3 de julio de 2022

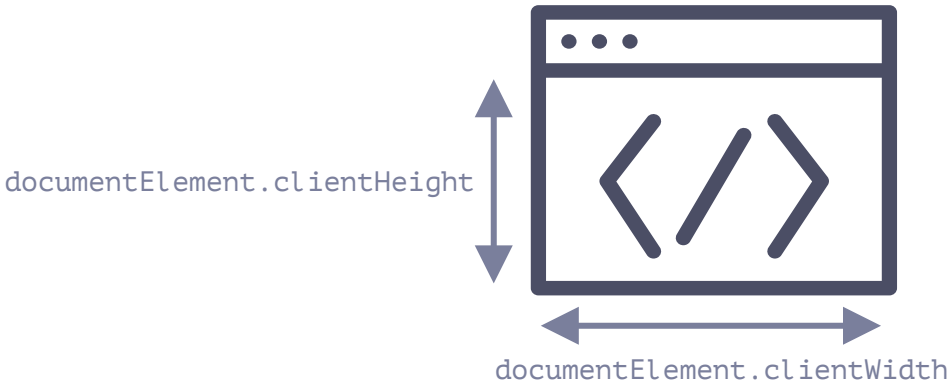
Tamaño de ventana y desplazamiento

¿Cómo encontramos el ancho y el alto de la ventana del navegador? ¿Cómo obtenemos todo el ancho y la altura del documento, incluida la parte desplazada? ¿Cómo desplazamos la página usando JavaScript?

Para la mayoría de estas cuestiones, podemos usar el elemento de documento raíz `document.documentElement`, que corresponde a la etiqueta `<html>`. Pero hay métodos y peculiaridades adicionales lo suficientemente importantes para considerar.

Ancho/alto de la ventana

Para obtener el ancho y alto de la ventana, podemos usar `clientWidth / clientHeight` de `document.documentElement`:



Por ejemplo, este botón muestra la altura de su ventana:

alert(document.documentElement.clientHeight)

No *window.innerWidth/Height*

Los navegadores también admiten propiedades `window.innerWidth / innerHeight`. Se parecen a lo que queremos. Entonces, ¿por qué no usarlos?

Si existe una barra de desplazamiento, y ocupa algo de espacio, `clientWidth / clientHeight` proporciona el ancho/alto sin ella (resta el espacio desplazado). En otras palabras, devuelven ancho/alto de la parte visible del documento, disponible para el contenido.

... Y `window.innerWidth / innerHeight` incluye la barra de desplazamiento.

Si hay una barra de desplazamiento y ocupa algo de espacio, estas dos líneas muestran valores diferentes:

```
1  alert( window.innerWidth ); // ancho de la ventana completa
2  alert( document.documentElement.clientWidth ); // ancho de ventana menos el
```

En la mayoría de los casos, necesitamos el ancho de ventana *disponible*, para dibujar o colocar algo. Es decir: el espacio del desplazamiento si hay alguno. Entonces deberíamos usar `documentElement.clientHeight/Width`.

DOCTYPE es importante

Tenga en cuenta que las propiedades de geometría de nivel superior pueden funcionar de manera un poco diferente cuando no hay `<!DOCTYPE HTML>` en HTML. Pueden suceder cosas extrañas.

En HTML moderno siempre debemos escribir `DOCTYPE`.

Ancho/Alto del documento

Teóricamente, como el elemento del documento raíz es `document.documentElement`, e incluye todo el contenido, podríamos medir el tamaño completo del documento con `document.documentElement.scrollWidth / scrollHeight`.

Pero en ese elemento, para toda la página, estas propiedades no funcionan según lo previsto. ¡En Chrome/Safari/Opera si no hay desplazamiento, entonces `documentElement.scrollHeight` puede ser incluso menor que `documentElement.clientHeight`! Suena como una tontería, raro, ¿verdad?

Para obtener de manera confiable la altura completa del documento, debemos tomar el máximo de estas propiedades:

```
1  let scrollHeight = Math.max(
2    document.body.scrollHeight, document.documentElement.scrollHeight,
3    document.body.offsetHeight, document.documentElement.offsetHeight,
4    document.body.clientHeight, document.documentElement.clientHeight
5  );
6
7  alert('Altura completa del documento, con parte desplazada: ' + scrollHeight);
```

¿Por qué? Mejor no preguntes. Estas inconsistencias provienen de tiempos antiguos, no una lógica "inteligente".

Obtener el desplazamiento actual

Los elementos DOM tienen su estado de desplazamiento actual en sus propiedades `elem.scrollLeft/scrollTop`.

El desplazamiento de documentos, `document.documentElement.scrollLeft / Top` funciona en la mayoría de los navegadores, excepto los más antiguos basados en WebKit, como Safari (bug 5991), donde deberíamos usar `document.body` en lugar de `document.documentElement`.

Afortunadamente, no tenemos que recordar estas peculiaridades en absoluto, porque el desplazamiento está disponible en las propiedades especiales `window.pageXOffset/pageYOffset`:

```
1  alert('Desplazamiento actual desde la parte superior: ' + window.pageYOffset);
2  alert('Desplazamiento actual desde la parte izquierda: ' + window.pageXOffset);
```

Estas propiedades son de solo lectura.

> **También disponible como propiedades `window`: `scrollX` y `scrollY`**
>
> Por razones históricas existen ambas propiedades, pero ambas son lo mismo:
>
> - `window.pageXOffset` es un alias de `window.scrollX`.
> - `window.pageYOffset` es un alias de `window.scrollY`.

Desplazamiento: scrollTo, scrollBy, scrollIntoView

> **Importante:**
>
> para desplazar la página desde JavaScript, su DOM debe estar completamente construido.
>
> Por ejemplo, si intentamos desplazar la página desde el script en `<head>`, no funcionará.

Los elementos regulares se pueden desplazar cambiando `scrollTop/scrollLeft`.

Nosotros podemos hacer lo mismo para la página usando `document.documentElement.scrollTop/Left` (excepto Safari, donde `document.body.scrollTop/Left` debería usarse en su lugar).

Alternativamente, hay una solución más simple y universal: métodos especiales window.scrollBy(x,y) y window.scrollTo(pageX,pageY).

- El método `scrollBy(x, y)` desplaza la página *en relación con su posición actual*. Por ejemplo, `scrollBy(0,10)` desplaza la página `10px` hacia abajo.

  El siguiente botón demuestra esto:

  window.scrollBy(0,10)

- El método `scrollTo(pageX, pageY)` desplaza la página *a coordenadas absolutas*, de modo que la esquina superior izquierda de la parte visible tiene coordenadas `(pageX, pageY)` en relación con la esquina superior izquierda del documento. Es como configurar `scrollLeft / scrollTop` .

  Para desplazarnos hasta el principio, podemos usar `scrollTo(0,0)` .

  window.scrollTo(0,0)

Estos métodos funcionan para todos los navegadores de la misma manera.

scrollIntoView

Para completar, cubramos un método más: elem.scrollIntoView(top).

La llamada a `elem.scrollIntoView(top)` desplaza la página para hacer visible `elem` . Tiene un argumento:

- si `top=true` (ese es el valor predeterminado), la página se desplazará para que aparezca `element` en la parte superior de la ventana. El borde superior del elemento está alineado con la parte superior de la ventana.
- si `top=false` , la página se desplaza para hacer que `element` aparezca en la parte inferior. El borde inferior del elemento está alineado con la parte inferior de la ventana.

El botón a continuación desplaza la página para mostrarse en la parte superior de la ventana:

this.scrollIntoView()

Y este botón desplaza la página para mostrarla en la parte inferior:

this.scrollIntoView(false)

Prohibir el desplazamiento

A veces necesitamos hacer que el documento sea "inescrutable". Por ejemplo, cuando necesitamos cubrirlo con un mensaje grande que requiere atención inmediata, y queremos que el visitante interactúe con ese mensaje, no con el documento.

Para hacer que el documento sea inescrutable, es suficiente establecer `document.body.style.overflow="hidden"` . La página se congelará en su desplazamiento actual.

Prueba esto:

document.body.style.overflow = 'hidden'

document.body.style.overflow = ''

El primer botón congela el desplazamiento, el segundo lo reanuda.

Podemos usar la misma técnica para "congelar" el desplazamiento para otros elementos, no solo para `document.body` .

El inconveniente del método es que la barra de desplazamiento desaparece. Si ocupaba algo de espacio, entonces ese espacio ahora es libre y el contenido "salta" para llenarlo.

Eso parece un poco extraño, pero puede solucionarse si comparamos `clientWidth` antes y después del congelamiento, y si aumentó (la barra de desplazamiento desapareció) luego agregue `padding` a `document.body` en lugar de la barra de desplazamiento, para que mantenga el ancho del contenido igual.

Resumen

Geometría:

- Ancho/alto de la parte visible del documento (área de contenido ancho/alto): `document.documentElement.clientWidth/Height`
- Ancho/alto de todo el documento, con la parte desplazada:

```
let scrollHeight = Math.max(
  document.body.scrollHeight, document.documentElement.scrollHeight,
  document.body.offsetHeight, document.documentElement.offsetHeight,
  document.body.clientHeight, document.documentElement.clientHeight
);
```

Desplazamiento:

- Lee el desplazamiento actual: `window.pageYOffset/pageXOffset` .
- Cambia el desplazamiento actual:
  - `window.scrollTo(pageX,pageY)` – coordenadas absolutas
  - `window.scrollBy(x,y)` – desplazamiento relativo al lugar actual,
  - `elem.scrollIntoView(top)` – desplácese para hacer visible el `elem` (alineación con la parte superior/inferior de la ventana).

Lección anterior | Próxima lección

Compartir

Mapa del Tutorial

Comentarios

- Si tiene sugerencias sobre qué mejorar, por favor enviar una propuesta de GitHub o una solicitud de extracción en lugar de comentar.
- Si no puede entender algo en el artículo, por favor explique.
- Para insertar algunas palabras de código, use la etiqueta `<code>` , para varias líneas – envolverlas en la etiqueta `<pre>` , para más de 10 líneas – utilice una entorno controlado (sandbox) (plnkr, jsbin, codepen...)

acerca del proyecto contáctenos