Aplicaciones Web modernas basadas en estándares abiertos

Huibert Aalbers
Senior Certified Software IT Architect

IT Insight podcast

- Este podcast pertenece a la serie IT Insight
- Pueden suscribirse al podcast a través de iTunes.
- El material adicional (presentación en formato PDF, white papers y otros) se obtienen directamente del sitio http://www.huibert-aalbers.com en la sección de IT Insight
- Me pueden enviar comentarios acerca del podcast o preguntas a mi correo personal, huibert_aalbers@mac.com

Agenda

- Una nueva generación de aplicaciones Web
- La propuesta basada en estándares que ofrece la W3C
  - El nuevo tag <canvas>
  - El estándar SVG
- Conclusión

Una nueva generación de aplicaciones Web

- La velocidad de transmisión de datos que se puede lograr ahora en casi cualquier parte del mundo es tal que ahora parece razonable diseñar aplicaciones Web que puedan sustituir incluso las aplicaciones de escritorio más complejas
- Son varias las empresas las que nos quieren convencer de que esto no se puede hacer con los estándares actuales de la W3C (HTML, CSS, etc) y que se requiere agregar funcionalidad propietaria a los browsers para poder crear aplicaciones atractivas y poderosas
- La realidad es que esto no es cierto, los estándares han evolucionado mucho y hoy en día es posible crear aplicaciones realmente sofisticadas basadas en exclusivamente en estándares

Una nueva generación de aplicaciones Web

- En esta presentación no se pretende cubrir todas las tecnologías estándar relativamente nuevas que permiten crear aplicaciones web que se parecen cada vez más a las de escritorio
- En esta presentación solo cubriremos dos tecnologías, el tag <canvas> y SVG, con el objetivo de generar interés entre los desarrolladores Web que quieren crear aplicaciones revolucionarias y que piensan que esto no se puede hacer con tecnologías abiertas que corran en cualquier browser

El origen del tag <canvas>

- En el 2005, Apple lanzó Mac OS X 10.4, Tiger. Entre las novedades incluía la posibilidad de crear Widgets de forma sencilla utilizando HTML y JavaScript.
- Sin embargo, para poder crear Widgets atractivos se necesitaba poder crear un objeto en el que se pudiera dibujar libremente, cosa que no existía. Por eso Apple creó el tag <canvas> que posteriormente se agregó a Safari, el browser que produce esa empresa
- De forma simultánea, Apple propuso esa extensión como parte de HTML 5

¿Qué browsers soportan <canvas>?

- El tag <canvas> es parte de la especificación Web Applications 1.0 de la organización WhatWG, la cual se conoce también como HTML 5, por lo que muchos navegadores ya lo soportan
  - Safari 2.0 y posterior
  - Firefox 1.5, y otros browsers basados en Gecko 1.8 y posterior
  - Opera 9 y posterior
- A pesar de que Microsoft participa en el esfuerzo por desarrollar HTML, ninguna versión de IE soporta actualmente el tag de forma nativa
  - Se puede resolver este problema usando una librería escrita en JavaScript

Soporte de <canvas> en IE

- A pesar de que IE no soporta el tag <canvas>, este browser soporta VML (Vector Markup Language) un formato XML obsoleto utilizado para representar gráficos vectoriales
- Ni VML ni PGML (un formato similar impulsado por Adobe, Sun y otros) fueron ratificados como estándares de la W3C pero son el origen de SVG, que sustituye a ambos.
- Google ha desarrollado una librería en JavaScript que utiliza VML para implementar el tag <canvas> en IE 5.5+
  - http://excanvas.sourceforge.net/
  - Solo se requiere agregar una línea de código a la página HTML
    - <!--[if IE]><script type="text/javascript" src="excanvas.js"></script><![endif]-->

Un ejemplo sencillo del uso de <canvas>

```
<html>
<body>
<canvas id="animated" width="500px" height="100px"></canvas>
<script>
var animatedctx = document.getElementById('animated').getContext('2d');
var r,g,b,alpha;

function addBox() {
  animatedctx.beginPath();

  // borra la pantalla aleatoria
  if (Math.random() > 0.99) animatedctx.clearRect(0,0,500,100);
  r = Math.round(Math.random()*255); g = Math.round(Math.random()*255);
  b = Math.round(Math.random()*255); alpha = (Math.round(Math.random()*100))/100;
  animatedctx.fillStyle='rgba('+r+','+g+','+b+','+alpha+')';

  animatedctx.rect(Math.random()*500,Math.random()*100,Math.random()*100, Math.random()*100 );
  animatedctx.fill();

  setTimeout('addBox()', 50 );
}
addBox();
</script>
</body>
</html>
```

El manejo de gráficos en un Canvas

- Al haber sido desarrollado por Apple, el API para manipular un Canvas es similar al de Core Graphics (Quartz) el API para dibujar en 2D en Cocoa bajo OS X
- Características que soporta el API
  - Trazos complejos
  - Regiones
  - Imágenes de tipo bitmap
  - Gradientes
  - Transformaciones
  - Sombras

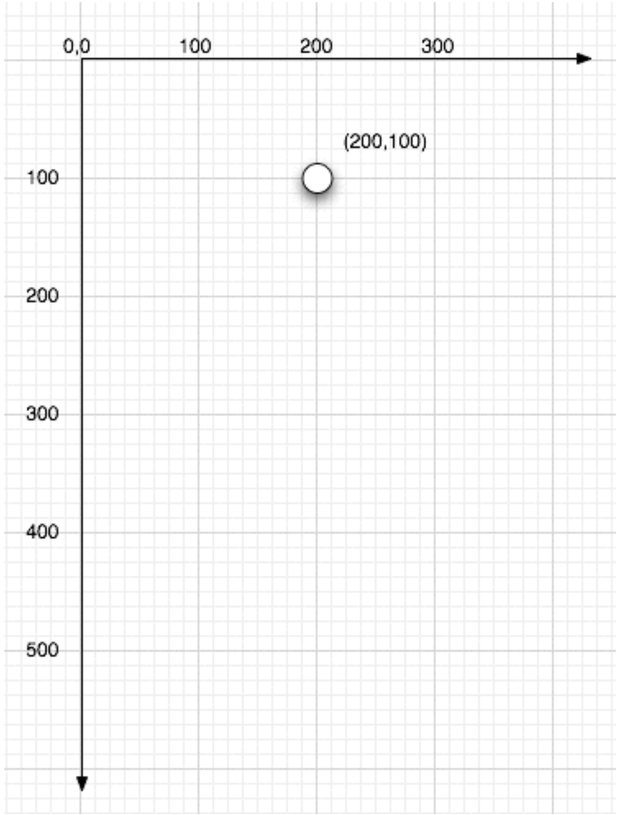
¿Cómo usar <canvas>?

- El elemento <canvas> se puede accesar por JavaScript a través del DOM
- Para conseguir fácilmente una referencia al espacio en el que vamos a pintar, lo mejor es usar un identificador único en el tag
- El objeto canvas contiene un atributo “context” al que se le mandan las intrucciones para dibujar

```
<html>
<body>
<canvas id="canvas1" width="500px" height="100px"></canvas>

<script>
   var context = document.getElementById('canvas1').getContext('2d');
   context.fillrect(10,10,150,100);
</script>

</body>
</html>
```

Sistema de coordenadas

- Como en el formato PDF, las coordenadas de la página se miden desde el extremo superior izquierdo
- Es posible mover el origen de las coordenadas
  - ctx.translate(x,y);
- También se puede rotar ‘la hoja’
  - ctx.rotate(ángulo);
  - El ángulo se mide en radianes



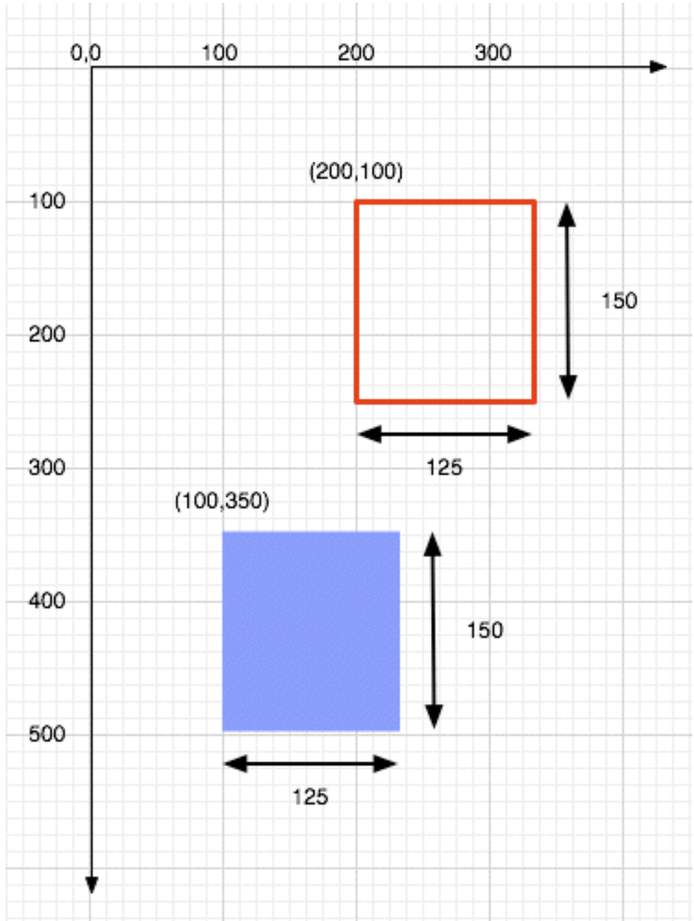
Sistema de coordenadas

```
<html>
<body>
<canvas id="canvas2" width="250px" height="250px"></canvas>

<script>
  var ctx = document.getElementById('canvas2').getContext('2d');
  var i,steps,angle,alpha;

  steps = 20;
  angle = 2*3.14/steps;
  ctx.translate(125,125);

  for (i=0;i<steps;i++) {
    ctx.beginPath();
    ctx.rotate(angle);

    alpha = (1/steps)*(i+1);
    ctx.fillStyle='rgba(255,0,0,'+alpha+')';

    ctx.rect(0,0,25,100);
    ctx.fill();
  }
</script>
</body>
</html>
```

Dibujar un rectángulo

- Rectángulos
  - ctx.fillRect(x, y, ancho, alto);
  - ctx.strokeRect(x, y, ancho, alto);

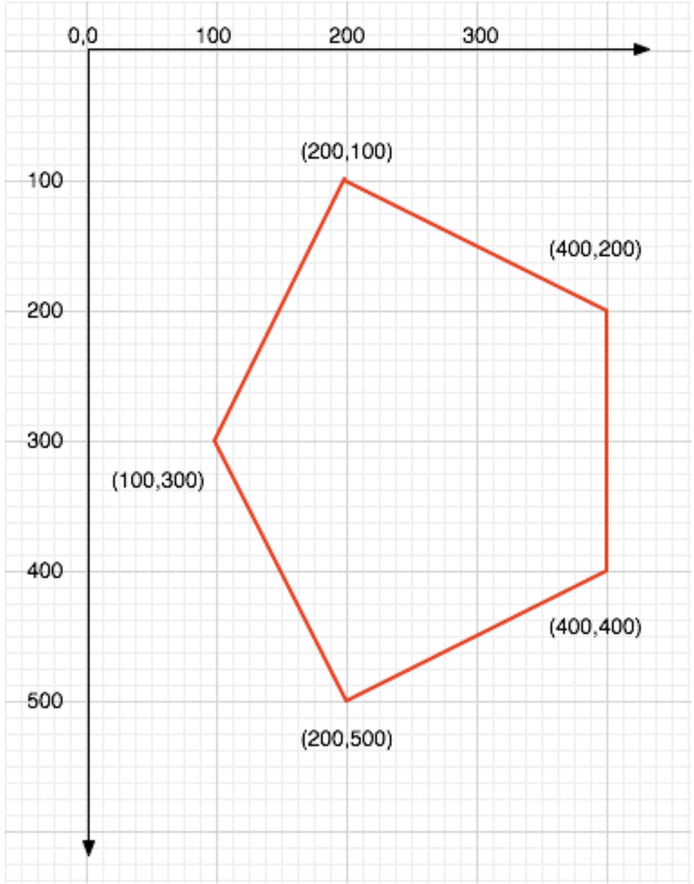
```
context.strokeStyle = ‘rgba(255,0,0,1.0);
context.strokeRect(200,100,125,150);

context.fillStyle = ‘rgba(115,135,255,1.0);
context.fillRect(100,350,125,150);
```



Dibujar un polígono

- Los polígonos se crean trazando su forma mediante el uso de líneas

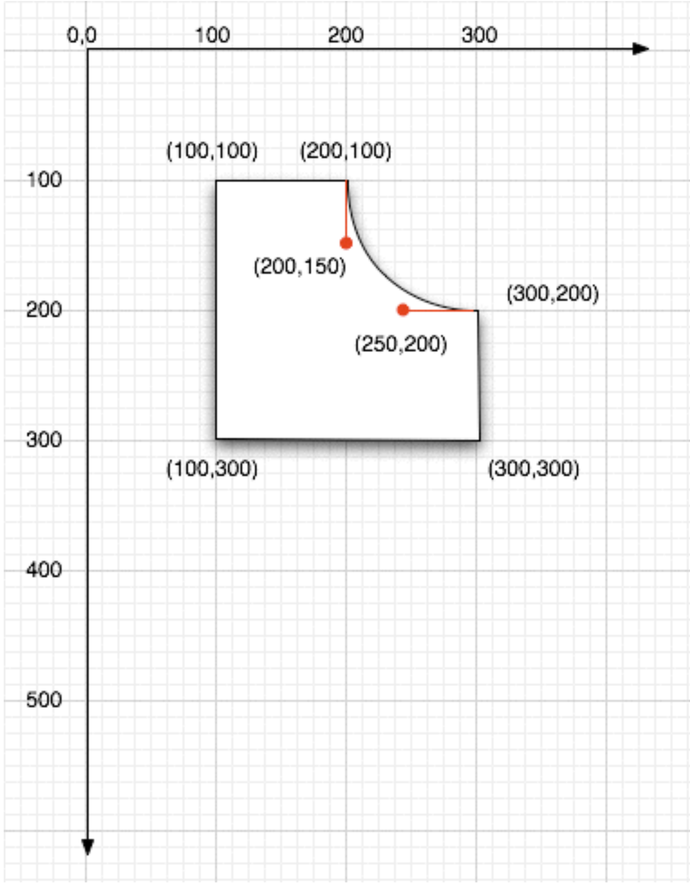
```
context.beginPath();
  context.moveTo(200,100);
  context.lineTo(400,200);
  context.lineTo(400,400);
  context.lineTo(200,500);
  context.lineTo(100,300);
context.closePath();

context.stroke();
```



Dibujar formas complejas

- Es posible crear formas complejas usando curvas de Bézier

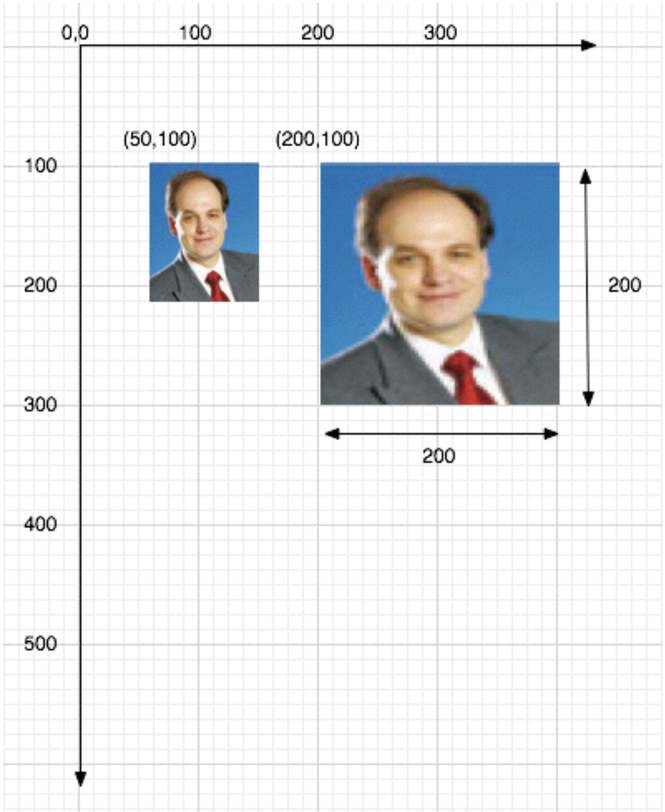
```
ctx.beginPath();
  ctx.moveTo(100,100);
  ctx.lineTo(200,100);
  ctx.bezierCurveTo(200,150,250,200,300,200);
  ctx.lineTo(300,300);
  ctx.lineTo(100,300);
ctx.closePath();

ctx.stroke();
```



Manipulación de Imágenes

- Es muy sencillo pintar imágenes en un canvas

```
var ctx = document.getElementById('canvas1').getContext('2d');
ctx.drawImage(img,50,100);
ctx.drawImage(img,200,100,300,300);
```



¿Dónde conseguir más información?

- Especificación
  - http://www.w3c.org
  - http://developer.apple.com/documentation/AppleApplications/Reference/SafariJSRef/Classes/Canvas.html
- Ejemplos
  - http://developer.mozilla.org/en/docs/Canvas_tutorial
  - http://www.abrahamjoffe.com.au/ben/canvascape/
  - http://the.fuchsia-design.com/2007/06/wwdc-2007-webkit-svg-demos.html
  - http://www.starxpert.fr/mozilla/demos/MouseMove/canvas.html

SVG

- SVG (Scalable Vector Graphics) es un formato XML estándar desarrollado por la W3 (World Wide Web Consotium) para representar gráficos vectoriales en 2D
  - SVG es el resultado de la estandarización de dos formatos, VML (impulsado por Microsoft y Macromedia) y PGML (propuesto por Sun Microsystems y Adobe)
  - SVG 1.0 se convirtió en una recomedación de la W3C en septiembre del 2001
  - SVG 1.1 (el más usado) se presentó en el 2003
  - SVG 1.2 fue lanzado en agosto del 2006

¿Qué browsers soportan SVG?

- SVG es un estándar emergente que de alguna manera compite con la costumbre de utilizar imágenes en formatos bitmap, por lo que la adopción del estándar ha sido relativamente lenta
  - Safari 2.0.2 y posterior
  - Firefox 1.5, y otros browsers basados en Gecko 1.8 y posterior
  - Opera 9 y posterior
- Ninguna versión de IE soporta actualmente SVG de forma nativa
  - Afortunadamente, Adobe provee un plug-in que permite visualizar documentos SVG y existe otro llamado Renesis

SVG

- SVG permite representar gráficas vectoriales a través de un archivo XML

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 20010904//EN" "http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd">
<svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" xml:space="preserve" width="600" height="100" viewBox="0 0
1200 200">
  <!-- Pattern Definition -->
  <defs>
    <pattern id="checkerPattern" patternUnits="userSpaceOnUse" x="0" y="0" width="20" height="20" viewBox="0 0 10 10">
      <rect x="0" y="0" width="5" height="5" fill="lightblue" />
      <rect x="5" y="5" width="5" height="5" fill="lightblue" />
    </pattern>
    <linearGradient x1="10%" x2="90%" id="myFillGrad" >
      <stop offset="5%" stop-color="red" />
      <stop offset="95%" stop-color="blue" stop-opacity="0.5" />
    </linearGradient>
    <linearGradient id="myPad" spreadMethod="pad" xlink:href="#myFillGrad" />
  </defs>
  <!-- Background -->
  <rect x="0" y="0" width="100%" height="100%" fill="url(#checkerPattern)" />
  <!-- Gradient Example -->
  <rect x="20" y="20" width="1160" height="160" fill="url(#myPad)" stroke="black" stroke-width="2" />
   <line x1="136" y1="0" x2="136" y2="320" stroke="black" stroke-width="2" />
   <line x1="1080" y1="0" x2="1080" y2="320" stroke="black" stroke-width="2" />
</svg>
```

Integración de SVG en un archivo HTML

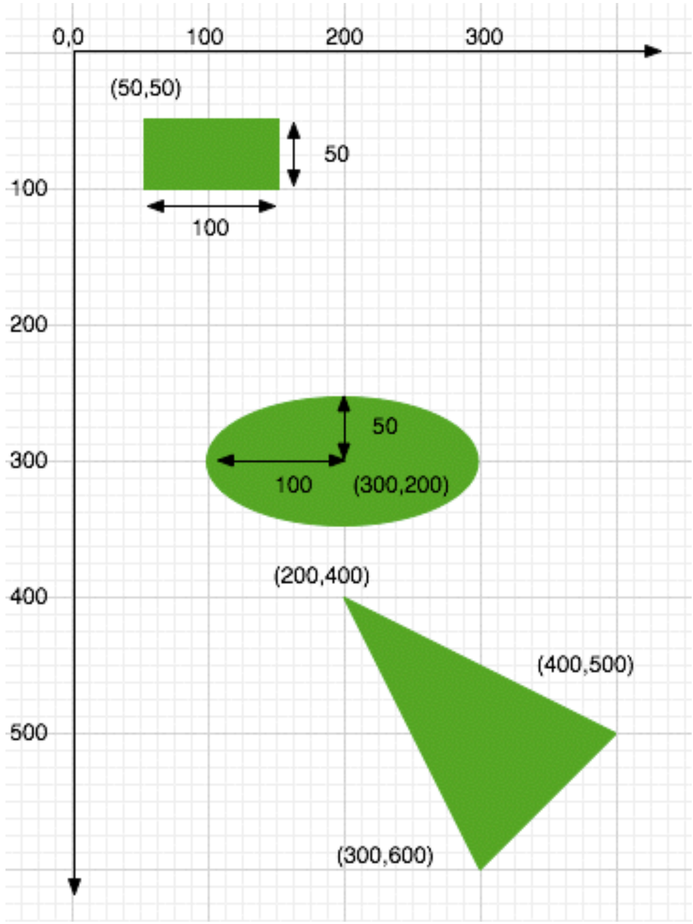
- El XML que contiene los datos SVG normalmente se guarda en un archivo independiente
- En la página HTML es necesario agregar la siguiente instrucción para importar el archivo SVG

```
<object data="grafica.svg" type="image/svg+xml" width="800" height="600">
```

Introducción a SVG

- SVG provee soporte para representar las habituales formas básicas en XML

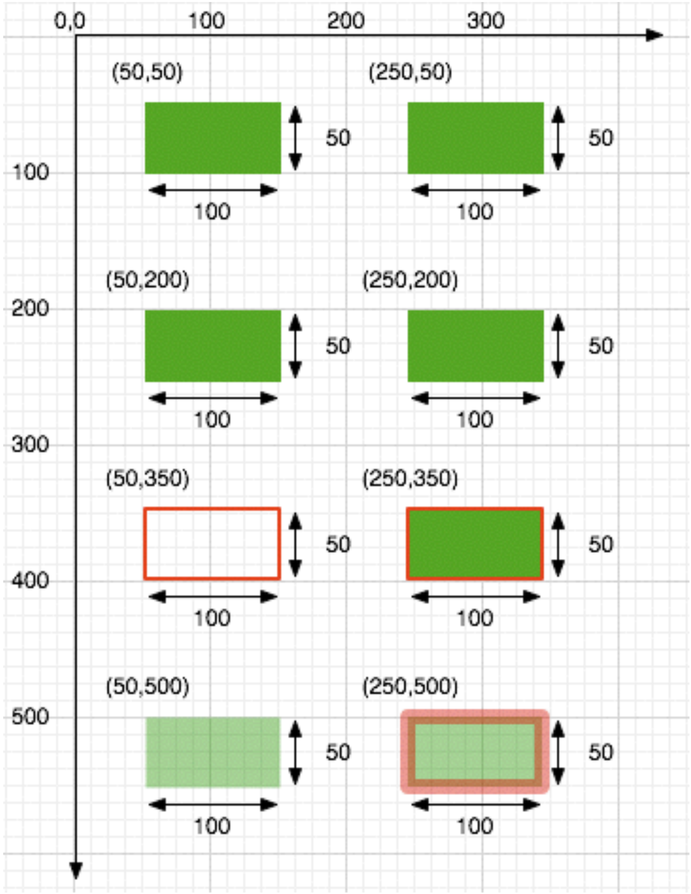
```
<rect x=”50” y=”50” width=”100” height=”50” />
<ellipse cx=”300” cy=”200” rx=”100” ry=”50” />
<polygon points=”200,400 300,600 400,500” />
```



Introducción a SVG

- SVG ofrece múltiples maneras para controlar cómo se pintan los objetos

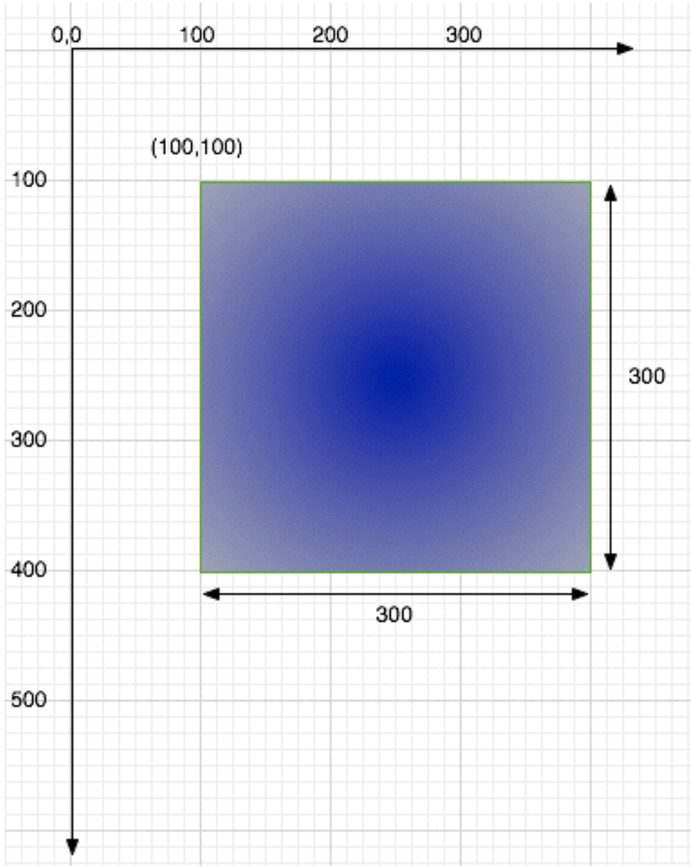
```
<rect ... fill=”green” />

<rect ... fill=”rgb(0,255,0)” />

<rect ... fill=”rgb(0,100%,0)” />

<rect ... fill=”#00FF00” />

<rect ... stroke=”#FF0000” />

<rect ... fill=”#00FF00” stroke=”#FF0000” />

<rect ... fill=”green” fill-opacity=”0.5” />

<rect ... fill=”green” fill-opacity=”0.5”  stroke=”#00FF00” stroke-width=”8” stroke-opacity=”0.5” />
```



Introducción a SVG

- SVG permite usar de forma sencilla gradientes para rellenar objetos

```
<radialGradient id=”grad2” cx=”0.5” cy=”0.5” r=”0.5” />
  <stop offset=”0” stop-color=”#000099” />
  <stop offset=”1” stop-color=”#8C93AA” />
</radialGradient>

<rect x=”100” y=”100” width=”300” height=”300” fill=”url(#grad2)” />
```



SVG y CSS

- Al igual que con HTML, en un archivo SVG se pueden definir estilos para los objetos

```
<svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" viewBox="0 -51 894 195"
preserveAspectRatio="xMidYMid meet">
  <title>Prueba</title>
<defs>
    <style type="text/css"><![CDATA[
      rectstyle {
        fill:blue;
        stroke:black;
        stroke-width:4;
        stroke-dasharray:10,5;
      }
    ]]></style>
</defs>

<rect width="100" height="100" class="rectstyle" />
</svg>
```

- Utilizando pseudo clases como :hover o :focus es posible agregar efectos interactivos básicos

SVG y Javascript

- Un archivo SVG puede contener rutinas Javascript para poder crear gráficas interactivas
  - Esto funciona igual que en una página HTML donde primero se navega a través del DOM para ubicar el elemento que se quiere modificar usando una de las siguientes funciones
    - .parentNode
    - .childNodes
    - .firstChild
    - .getElementById(id)
    - .getElementsByTagNameNS(ns, nombre)
  - Luego se modifican los valores de los atributos del elemento
    - .getAttribute(nombre)
    - .getAttributeNS(ns,nombre)
    - .setAttribute(nombre,valor)
    - .setAttributeNS(ns,nombre,valor)

SVG y Javascript

- Esta sencilla función crea un rectángulo

```
function crearRectangulo (x, y, ancho, alto) {
   var rectangulo = document.createElementNS('http://www/w3/org/2000/svg','rect');

   rectangulo.setAttribute('x', x);
   rectangulo.setAttribute('y', y);
   rectangulo.setAttribute('width', ancho);
   rectangulo.setAttribute('height', alto);

   return rectangulo;
}
```

SVG y Javascript

- Mediante el uso de Javascript es posible agregar interactividad avanzada a los gráficos SVG
  - Para la gestión de eventos predefinidos
    - onclick

      ```
      •<rect x="20" y="40" width="50" height="30" fill="red" onclick="showEvtData(evt)" />
      ```

    - onmouseover
    - onmousemove
    - etc.
  - Para la gestión de eventos definidos al tiempo de ejecución
    - .addEventListener(‘click’, mifuncion, false);

¿Cuándo usar SVG o <Canvas>?

	SVG	Canvas
Imágen estática	✓	
Gran número de objetos		✓
Interactividad	✓	
Animación	✓	
Existe un modelo de datos		✓

Frameworks

- Desarrollar aplicaciones web avanzadas utilizando los últimos avances en estándares abiertos puede resultar complicado. Por eso, han aparecido varios frameworks que simplifican el trabajo de los desarrolladores.
  - DOJO

    http://dojotoolkit.org
  - SproutCore

    http://www.sproutcore.com
  - Google Web Toolkit

    http://code.google.com/webtoolkit/

Google Code

SproutCore a javascript framework

dojo great experiences ...for everyone

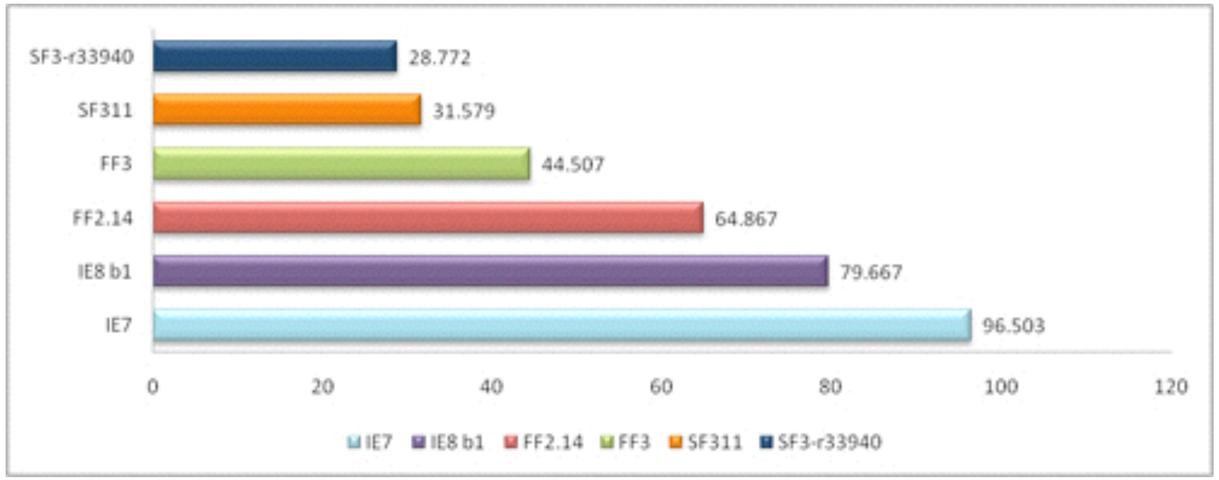
Ejemplos

- En esta presentación hemos cubierto solo parte de los avances recientes en tecnologías de Web estándar. Lo mejor para entender el potencial que ofrecen es ver algunas de las aplicaciones que las usan
  - 280 Slides (Programa de presentaciones estilo PowerPoint)

    http://280slides.com/
  - MobileMe (e-mail, contactos, agenda y galería de fotos)
  - http://mobileme.com

La importancia de usar un buen browser

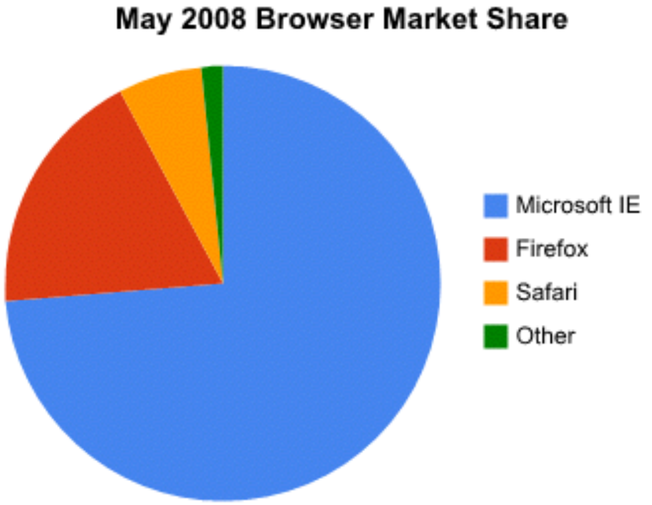
- De la misma manera que un sistema operativo eficiente es importante para que las aplicaciones funcionen bien, un browser que ejecuta Javascript de forma óptima es fundamental para explotar esta nueva generación de aplicaciones Web
  - No todos los browsers son iguales, las diferencias pueden llegar a ser significativas



Conclusión

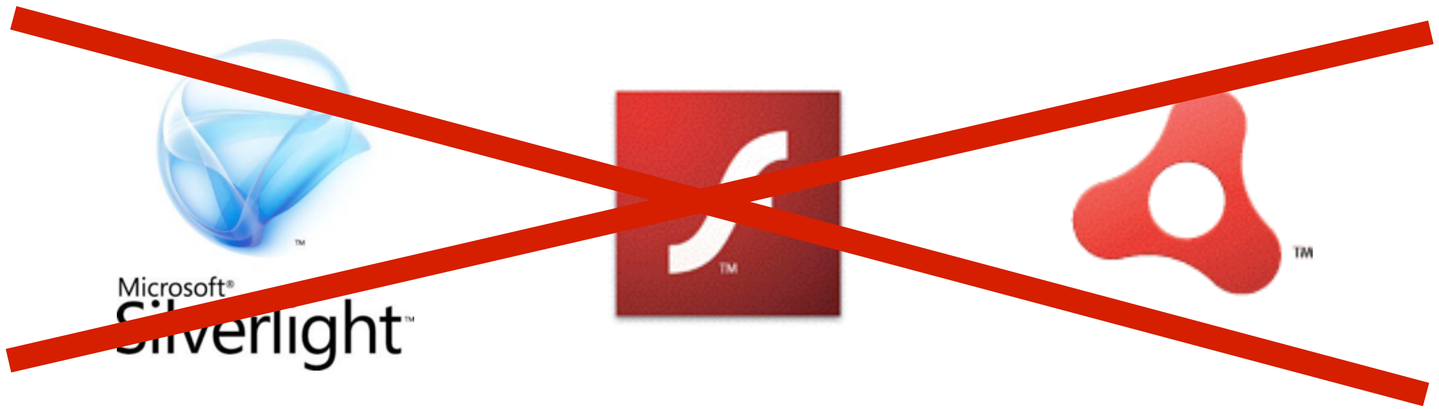
- Internet está cambiando rápidamente. Ya están lejos los tiempos en los que el 95% de los usuarios usaban MSIE para conectarse a la red. En este momento IE tiene menos de 75% de mercado y sigue perdiendo adeptos cada día.

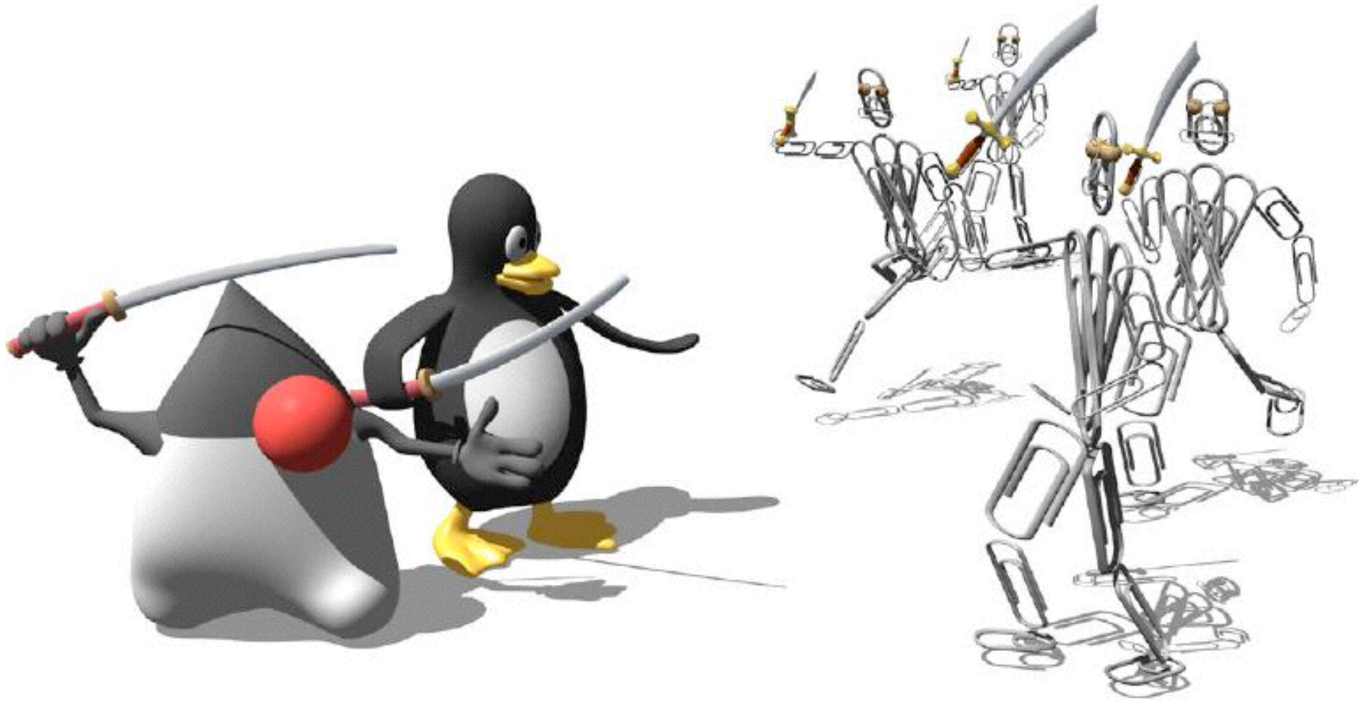
- Actualmente los desarrolladores que apuestan por una tecnología propietaria corren el riesgo de perder a gran parte de su audiencia



Conclusión

- Contrariamente a lo que muchos piensan, las tecnologías estándar para crear páginas web dinámicas han avanzado de forma dramática recientemente
  - Esto hace cada vez menos necesario recurrir a tecnologías proprietarias para crear aplicaciones web atractivas y dinámicas
  - Es una buena noticia para los usuarios que van a poder usar este nuevo tipo de aplicaciones sin necesidad de tener que instalar plug-ins





Para mayor información, contácteme directamente
huibert_aalbers@mac.com