¿Para que sirve HTML5 visibility API?. Al principio nos puede parecer un API un poco tonta que ya simplemente se encarga de comprobar si la ventana de navegador que tenemos activa esta visible o no. Sin embargo esto puede llegar a ser crítico a nivel de rendimiento. ¿Por qué? , muy sencillo ,cada día los usuarios tenemos mas pestañas de navegador abiertas. Vamos a ver un ejemplo que nos clarifique**:**

```
<html>
<head>
<script src="jquery-1.11.1.js"></script>
<script type="text/javascript">

$(document).ready(function() {

var i=0;
setInterval(function() {

$("body").append("<p>iteracion"+ i+"</p>");
i++;

},1000);

});
</script>
</head>
<body>
</body>
</html>
```

Se trata de una aplicación sencilla que usa un setInterval para ir imprimiendo párrafos.

iteracion0

iteracion1

iteracion2

Supongamos que cambiamos de pestaña de navegador y nos ponemos a leer un artículo cualquiera. Cuando volvamos a nuestra aplicación el numero de mensajes habrá crecido .

iteracion0

iteracion1

iteracion2

iteracion3

iteracion4

iteracion5

iteracion6

iteracion7

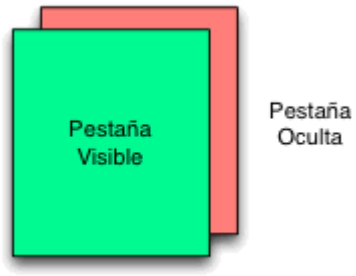
iteracion8

iteracion9

iteracion10

iteracion11

¿Era esto necesario?. La respuesta es que depende , pero hay que reconocer que en muchos casos no hacía falta actualizar la página si el usuario no la esta observando en ese momento. Si el usuario deja de mirar la página de bolsa con las cotizaciones , no es muy útil que en background se la sigamos actualizando. ¿Cómo podemos solventar esto?



HTML5 Visibility API al rescate

La solución es usar el API de Visibility de HTML 5 y cambiar el código por el siguiente:

```
<html>
<head>
<script src="jquery-1.11.1.js"></script>
<script type="text/javascript">

$(document).ready(function() {

var i=0;
setInterval(function() {

var visible= document.hidden;

if (!visible) {

$("body").append("<p>iteracion"+ i+"</p>");
i++;
}
```

```
},1000);

});
</script>
</head>
<body>
</body>
</html>
```

De esta forma las iteraciones solo avanzarán cuando estemos en la propia página. HTML Visibily API se usa mucho para mejorar la gestión de peticiones AJAX y mejorar el rendiento.

Otros artículos relacionados:HTML5 WebWorker , HTML5 Prefeching , LocalStorage