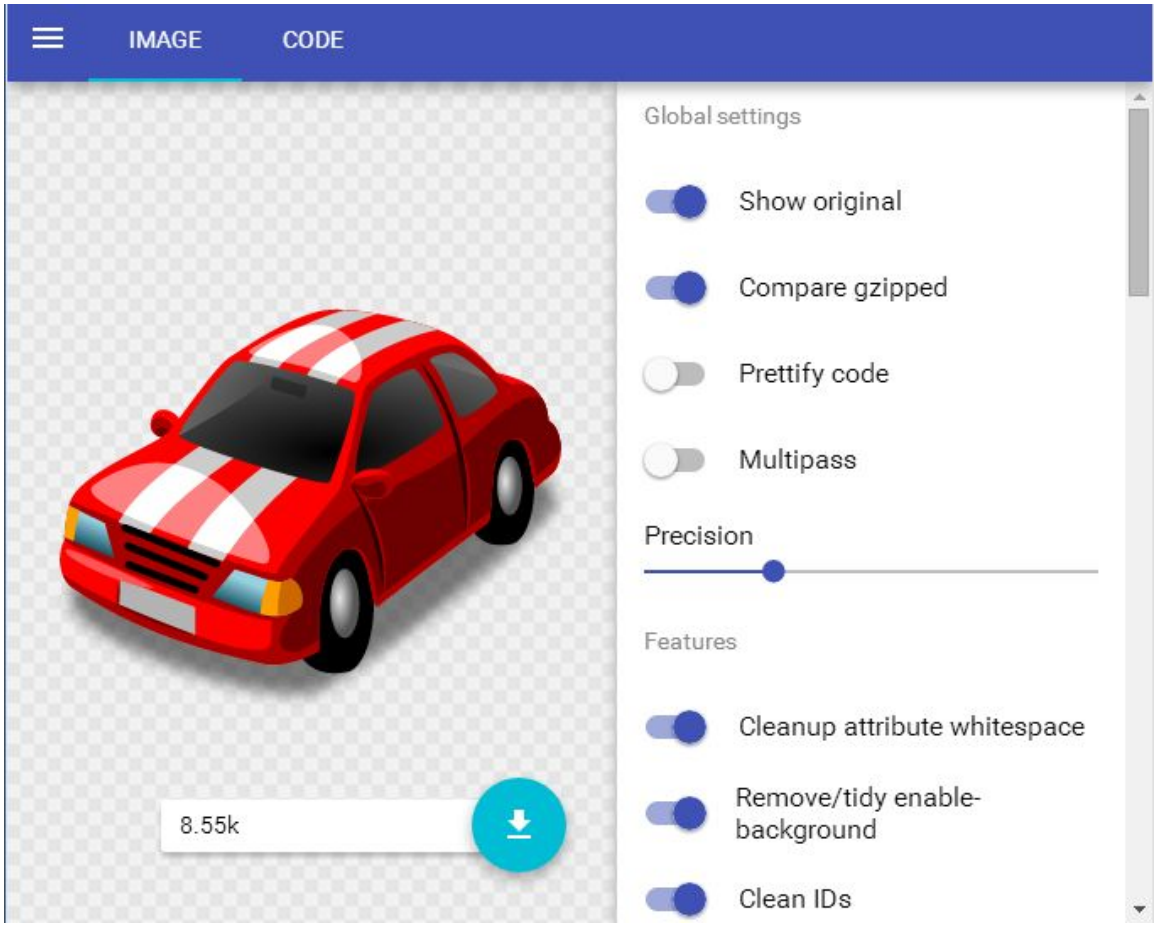
Лекція 26.

WEB. CSS фреймворки. Работа с графикой SVG – анимация, спрайты, инструменты.

Полезные инструменты для работы с SVG.

- Онлайн-инструмент для сжатия SVG: https://jakearchibald.github.io/svgomg/



Полезные инструменты для работы с SVG.

- Онлайн-инструмент для обработки и сжатия SVG: http://petercollingridge.appspot.com/svg-editor

Полезные инструменты для работы с SVG.

- Replace an <img> element with an inline SVG. https://www.npmjs.com/package/svg-inject

- Позволяет использовать преимущества не-инлайнового SVG, в то же время работая со всеми свойствами SVG используя CSS-селекторы.
  - Например: псевдокласс hover и т.д.

HTML

```
<img src="lightbulb.svg" class="svg-inject" />
```

JS

```
$('.svg-inject').svgInject();
```

SVG-анимация.

- Сферы использования:
  - Создание рекламных баннеров, объявлений.

- Возможны несколько путей для SVG-анимации:
  - Использование тега <animate> прямо в SVG коде (спецификация анимаций SMIL).
  - Использование библиотек Snap.svgИспользование библиотек Snap.svg, SVG.js и подобных.
  - Анимация с помощью CSS-свойств прямо в инлайновом SVG.

Что можно анимировать в SVG?

- Простейший случай – свойства:
  - заливка;
  - цвет, толщина границы;
  - размер изображения.

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1"
width="300px" height="300px" viewBox="0 0 300 300">
  <polygon
    fill = "#FF931E"
    stroke = "#ED1C24"
    stroke-width = "5"
    points = "279.1,160.8 195.2,193.3 174.4,280.8
117.6,211.1 27.9,218.3 76.7,142.7 42.1,59.6 129.1,82.7
197.4,24.1 202.3,114 "/>
</svg>
```

star.svg

Свойства SVG:

Их очень много!

- Такие же как и у CSS:
  - font, font-family, font-size, font-size-adjust, font-stretch, font-style, font-variant, font-weight, direction, letter-spacing, text-decoration, unincode-bidi, word-spacing, visibility, text-rendering, writing-mode, clip-path, mask-opacity, filter, pointer-events, image-rendering, clip, color, cursor, display, overflow.
- Уникальные SVG-свойства:
  - clip-rule, flood-color, flood-opacity, stop-opacity, kerning, tech-anchor, color-profile, color-rendering, fill, fill-opacity, fill-rule, marker, marker-end, marker-mid, marker-start, stroke, stroke-width, stop-color, lighting-color, enable-background, dominant-baseline, color-interpolation-filters, color-interpolation, glyph-orientation-horizontal, glyph-orientation-vertical, shape-rendering, baseline-shift, alignment-baseline, stroke-miterlimit, stroke-linejoin, stroke-linecap, stroke-dashoffset, stroke-dasharray, stroke-opacity.
- И это только в SVG1. В SVG2 добавлены и другие свойства.

Как задать свойства для SVG?

- С помощью внешнего файла стилей CSS.
- С помощью инлайновых стилей:

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1"
style="width: 300px; height: 300px;" viewBox="0 0 300
300">
  <polygon
    style="fill: #FF931E; stroke: #ED1C24; stroke-width:
5;"
    points = "279.1,160.8 195.2,193.3 174.4,280.8
117.6,211.1 27.9,218.3 76.7,142.7 42.1,59.6 129.1,82.7
197.4,24.1 202.3,114 "/>
</svg>
```

star.svg

Как задать свойства для SVG?

- С помощью встроенных стилей (внутри SVG):

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1"
width="300px" height="300px" viewBox="0 0 300 300">
    <style type="text/css">
        polygon { fill: ... ;}
    </style>
    <polygon points = "279.1,160.8 195.2,193.3
174.4,280.8   117.6,211.1 27.9,218.3 76.7,142.7 42.1,59.6
129.1,82.7 197.4,24.1 202.3,114 "/>
</svg>
```

star.svg

Как задать свойства для SVG?

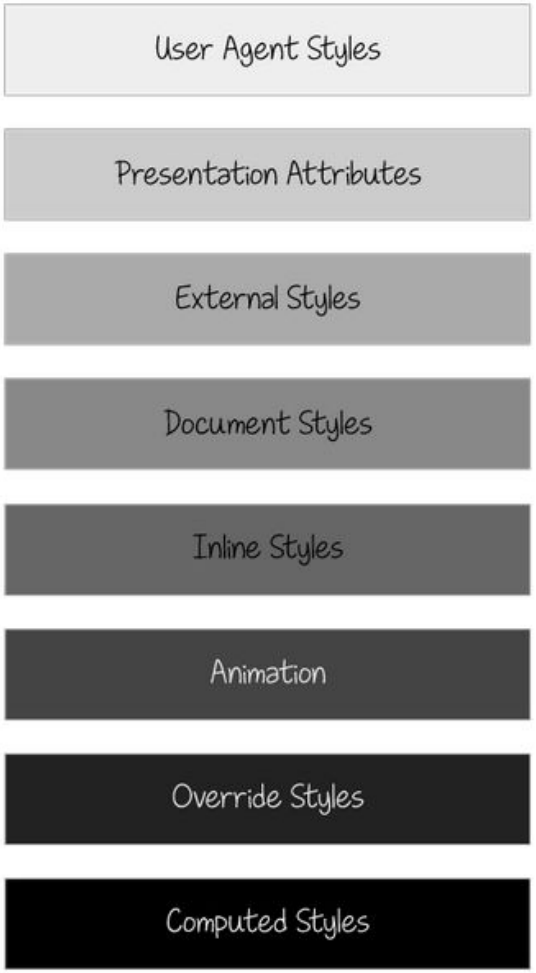
- С помощью встроенных стилей (за пределами SVG):

```
<!DOCTYPE html> <html><head>...</head>
    <body>
        <style type="text/css">
            svg { width: ...; }
            polygon { fill: ... ; }
        </style>
        <svg version="1.1" viewBox="0 0 300 300">
            <!--SVG content-->
        </svg>
    </body>
</html>
```

star.svg

Приоритеты стилей для SVG

```
<style>
    circle {fill: orange;}
</style>
<svg version="1.1" viewBox="0 0 400
400">
    <g fill="yellow">
        <circle cx="100" cy="100"
r="100" fill="blue" style="fill:
deepPink;" />
        <!-- ... -->
    </g>
</svg>
```



Styles lower in the diagram override those above them

Анимация SVG с помощью CSS.

- Может анимировать только свойства, совместные с CSS, и не может анимировать свойства, присущие только SVG.
  - Например, color – может, fill – не может.

Сравнение типа вставки SVG

Embedding Technique	CSS Animations	CSS Interactions
`<img src="mySVG.svg" alt=".." />`	Yes, only if inside <svg>	No
`.el {background: url(mySVG.svg);}`	Yes, only if inside <svg>	No
`<object type="image/svg+xml" data="mySVG.svg"><!--fallback--></object>`	Yes, only if inside <svg>	Yes, only if inside <svg>
`<embed type="image/svg+xml" src="mySVG.svg" />`	Yes, only if inside <svg>	Yes, only if inside <svg>
`<iframe src="mySVG.svg"><!--fallback--></iframe>`	Yes, only if inside <svg>	Yes, only if inside <svg>
`<svg><!--SVG content--></svg>`	**Yes**	**Yes**

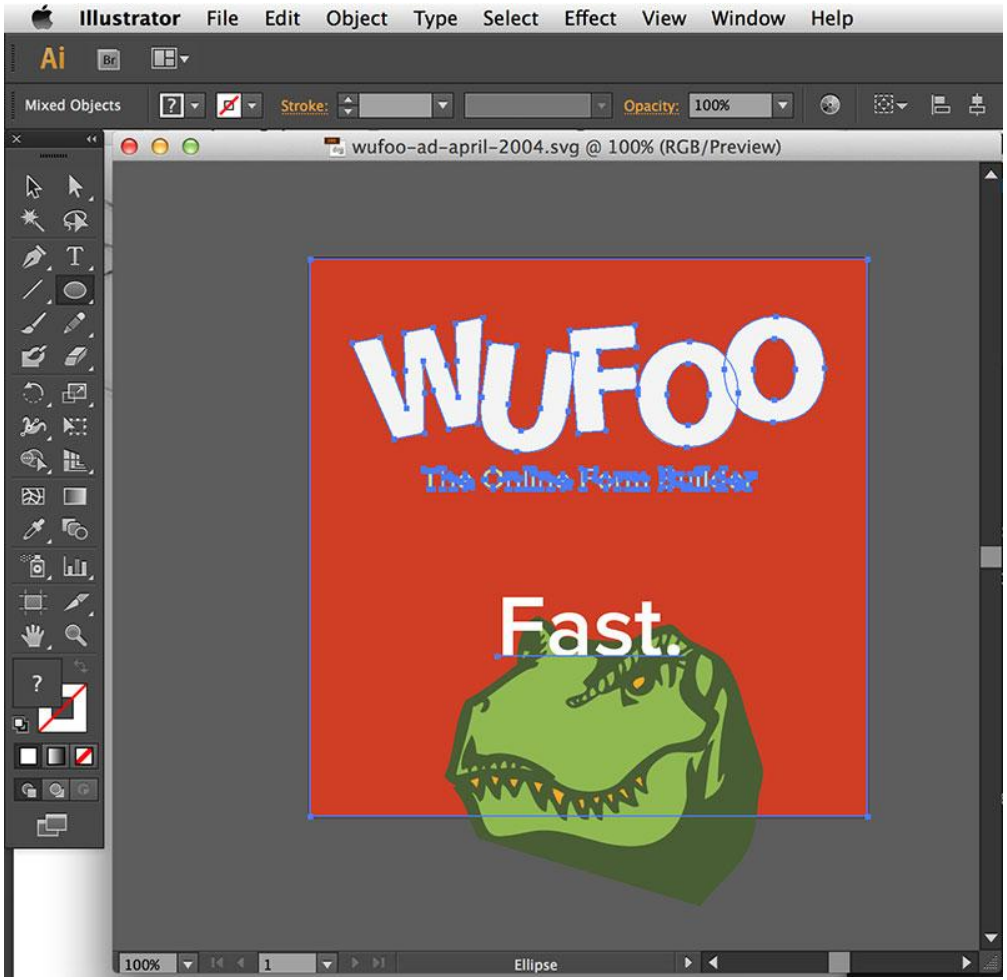
Анимация SVG с помощью CSS.

Порядок создания анимационного блока SVG.

- 1. Создание плана анимации и SVG- изображения.
- 2. Сохранение, сжатие и обработка SVG- изображения.
- 3. Создание нужных классов и идентификаторов для дальнейшего управления анимацией.
- 4. Вставка SVG в HTML.
  - например, с помощью svg-inject, PHP или же напрямую inline.
- 5. Создание CSS для анимации элементов SVG-изображения.

Пример создания рекламного баннера SVG

- 1. Создаем SVG-изображение.



Пример создания рекламного баннера SVG

- 1. Создаем SVG-изображение – особенности:
  - Весь текст в логотипе (Wufoo) выполнен и сохранен контурами (outlines) для их дальнейшей раздельной анимации.
  - Текст «Fast» выполнен и сохранен как текст (он будет только заменяться на другой текст).

- 2. Сохранение в SVG.
  - Можно сохранить в SVG прямо в AI.
  - Обрабатываем и сжимаем файл с помощью сервиса https://jakearchibald.github.io/svgomg/

Пример создания рекламного баннера SVG

- 3. Создание нужных классов и идентификаторов для дальнейшего управления анимацией

```
<svg version="1.1" id="wufoo-ad"
    xmlns="http://www.w3.org/2000/svg"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    viewBox="0 0 400 400"
    enable-background="new 0 0 400 400"
    xml:space="preserve">

  <!-- background -->
  <rect class="wufoo-background" fill="#D03E27" width="400" height="400"/>
  <!-- logo letters -->
    <path class="wufoo-letter" fill="#F4F4F4" d="M60.858,129...." />
    <path class="wufoo-letter" fill="#F4F4F4" d="..." />
    <path class="wufoo-letter" fill="#F4F4F4" d="..." />
    <path class="wufoo-letter" fill="#F4F4F4" d="..." />
    <path class="wufoo-letter" fill="#F4F4F4" d="..." />
  <!-- dinosaur -->
  <g class="trex">
      <path ... />
      <path ... />
  </g>
    <text transform="matrix(1 0 0 1 134.1416 285.127)" class="text-1">
    Fast.</text>
    <text transform="matrix(1 0 0 1 112 285.127)" class="text-2">
    Smart.</text>
    <text transform="matrix(1 0 0 1 39 285.127)" class="text-3">
    Formidable.</text>
</svg>
```

Пример создания рекламного баннера SVG

- 4. Вставка SVG в HTML.
- 5. Создание CSS для анимации элементов SVG-изображения.

- Вначале анимируем текст целиком
  - текст поочередно появляется (задержка 1,5 сек) и исчезает.

```
@keyframes hideshow {
  0% { opacity: 1; }
  10% { opacity: 1; }
  15% { opacity: 0; }
  100% { opacity: 0; }
}

.text-1 { animation: hideshow 10s ease infinite; }
.text-2 { opacity: 0; animation: hideshow 10s 1.5s ease infinite; }
.text-3 { opacity: 0; animation: hideshow 10s 3s ease infinite; }
```

Пример создания рекламного баннера SVG

- 5. Создание CSS для анимации элементов SVG-изображения.
- Анимация длительностью 5 секунд с обратным ходом, бесконечная.
- Используем SCSS!

```
.wufoo-letter {
  animation: kaboom 5s ease alternate infinite;
  &:nth-child(2) {     animation-delay: 0.1s;   }
  &:nth-child(3) {     animation-delay: 0.2s;   }
  &:nth-child(4) {     animation-delay: 0.3s;   }
  &:nth-child(5) {     animation-delay: 0.4s;   }
}

@keyframes kaboom {
  90% {       transform: scale(1.0);   }
  100% {      transform: scale(1.1);   }
}
```

Пример создания рекламного баннера SVG

- 5. Создание CSS для анимации элементов SVG-изображения.
- Высовываем динозавра через 6,5 секунд после начала всей анимации. Затем быстро убираем.

```
@keyframes popup {
  0% {
    transform: translateY(150px);
  }
  34% {
    transform: translateY(20px);
  }
  37% {
    transform: translateY(150px);
  }
  100% {
    transform: translateY(150px);
  }
}

.trex {
  transform: translateY(150px);
  animation: popup 10s 6.5s ease infinite;
}
```

Пример создания рекламного баннера SVG

- Хак – создание responsible SVG animation.
  - Основная идея – обертка (“wrap“) будет всегда квадратной и определяться шириной.
  - Высоту обертки делаем нулевой, ставим верхнее внутреннее поле 100%
  - Абсолютно позиционируем SVG, делаем его 100% высоты и ширины.
- Чаще всего, так как это банер, и он должен быть кликабельным, вместо для обертки ставят <a href="">, и делают обертку блоком display: block;

```
<div class="wufoo-ad-wrap">
  <svg class="wufoo-ad">
    ...
  </svg>
</div>
```

```
.wufoo-ad-wrap {
  height: 0;
  padding-top: 100%;
  position: relative;
}
.wufoo-ad {
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
}
```

Другой вариант создания fluid svg.

- Удаляем '**width**' и '**height**' свойства с тега <svg>.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
viewBox=".." width=".." height=".." >

    <!-- SVG content -->

</svg>
```

logo.svg

- Добавляем свойство '**viewBox**' если оно не установлено.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
viewBox="..">

    <!-- SVG content -->

</svg>
```

logo.svg

Другой вариант создания fluid svg.

- Устанавливаем свойство '**preserveAspectratio**' в значение '**xMidYMid meet**'.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
viewBox=".." preserveAspectRatio="xMidYMid meet">

    <!-- SVG content -->

</svg>
```

logo.svg

- Для фоновых изображении, встроенных <embed> SVG – хак не нужен.
  - только для IE нужно указать:

```
<img src="logo.svg" alt=".." />
```

```
img {
    width: 100%;
}
```

SMIL Animation

- SMIL - Synchronized Multimedia Integration Language – XML язык для описания мультимедийных презентаций.
  - Определяет тайминг, переходы, анимации и т.п.

- Определенные свойства SVG элементов не анимируются через CSS анимации.
- SVG путь содержит определенный набор данных (d="" атрибут) которые определяют форму пути.
  - Эти данные могут быть анимированы через SMIL!
  - Кроме того, могут быть анимированы через JavaScript (существуют некоторые библиотеки - http://snapsvg.io/).

Сравнение анимации через SMIL и JavaScript

- JS анимации не работают тогда, когда SVG внедрен через тег <img>!
- JS анимации не работают тогда, когда SVG внедрен через свойство background-image в CSS.
- SMIL animations работают в обоих приведенных выше случаях!

- Однако, у JS тоже есть преимущества:
  - IE не поддерживает SMIL !
  - Определить «не поддерживаемость» можно через modernizr http://modernizr.com/ - делаем fallback и заменяем анимацию на JavaScript или же не делаем анимацию.

Поддержка SMIL браузерами

- IE – не поддерживает!
- Определить «не поддерживаемость» можно через modernizr http://modernizr.com/

SVG анимация через анимационные элементы

- Анимационные элементы в SVG изначально были определены в SMIL, затем появились и в самом стандарте SVG.
- <animate> - позволяет анимировать скалярные атрибуты и свойства в течение времени.
- <set> - представляет собой удобное сокращение для анимации, которое полезно для назначения значения анимации к нечисловым атрибутам и свойствам,
  - Например – атрибут видимости.
- <animateMotion> - передвигает элемент вдоль пути.
- <animateColor> - изменяет значения цвета определенных атрибутов или же свойств с течением времени.
  - Данное свойство уже признано устаревшим...

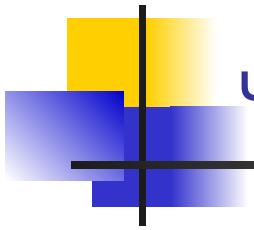
SVG анимация через анимационные элементы

В дополнение, SVG содержит расширения, совместимые с SMIL анимациями, и расширяющие <animateMotion>:

- <AnimateTransform> - позволяет анимировать один из SVG атрибутов в течение времени, как transform.
- Path (атрибут) - позволяет определить данный путь для SVG как траекторию для animateMotion элемента.
- <Mpath> - используется в сочетании с animateMotion для ссылки на траекторию движения, который должен быть использован в качестве траектории движения. Элемент mpath входит внутрь animateMotion, перед закрывающим тегом.
- Keypoints (атрибут) - используются в качестве атрибута для animateMotion для обеспечения точного контроля скорости траектории движения анимации.
- rotate (атрибут) - используется в качестве атрибута для animateMotion контроля вращения.

Что стоит почитать

- Tips for Creating Accessible SVG http://www.sitepoint.com/tips-accessible-svg/
- A Complete Guide to SVG Fallbacks https://css-tricks.com/a-complete-guide-to-svg-fallbacks/
- https://css-tricks.com/search-results/?q=svg
- grunt-svgstore. Merge SVGs from a folder https://github.com/FWeinb/grunt-svgstore
- Grumpicon. A Web app for the Grunticon workflow. http://www.grumpicon.com/
- Анимация SVG-элемента path http://habrahabr.ru/post/207908/
- Animating Vectors with SVG http://24ways.org/2013/animating-vectors-with-svg/
- A JQUERY PLUGIN FOR SVG PATH ANIMATION http://lazylinepainter.info/



Что стоит почитать

- Styling & Animating Scalable Vector Graphics with CSS http://slides.com/sarasoueidan/styling-animating-svgs-with-css--2#/
- http://tutsplus.github.io/Styling-Iconic/styling/index.html
- How to Scale SVG https://css-tricks.com/scale-svg/
- A Guide to SVG Animations (SMIL) https://css-tricks.com/guide-svg-animations-smil/