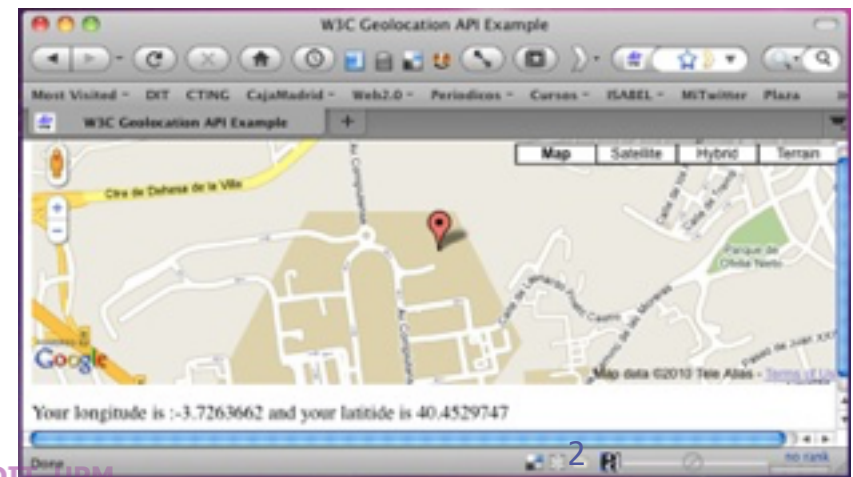
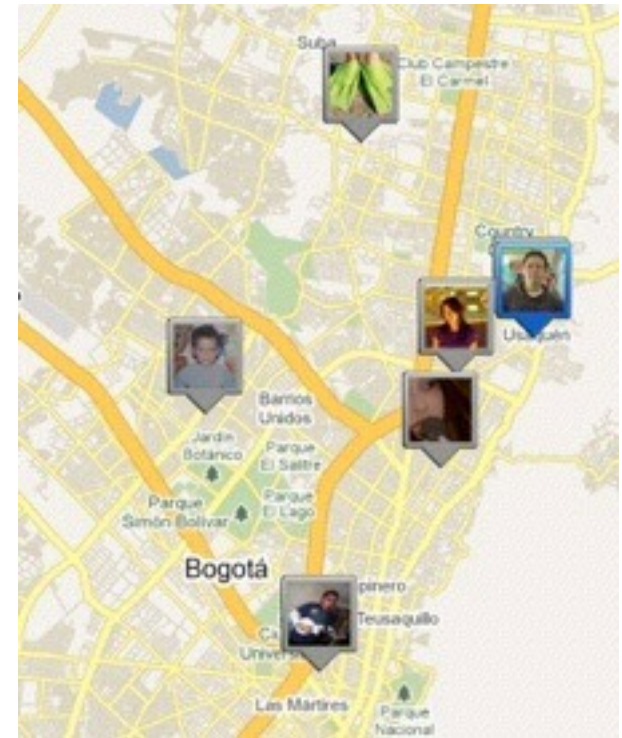




Geolocalización en HTML5

Geolocalización y Sensores

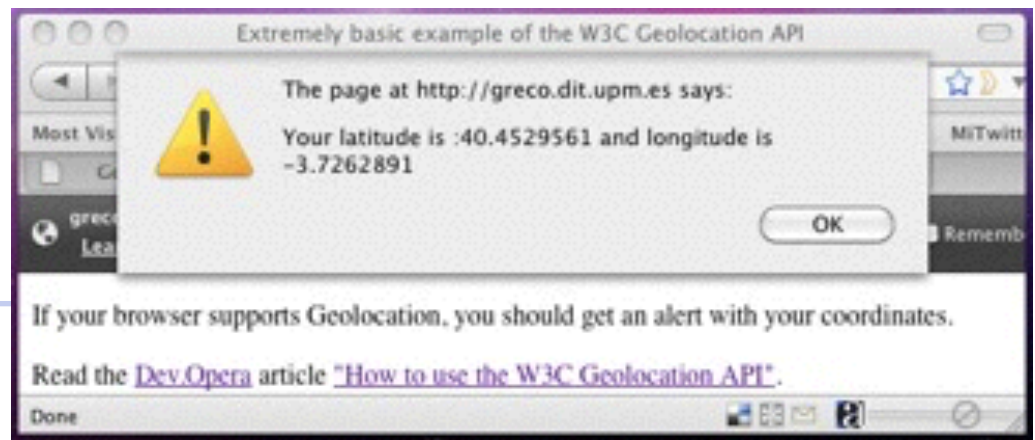
- ◆ HTML5 puede soportar geolocalización
 - En todo tipo de clientes
 - ◆ PCs, móviles tabletas,
- ◆ El interfaz de geolocalización
 - da acceso tambien a otros sensores
 - ◆ Brújula, acelerometro,



Geolocalización

- ◆ La geolocalización se realiza siguiendo jerarquía de consultas
 - GPS -> antena WIFI -> antena GSM o 3G -> IP fija ->
 - ◆ Se devuelve la respuesta más precisa
- ◆ La geolocalización está accesible en el objeto **navigator.geolocation**
 - con método **getCurrentPosition(successFunction, errorFunction)**
 - ◆ Permite conocer
 - Latitud y longitud en formato decimal
 - Altitud y precisión de la altitud
 - Dirección y velocidad
- ◆ Norma y tutoriales
 - <http://dev.w3.org/geo/api/spec-source.html>
 - <http://dev.opera.com/articles/view/how-to-use-the-w3c-geolocation-api/>
 - <http://code.google.com/apis/maps/index.html>
- ◆ **OJO!** Geolocalización puede no funcionar por restricciones de seguridad
 - Usar el navegador Firefox para probar los ejemplos geolocalizados en local

Ejemplo Geolocation



```
<!DOCTYPE html>
<html>
<head>
<title>Example of W3C Geolocation API</title>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<script type="text/javascript">

if (navigator.geolocation) { //Check if browser supports W3C Geolocation API
    navigator.geolocation.getCurrentPosition (successFunction, errorFunction);
} else { alert('Geolocation is not supported in this browser.')}

function successFunction(position) {
    var lat = position.coords.latitude;
    var long = position.coords.longitude;
    alert('Your latitude is :'+lat+' and longitude is '+long);
}

function errorFunction(position) { alert('Error!'); }
</script>
</head>
<body>
<p>If your browser supports Geolocation, you should get an alert with your coordinates.</p>

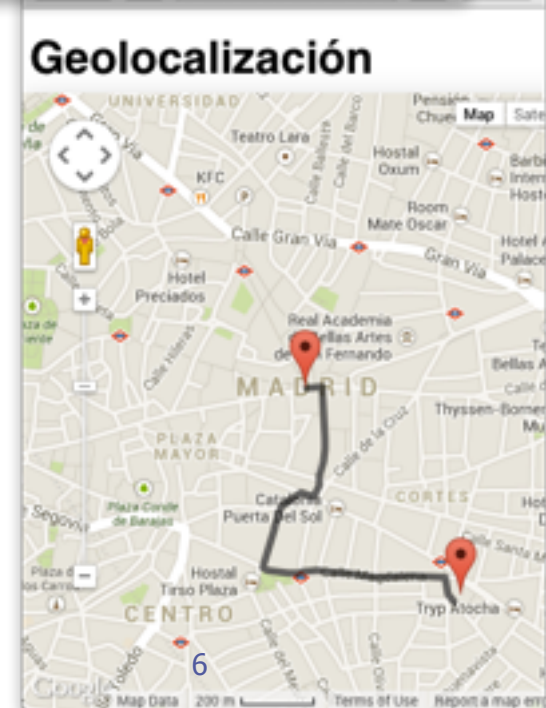
<p>Read the <a href="http://dev.opera.com">Dev.Opera</a> article <a
href="http://dev.opera.com/articles/view/how-to-use-the-w3c-geolocation-api/">"How
to use the W3C Geolocation API"</a>.
</body>
</html>
```



Aplicación HTML5 geolocalizada en Google Maps

Geolocalización con gmaps.js

- ◆ Aplicación de geolocalización
 - Carga un mapa centrado en nuestra posición
 - ◆ que se indica con un marcador
- ◆ Usamos librería gmaps.js para acceso a Google Maps
 - librería muy potente y sencilla de utilizar
 - ◆ <http://hpneo.github.io/gmaps/>
 - Se recomienda consultar documentación y ejemplos
 - La librería de Google es bastante mas compleja
- ◆ Se añade al mapa un manejador de eventos click/tap
 - que calcula la ruta hasta el punto indicado



Geo-mapa

```
<link rel="stylesheet" type="text/css" href="mypath.css" />
```

```
<script type="text/javascript" src="http://maps.google.com/maps/api/js?sensor=true"></script>  
<script type="text/javascript" src="gmaps.js"></script>
```

```
<script type="text/javascript" src="zepto.min.js"></script>  
<script type="text/javascript">  
  var map, lat, lng;
```

```
$(function(){
```

```
function geolocalizar(){  
  GMaps.geolocate({  
    success: function(position){  
      lat = position.coords.latitude; // guarda coords en lat y lng  
      lng = position.coords.longitude;  
  
      map = new GMaps({ // muestra mapa centrado en coords [lat, lng]  
        el: '#map',  
        lat: lat,  
        lng: lng  
      });  
      map.addMarker({ lat: lat, lng: lng}); // marcador en [lat, lng]  
    },  
    error: function(error) { alert('Error: '+error.message); },  
    not_supported: function(){ alert("No soporta geolocalización"); },  
  });  
};
```

```
geolocalizar();
```

```
});
```

```
</script>
```

```
</head><body>
```

```
<h1>Geolocalización</h1>
```

```
<div id="map"></div>
```

```
</body></html>
```



Geo-mapa

```
<script type="text/javascript">
  var map, distancia, lat, lng;

  $(function(){

    function enlazarMarcador(e){
      .....
    };

    function geolocalizar(){
      GMaps.geolocate({
        success: function(position){
          lat = position.coords.latitude; // guarda coords en lat y lng
          lng = position.coords.longitude;

          map = new GMaps({ // muestra mapa centrado en coords [lat, lng]
            el: '#map',
            lat: lat,
            lng: lng,
            click: enlazarMarcador, // eventos click y tap sobre el mapa
            tap: enlazarMarcador
          });
          map.addMarker({ lat: lat, lng: lng}); // marcador en [lat, lng]
        },
        error: function(error) { alert('Error: '+error.message); },
        not_supported: function(){ alert("No soporta geolocalización"); },
      });

      geolocalizar();
    });
  });
</script>
```




```
<script type="text/javascript">
  var map, distancia, lat, lng;
```

```
$(function(){
```

```
function enlazarMarcador(e){
```

```
  // muestra ruta entre marcas anteriores y actuales
```

```
  map.drawRoute({
    origin: [lat, lng], // origen en coordenadas anteriores
    // destino en coordenadas del click o toque actual
    destination: [e.latLng.lat(), e.latLng.lng()],
    travelMode: 'driving',
    strokeColor: '#000000',
    strokeOpacity: 0.6,
    strokeWeight: 5
  });
```

```
  lat = e.latLng.lat(); // guarda coords para marca siguiente
  lng = e.latLng.lng();
```

```
  map.addMarker({ lat: lat, lng: lng}); // pone marcador en mapa
};
```

```
function geolocalizar(){
```

```
  ....
  click: enlazarMarcador, // eventos click y tap sobre el mapa
  tap: enlazarMarcador
  ....
};
```

```
geolocalizar();
```

```
});
```

```
</script>
```

Geo-mapa



```
body{
  font-family: 'Droid Sans', 'Helvetica', Arial, sans-serif;
}
```

```
#latlng{
  display: block;
  margin: 0;
  padding: 0;
```

```
  position: absolute; /* posición absoluta a navegador */
  top: 0; /* ajusta a borde de navegador */
  left: 0;
  right: 0;
  height: 50px;
```

```
#map{
  display: block;
  margin: 0;
  padding: 0;
```

```
  position: absolute; /* posición absoluta a navegador */
  top: 50px; /* 50px debajo de borde de navegador */
  left: 0; /* ajusta a borde de navegador */
  right: 0;
  bottom: 0;
  background: rgba(0,255,0,0.5); /* verde si no hay mapa */
```

Estilo CSS multi-terminal

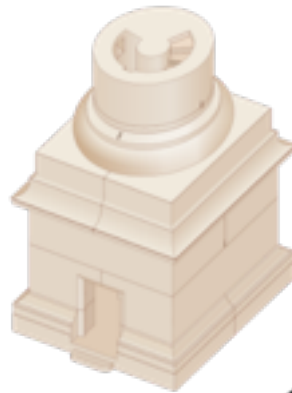




HTML5 SVG: Scalable Vector Graphics

SVG: Scalable Vector Graphics

- ◆ Formato de representación de gráficos vectoriales
 - Pueden cambiar de tamaño sin pérdida de calidad
- ◆ Recursos
 - Galeria Wikimedia: http://commons.wikimedia.org/wiki/Category:SVGs_by_subject
 - Editor SVG: <http://svg-edit.googlecode.com/svn/branches/2.5.1/editor/svg-editor.html>
 - Tutorial: <https://developer.mozilla.org/en-US/docs/Web/SVG>
 - Tutorial: <http://www.w3schools.com/svg/>



<http://commons.wikimedia.org/wiki/File:Compass.svg>

http://commons.wikimedia.org/wiki/SVG_examples

Ejemplo "Ajuste SVG"

- ◆ **"Ajuste SVG"** ilustra como reescalar una imagen SVG
 - Las imágenes en SVG reescalan sin perder calidad
 - ◆ porque son gráficos vectoriales
 - ◆ tutorial: <http://www.w3schools.com/svg/>
 - Las imágenes GIT, JPEG o PNG no reescalan bien
 - ◆ porque tienen una resolución determinada
- ◆ Esta WebApp tiene 2 botones: "+" y "-"
- ◆ Cada vez que pulsamos uno de estos botones
 - el tamaño de la imagen debe aumentar o disminuir un 10%
 - ◆ según pulsemos "+" y "-"




```

<!DOCTYPE html>
<html><head><title>Ejemplo SVG</title>
<script type="text/javascript"
  src="zepto.min.js" > </script>
<script type="text/javascript">
$(function(){
  var img = $('#img');

  $('#incr').on('click', function(){
    img.width(img.width()*1.1);
    img.height(img.height()*1.1);
  });

  $('#decr').on('click', function(){
    img.width(img.width()/1.1);
    img.height(img.height()/1.1);
  });
});
</script>
</head>
<body>
<h4> Ejemplo SVG </h4>
<button type="button" id="decr">-</button>
<button type="button" id="incr">+</button><p>



</body>
</html>

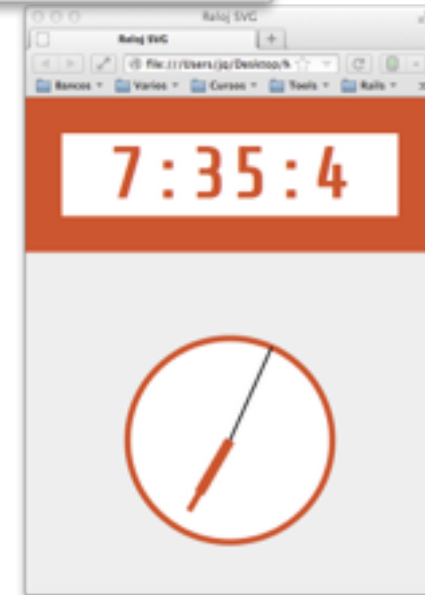
```

Ejemplo SVG



Ejemplo "Reloj SVG"

- ◆ **"Reloj SVG"** genera un reloj sencillo con SVG
 - El reloj se compone de
 - ◆ Un círculo negro
 - ◆ Tres líneas para las manecillas del reloj
- ◆ SVG puede animarse con JavaScript
 - modificando la representación DOM del reloj
 - ◆ Versión 1: las manecillas se mueven con transform
 - ◆ <https://developer.mozilla.org/en-US/docs/Web/SVG/Attribute/transform>
 - ◆ Version 2: Calcula las coordenadas de las manecillas
- ◆ Se añade estilo CSS
 - Mejora el aspecto y adapta al tamaño de la pantalla



```
<!DOCTYPE html>
<html>
<head><title>Reloj SVG</title>
  <meta charset="UTF-8"></head>
<body>
<h3>Reloj SVG</h3>
<div id="tex">texto</div>

<svg>
  <circle id="myCircle"
    cx="80" cy="80" r="50"
    fill="white" stroke="black" stroke-width="3"/>
  <line id="hor"
    x1="80" y1="80" x2="110" y2="80"
    style="stroke:grey;stroke-width:5"/>
  <line id="min"
    x1="80" y1="80" x2="80" y2="40"
    style="stroke:grey;stroke-width:3"/>
  <line id="seg"
    x1="80" y1="80" x2="115" y2="45"
    style="stroke:red;stroke-width:1"/>
</svg>

</body>
</html>
```



Reloj SVG

```

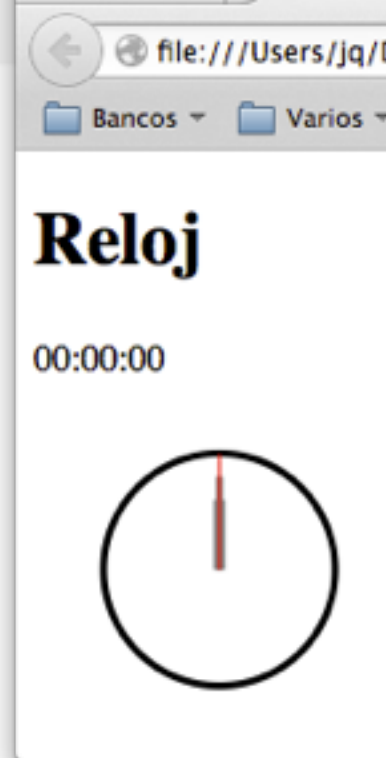
<head><title>Galeria</title><meta charset= UTF-8 >
<script type="text/javascript" src="http://zeptojs.com/zepto.min.js" ></script>
<script>
function animar() {
    var d = new Date();
    var s = d.getSeconds(); // grados = segundos * 6
    var m = d.getMinutes(); // grados = minutos * 6
    var h = d.getHours();
    var hh = h*30 + m/2; // grados de la manecilla de horas
    $("#tex").html(h + ":" + m + ":" + s);
    $("#hor").attr("transform", "rotate(" + hh + " 80 80)");
    $("#min").attr("transform", "rotate(" + m*6 + " 80 80)");
    $("#seg").attr("transform", "rotate(" + s*6 + " 80 80)");
}
$(function(){
    setInterval(animar, 1000);
    animar();
})
</script>
</head>
<body>
<h1>Reloj</h1>

<div id="tex">texto</div>

<svg>
<circle id='myCircle' cx='80' cy='80' r='50'
    fill='white' stroke='black' stroke-width='3' />
<line id="hor" x1='80' y1='80' x2='80' y2='50'
    style='stroke:grey;stroke-width:5' />
<line id="min" x1='80' y1='80' x2='80' y2='40'
    style='stroke:grey;stroke-width:3' />
<line id="seg" x1='80' y1='80' x2='80' y2='30'
    style='stroke:red;stroke-width:1' />
</svg></body></html>

```

SVG: Reloj animado con "transform"



Animar manecillas con coordenadas

- ◆ Para animar las manecillas del reloj
 - se incluye un script que cada segundo
 - ◆ recalcula las coordenadas exteriores
 - de las manecillas del reloj
 - El secundero tiene una longitud de 50 pixels
 - El minuterero tiene una longitud de 40 pixels
 - La manecilla horaria de 30 pixels
- ◆ Las coordenadas x_2 , y_2 de las manecillas de horas, minutos y segundos se calculan con las funciones
 - $x_2(\text{tiempo}, \text{unidades_por_vuelta}, x_1, \text{radio})$
 - $y_2(\text{tiempo}, \text{unidades_por_vuelta}, y_1, \text{radio})$



SVG: Reloj animado con coordenadas

```
<!DOCTYPE html>
<html>
<head>
  <title>Reloj SVG</title>
  <meta charset="UTF-8">
  <script type="text/javascript" src="http://zeptojs.com/zepto.min.js" >
</script>
```

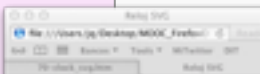
```
<script type="text/javascript">
```

```
function x2(n,i,x1,r) {return x1 + r*Math.sin(2*Math.PI*n/i)};
function y2(n,i,y1,r) {return y1 - r*Math.cos(2*Math.PI*n/i)};
```

```
function mostrar_hora( ) {
  var d = new Date();
  var h = d.getHours();
  var m = d.getMinutes();
  var s = d.getSeconds();
  $('#tex').html(h + ":" + m + ":" + s);
  $('#seg').attr('x2', x2(s,60,80,50)).attr('y2', y2(s,60,80,50));
  $('#min').attr('x2', x2(m,60,80,40)).attr('y2', y2(m,60,80,40));
  $('#hor').attr('x2', x2(h,12,80,30)).attr('y2', y2(h,12,80,30));
}
```

```
$(function(){
  setInterval(mostrar_hora, 1000);
  mostrar_hora();
})
```

```
</script>
```



Reloj SVG

23:6:22



Relojes con "estilo"



- ◆ Usando CSS e imágenes se pueden diseñar
 - Las capturas muestran pequeños cambios de diseño
 - ◆ que cambian muy significativamente la apariencia del reloj
 - Hacer clic en estos URLs para verlos
 - ◆ https://googledrive.com/host/0B48KCWfVwCIEMjFhUHM4d3FnSTg/09-clock_CSS.htm
 - ◆ https://googledrive.com/host/0B48KCWfVwCIEMjFhUHM4d3FnSTg/10_clock_CSS_a.html
 - ◆ https://googledrive.com/host/0B48KCWfVwCIEMjFhUHM4d3FnSTg/10_clock_CSS_b.htm
 - ◆ https://googledrive.com/host/0B48KCWfVwCIEMjFhUHM4d3FnSTg/10_clock_CSS_c.htm
 - ◆ https://googledrive.com/host/0B48KCWfVwCIEMjFhUHM4d3FnSTg/10_clock_CSS_d.htm

```
<!DOCTYPE html>
<html><head><title>Reloj SVG</title><meta charset="UTF-8">
<style type="text/css">
```

SVG: Reloj con estilo CSS

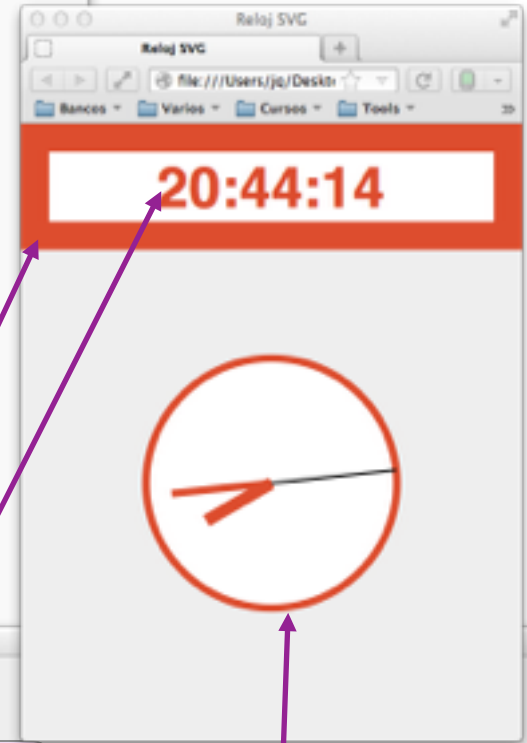
```
body, html {
  margin: 0px;
  padding: 0px;
  height: 100%;
  width: 100%;
  background-color: #eee;
}
```

```
.mitad {
  background-color: #db4e36;
  color: #db4e36;
  font-family: sans-serif;
  font-size: 5em;
  font-weight: bold;
  text-align: center;
  padding: 0.5em;
}
```

```
#tex {
  padding-top: 0.2em;
  background-color: #FFF;
}
```

```
#reloj {
  height: 100%;
  width: 100%;
}
```

```
</style>
....
</html>
```

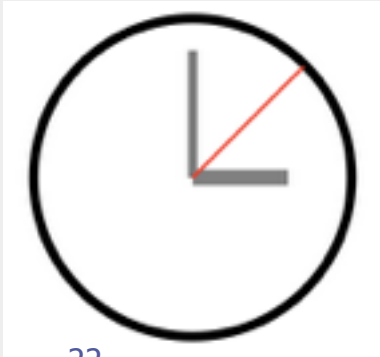


```
.....
<body>
  <div class="mitad">
    <div id="tex">texto</div>
  </div>
  <svg id="reloj" viewBox="0 0 200 200">
    <circle id="myCircle" cx="100" cy="70" r="50"
      fill="white" stroke="#db4e36" stroke-width="5" />
    <line id="hor" x1="100" y1="70" x2="110" y2="70"
      style="stroke:#db4e36;stroke-width:5" />
    <line id="min" x1="100" y1="70" x2="80" y2="70" />
  </svg>
</body>
```

Objetos SVG

- ◆ Los objetos SVG se pueden definir también como objetos externos en XML
 - Para importarlos con:
 - ◆ ``, `<object>`, `<embed>`, `<iframe>`
 - Tutorial: <http://tavmjong.free.fr/INKSCAPE/MANUAL/html/Web-Use.html>

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.0//EN"
    "http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd">
<svg xmlns="http://www.w3.org/2000/svg" width="120" height="120">
  <circle id='myCircle' cx='60' cy='60' r='50'
    fill='white' stroke='black' stroke-width='3' />
  <line x1='60' y1='60' x2='90' y2='60'
    style='stroke:grey;stroke-width:5' />
  <line x1='60' y1='60' x2='60' y2='20'
    style='stroke:grey;stroke-width:3' />
  <line x1='60' y1='60' x2='95' y2='25'
    style='stroke:red;stroke-width:1' />
</svg>
```



© Juan Quemada, DIT, UPM



Final del tema
Muchas gracias!

