enriching the web with **css filters**

Raul Hudea

rhudea@adobe.com

@rhudea

agenda

- what are filters ?
- filters - today
- filters - tomorrow
- demos
- q & a

what are filters

ways of modifying the rendering of HTML5 content

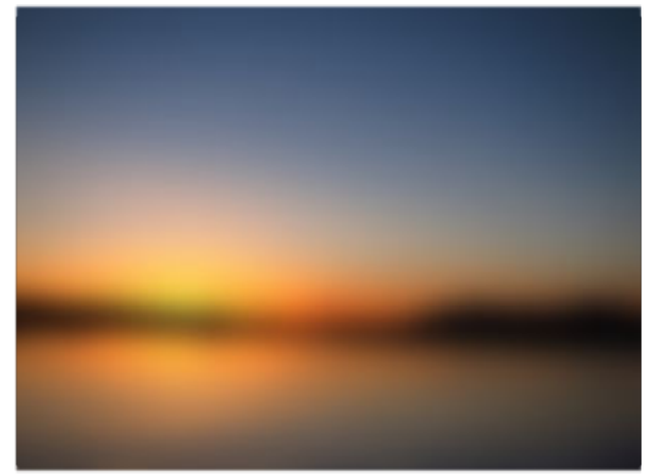
grayscale

brightness

sepia

invert

blur effect - images



- works on all browsers
- bandwidth / space issues
- does not work with any HTML content

blur effect - canvas

```
var imageData = ctx.getImageData(0, 0, canvas.width, canvas.height);
var pixelData = imageData.data;
for (var y = 0; y < canvas.height; y++) {
    for (var x = 0; x < canvas.width; x++) {

        ... blur algorithm (~ 30 lines) ...
        pixelData[i] = r;
        pixelData[i + 1] = g;
        pixelData[i + 2] = b;
        pixelData[i + 3] = a;
    }
}
ctx.putImageData(imageData, 0, 0, 0, 0, imageData.width, imageData.height);
```

- supported by all modern browsers
- access to pixel data
- all computations are done in JavaScript
- cannot draw HTML content

blur effect - webgl

```
gl = canvas.getContext( 'experimental-webgl' );
...
buffer = gl.createBuffer();
gl.bindBuffer( gl.ARRAY_BUFFER, buffer );
...
var program = gl.createProgram();
...
gl.linkProgram( program );
...
// Load program into GPU
gl.useProgram( currentProgram );
...
// 200+ line of boiler plate code
```

see the **webgl-boilerplate** project, by Paul Irish

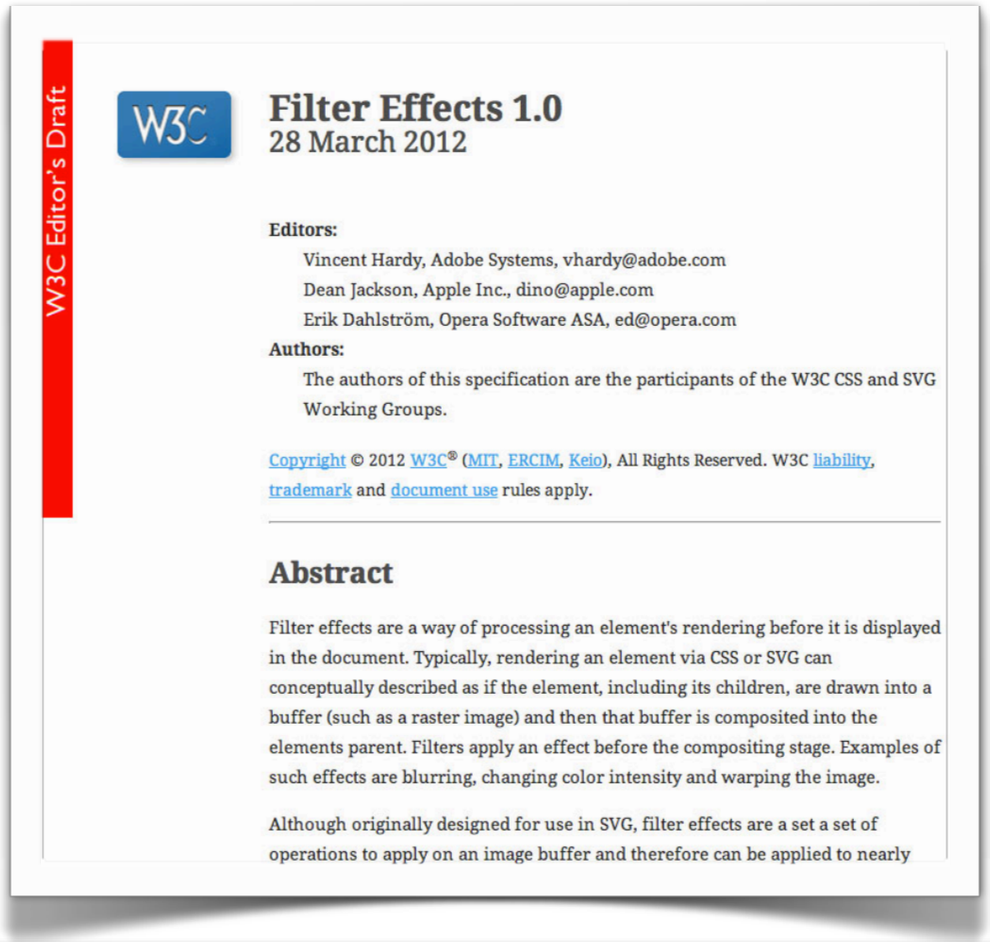
- can be faster than 2D canvas
- uses hardware acceleration
- supported on modern browsers (not IE)
- cannot draw HTML content

blur effect - svg

```
<svg>
    <defs>
        <filter id="Gaussian_Blur">
            <feGaussianBlur in="SourceGraphic" stdDeviation="2"/>
        </filter>
    </defs>
    <image x="0" y="0" width="180" height="180"
        xlink:href="bridge.jpg" filter="url(#Gaussian_Blur)">
    </image>
</svg>
```

- supported on modern browsers (also IE 10)
- hardware accelerated
- foreignObject can be used to render HTML (not on IE)

filter effects spec



W3C Editor's Draft

Filter Effects 1.0
28 March 2012

Editors:
Vincent Hardy, Adobe Systems, vhardy@adobe.com
Dean Jackson, Apple Inc., dino@apple.com
Erik Dahlström, Opera Software ASA, ed@opera.com
Authors:
The authors of this specification are the participants of the W3C CSS and SVG Working Groups.

Abstract

Filter effects are a way of processing an element's rendering before it is displayed in the document. Typically, rendering an element via CSS or SVG can conceptually described as if the element, including its children, are drawn into a buffer (such as a raster image) and then that buffer is composited into the elements parent. Filters apply an effect before the compositing stage. Examples of such effects are blurring, changing color intensity and warping the image.

Although originally designed for use in SVG, filter effects are a set a set of operations to apply on an image buffer and therefore can be applied to nearly

- filter effects for HTML / SVG
- evolved from SVG Filters spec

blur effect - css

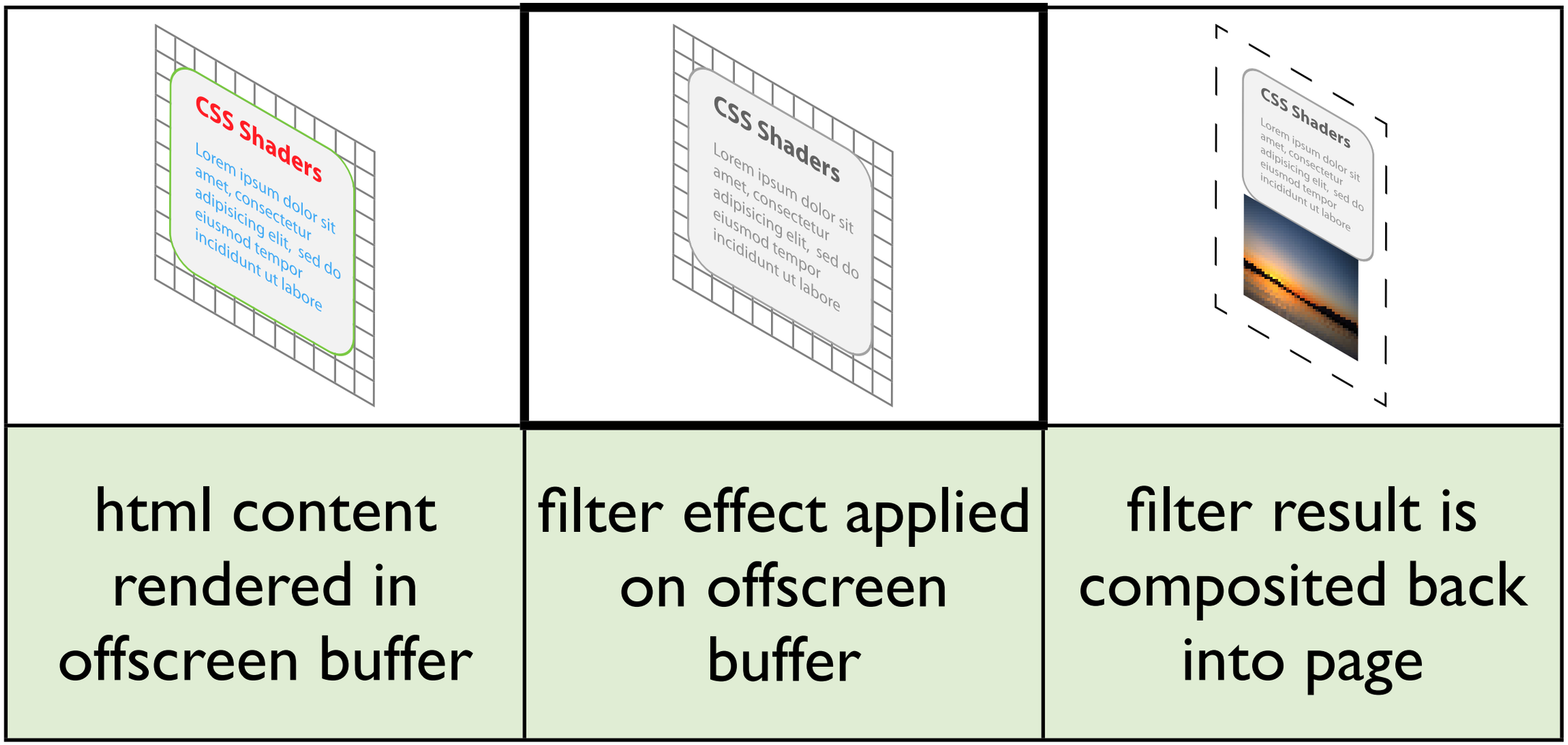
```
<style>
    #my_element { filter: blur(4); }
</style>
<div id="my_element"> ... </div>
```

- other predefined filters available (grayscale, invert, hue-rotate, drop-shadow, brightness...)
- hardware accelerated
- can be applied on any HTML content

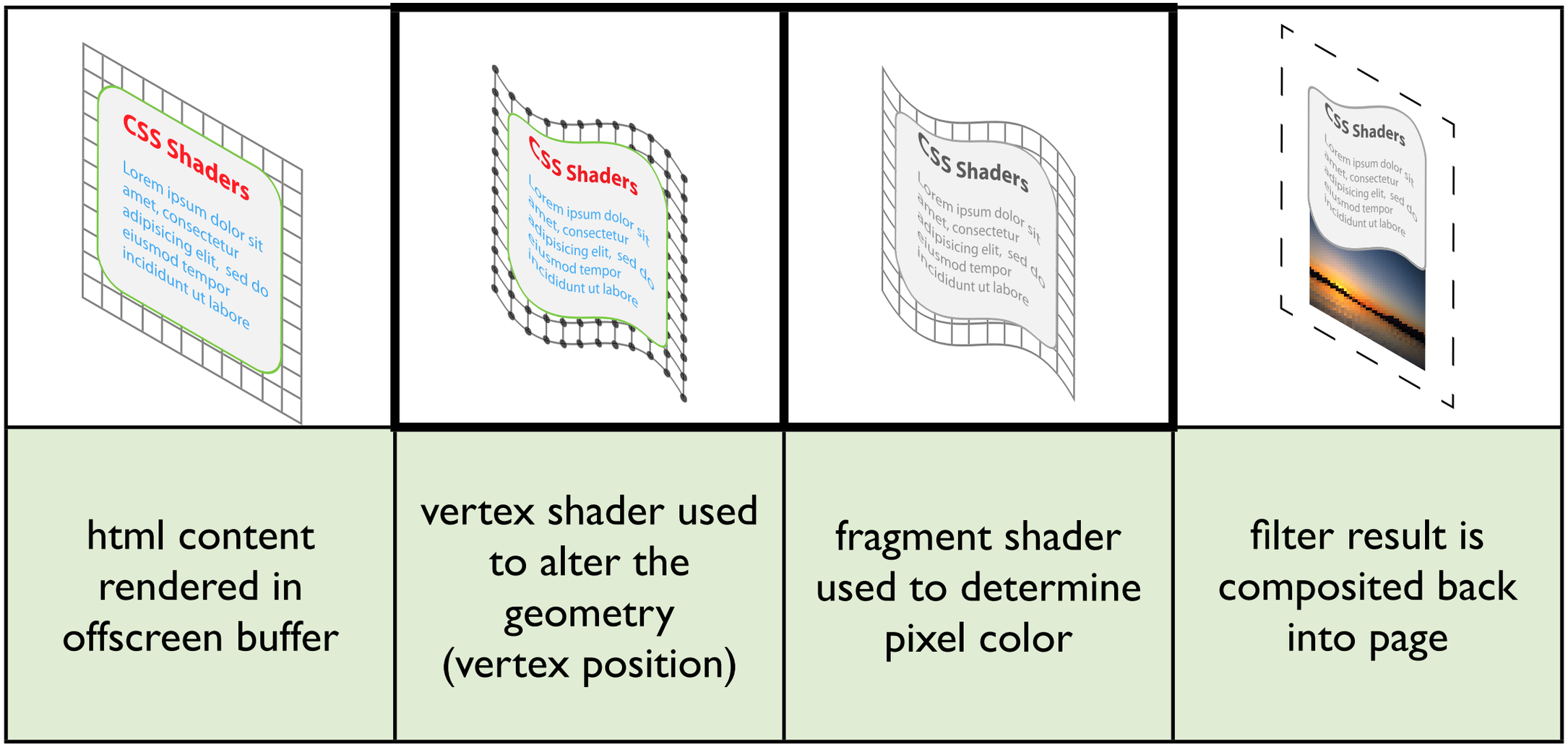
css shaders

- custom filter effects
- allow pixel and geometry manipulation
- use OpenGL ES shading language (same as WebGL)
- hardware accelerated

how filters work



how shaders work



css shaders

```
<style>
    #myElement {
        filter: custom(url(flag.vs) url(scale.fs),
        20 20,
        txf rotateX(30deg),
        amount 0.5);
    }
    </style>

<div id="myElement">..<div>
```

- use **custom** as filter function
- only one vertex and/or one fragment shader
- work on any HTML content

vertex shader

```
precision mediump float;
attribute vec4 a_position;
attribute vec2 a_texCoord;
uniform mat4 u_projectionMatrix;
varying vec2 v_texCoord;
uniform mat4 txf;
... const defines (PI, PHI) ...
void main() {
    v_texCoord = a_texCoord;
    vec4 pos = a_position;
    pos.z = 40.0 * cos(pos.x * PI * 2.0 + PHI);
    gl_Position = u_projectionMatrix * txf * pos;
```

- decides the vertex position
- operates on a mesh of vertices (distortion effects)
- can send data to fragment shaders

fragment shader

```
precision mediump float;
uniform sampler2D u_texture;
uniform float amount;
varying vec2 v_texCoord;
void main() {
    vec4 color = texture2D(u_texture, v_texCoord);
    vec4 icolor = vec4(color);

    icolor = vec4(1.0 - icolor.r, 1.0 - icolor.g,
                  1.0 - icolor.b, icolor.a);

    gl_FragColor = (1.0 - amount) * color + amount * icolor;
}
```

- decides the pixel color
- receives interpolated data from vertex shader

css shader studio

DEMO

filters - **now**

- canvas (2D/3D) for filter effects (not for general HTML content)
- SVG filters, supported in all major browsers
- SVG filters with foreignObject, for HTML content (not in IE)
- use a mix of canvas and SVG filters

filters - **soon**

- filter property will apply to all HTML content
- predefined filter effects
- CSS shaders (custom, GPU-run effects)

References

http://rhudea.github.com

links to specification, articles, demos and more

rhudea@adobe.com **@rhudea**

Thanks !

rhudea@adobe.com

@rhudea