

Javascript, parte III

Geolocalización

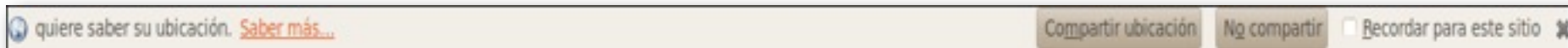


Averiguando dónde está el usuario

- API estándar de HTML5 que nos permite averiguar la posición geográfica del usuario
 - Hay métodos más precisos (GPS) y menos (a partir de la dirección IP o usando la red GSM)
 - El método exacto por el que se averigua la posición es transparente al desarrollador Javascript
 - Lo único que nos da el API son las coordenadas. Necesitaremos algún servicio adicional dsi queremos dibujar un mapa con la posición, etc. (p.ej. Google Maps)
- Este API no funciona en Explorer 8 y anteriores. Se pueden usar librerías alternativas, como Google Gears (funciona, pero el API es distinto)

Ejemplo simple

- Sin chequeo de errores ni opciones de localización
- `navigator.geolocation.getCurrentPosition()` obtiene la posición:
 - pero no la devuelve directamente. Hay que pasarle el nombre de una función que hará de *callback* (recibirá la posición en un parámetro)
 - La posición recibida es un objeto con dos campos: `coords` (con info como `latitude`, `longitude`, etc) y `timestamp`
 - Antes de obtener la posición el navegador va a pedir permiso al usuario. Si no se concede, el *callback* no se ejecuta



```
navigator.geolocation.getCurrentPosition(verPos);
function verPos(pos) {
    alert("Estás en (" + pos.coords.latitude + "," + pos.coords.longitude + ")");
}
```

- Se le puede pasar un segundo argumento a `getCurrentPosition`: una función que se llamará si se ha producido algún error
 - Por ejemplo el usuario no ha dado permiso, o no hay dispositivos de localización
 - La función recibe un argumento que es un objeto con dos campos. El más interesante es `code`, un código de error: 1:permiso denegado, 2:No se puede calcular la posición, 3:Timeout, 0>Error desconocido

```
navigator.geolocation.getCurrentPosition(verPos, verError);  
function verPos(pos) {  
    ...  
}  
  
function verError(error) {  
    if (error.code == 1)  
        alert("No has dado permiso para ver tu posición")  
}
```

- Tercer parámetro (opcional) de `getCurrentPosition`: objeto con tres campos:
 - `enableHighAccuracy` (booleano): indica si queremos una localización de precisión (p.ej. GPS) o nos basta con una aproximada (p.ej. usando la red de móvil)
 - `timeout` (nº en milisegundos) tiempo que estamos dispuestos a esperar que el dispositivo nos dé una posición. Pasado este tiempo se generará un error de timeout
 - `maximumAge` (nº en milisegundos) si el dispositivo tiene en cache una posición con una antigüedad inferior a esta, nos vale, no es necesario que calcule la actual.

```
//queremos alta precisión pero nos vale con la posición de hace un minuto
navigator.geolocation.getCurrentPosition(verPos, verError,
                                         {enableHighAccuracy: true, maximumAge:60000});
```

- Usar `watchPosition()` en lugar de `getCurrentPosition()` para obtener la posición cada X segundos
 - El navegador detecta cuándo ha cambiado la posición y solo llama al callback en ese caso
 - El callback se llamará varias veces (cada vez que cambie la posición)
 - `watchPosition()` devuelve un entero que podemos usar como parámetro de `clearWatch()` para que deje de llamar al callback a partir de este momento
- Como veremos, esto también se podría hacer con un temporizador (permite llamar a una función arbitraria cada X segundos), pero es más fácil y más eficiente usar `watchPosition`