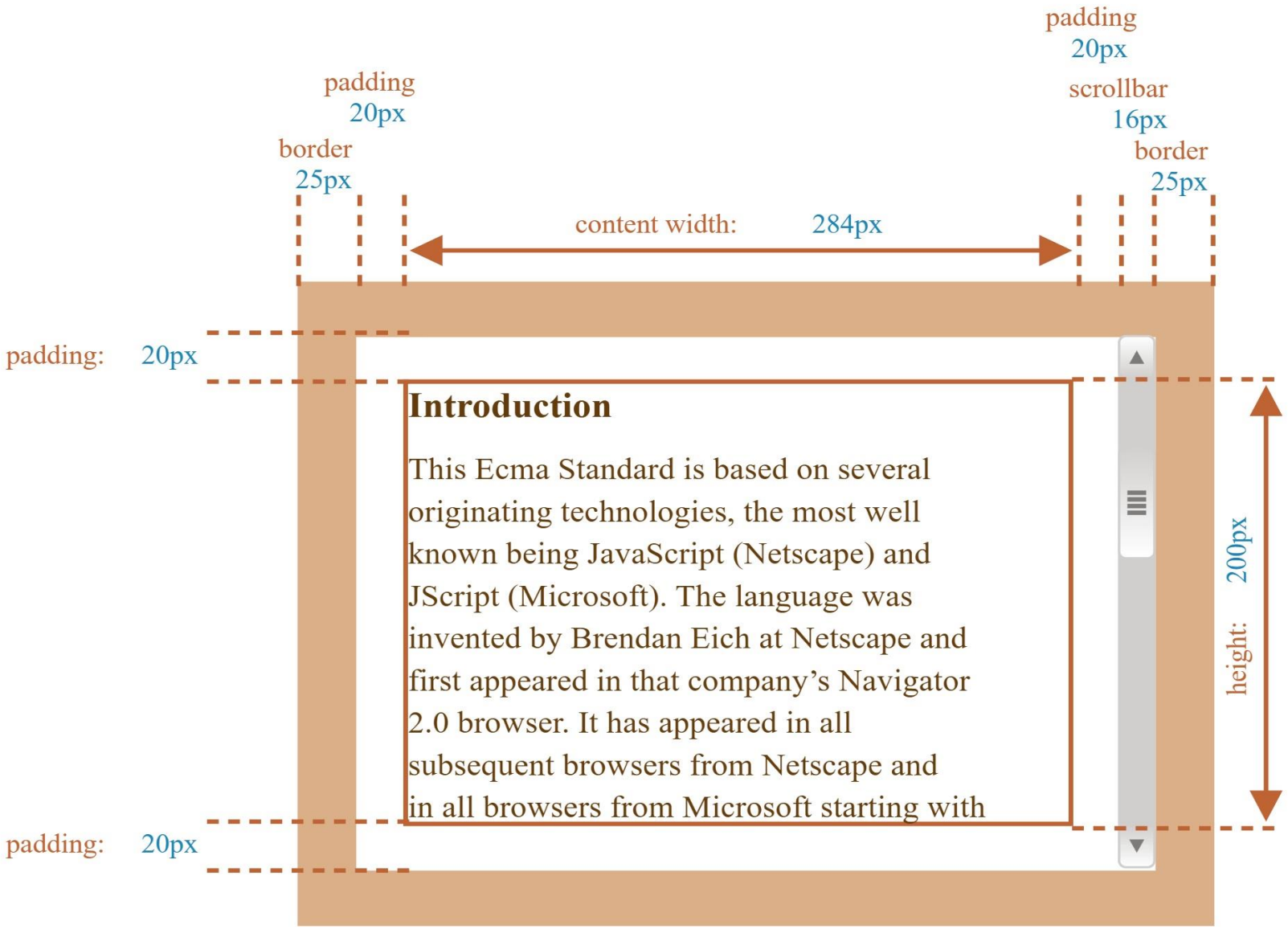
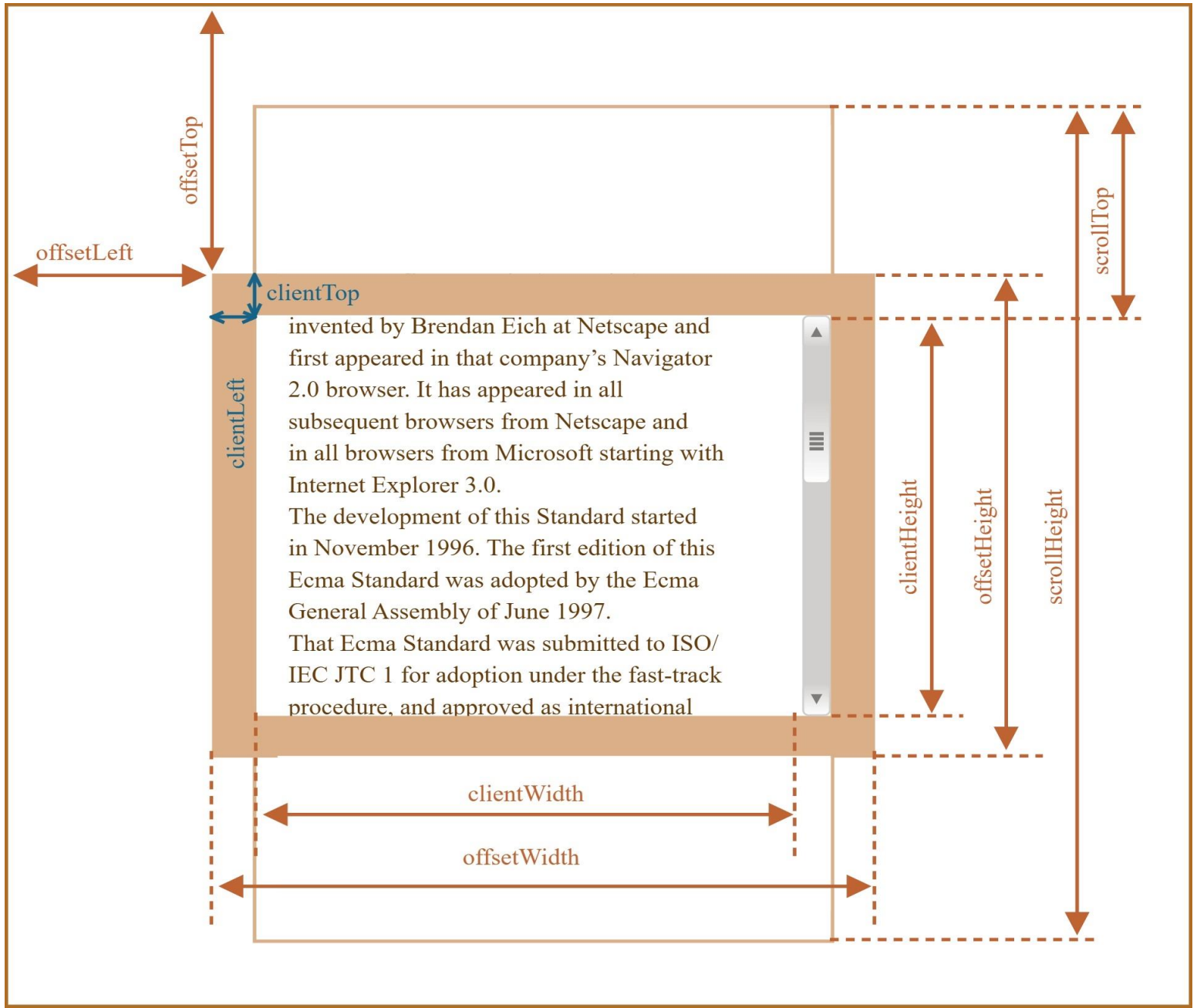
ELEMENT size and scrolling

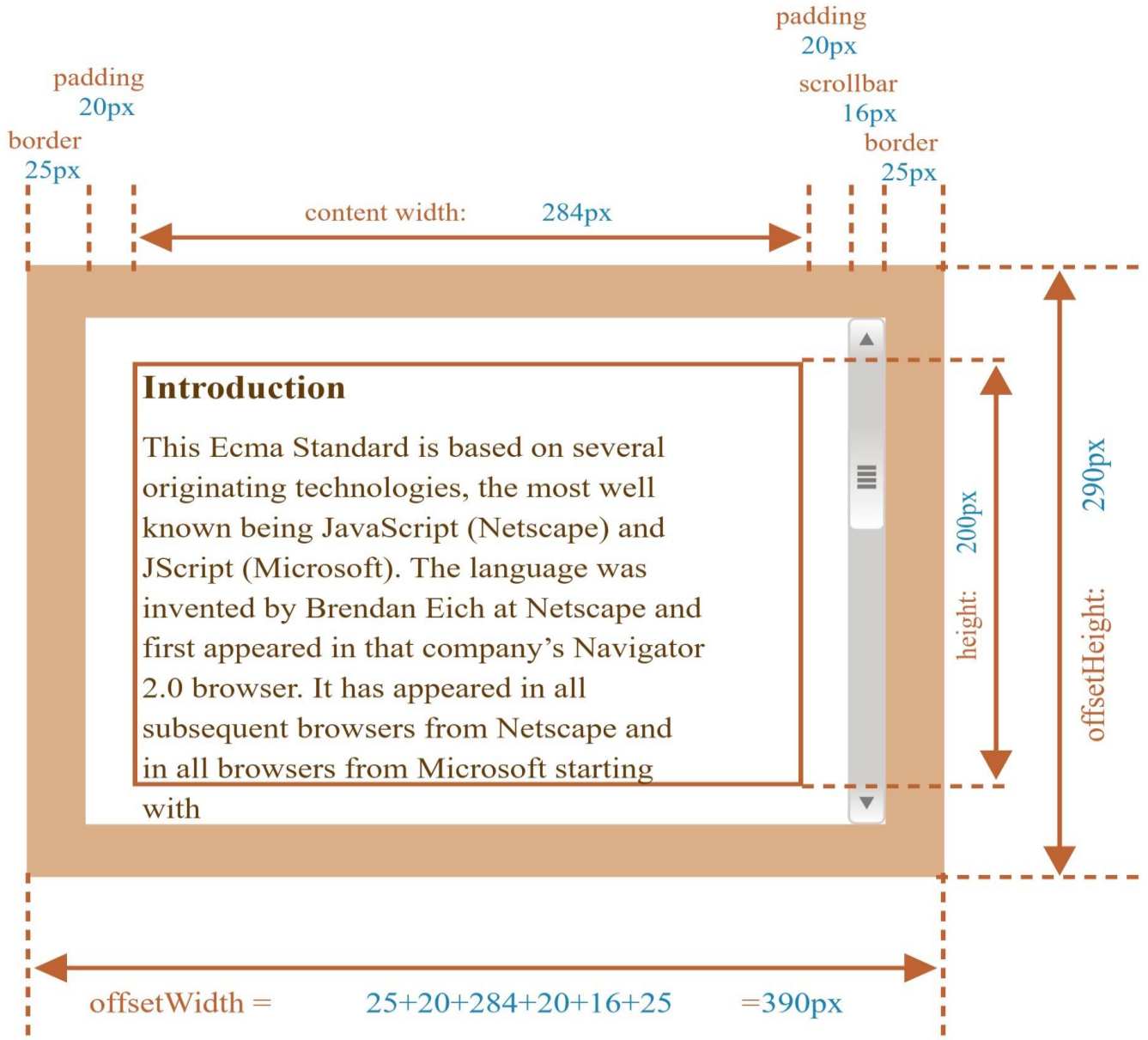


Geometry


offsetTop
offsetLeft
clientTop
clientLeft
invented by Brendan Eich at Netscape and first appeared in that company's Navigator 2.0 browser. It has appeared in all subsequent browsers from Netscape and in all browsers from Microsoft starting with Internet Explorer 3.0.
The development of this Standard started in November 1996. The first edition of this Ecma Standard was adopted by the Ecma General Assembly of June 1997.
That Ecma Standard was submitted to ISO/ IEC JTC 1 for adoption under the fast-track procedure, and approved as international
clientHeight
offsetHeight
scrollHeight
scrollTop
clientWidth
offsetWidth

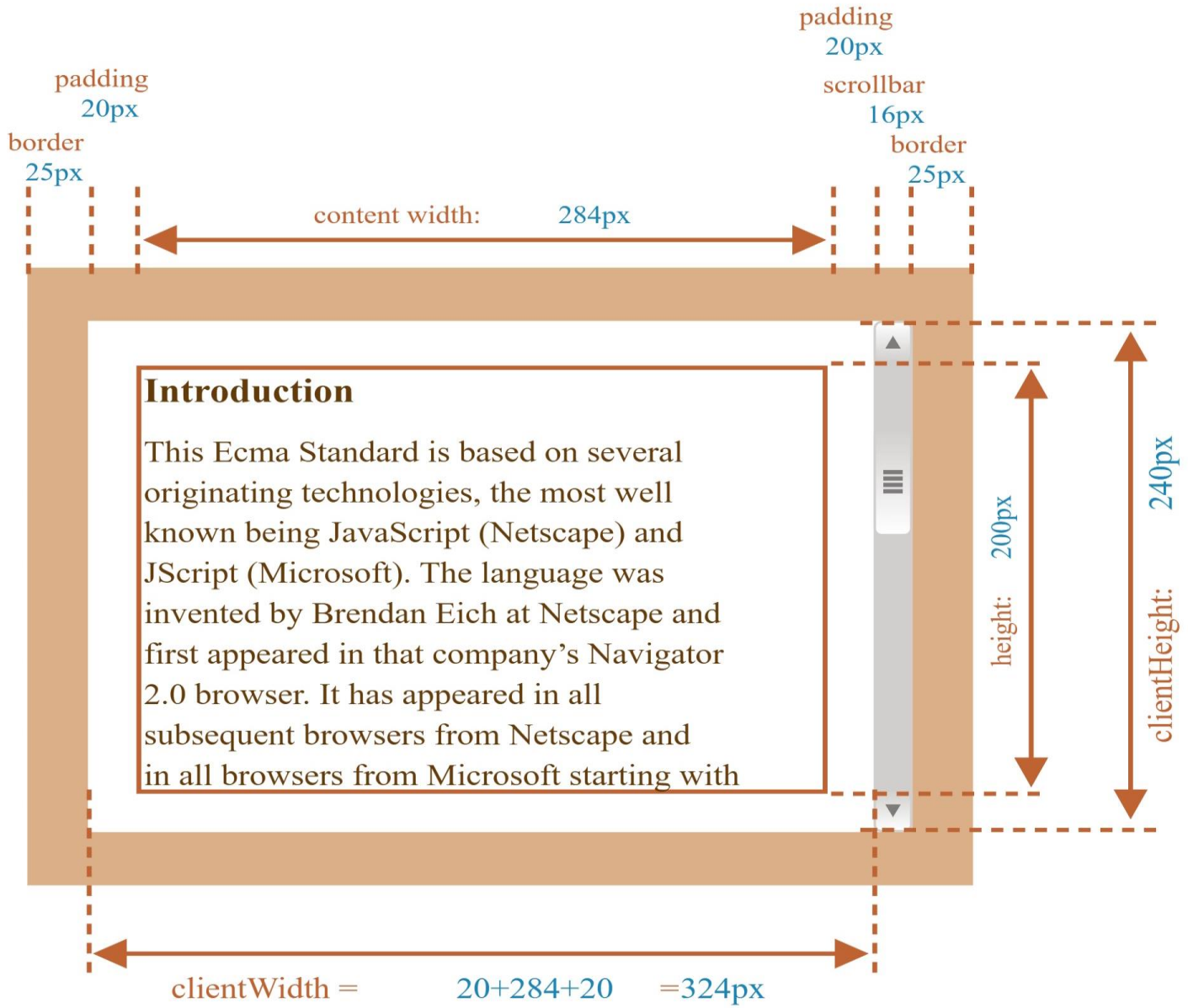
offsetWidth / offsetHeight

(outside properties)



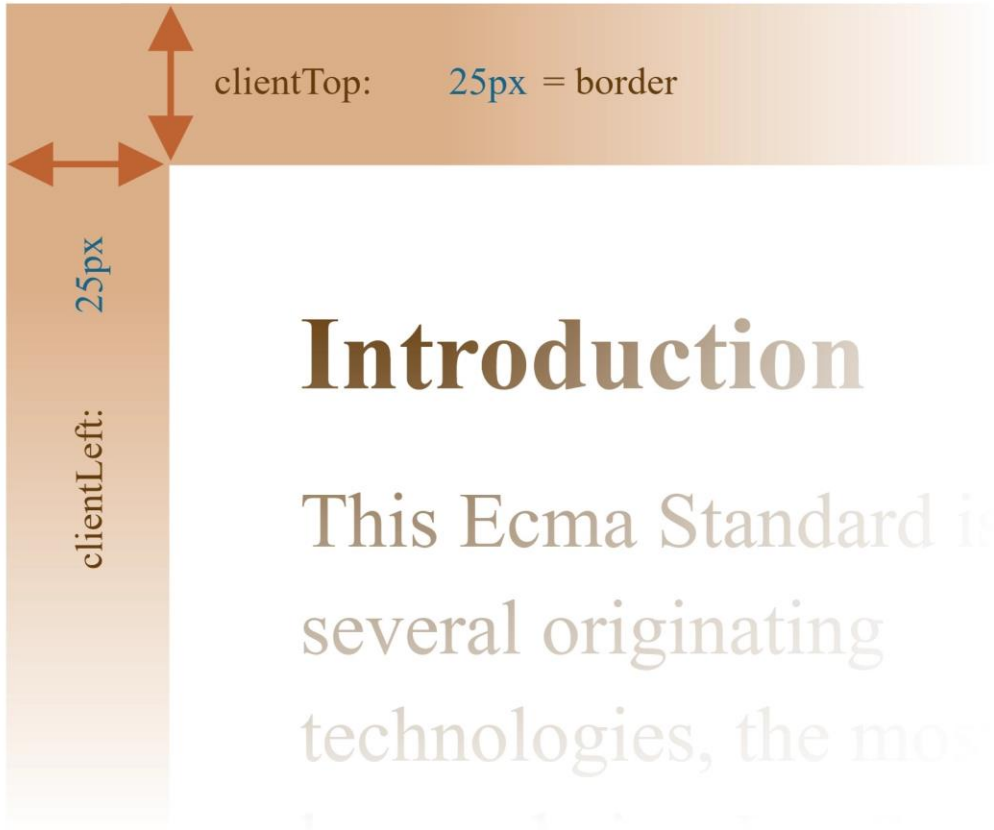
clientWidth / clientHeight

(including padding but without border and scrollbar)



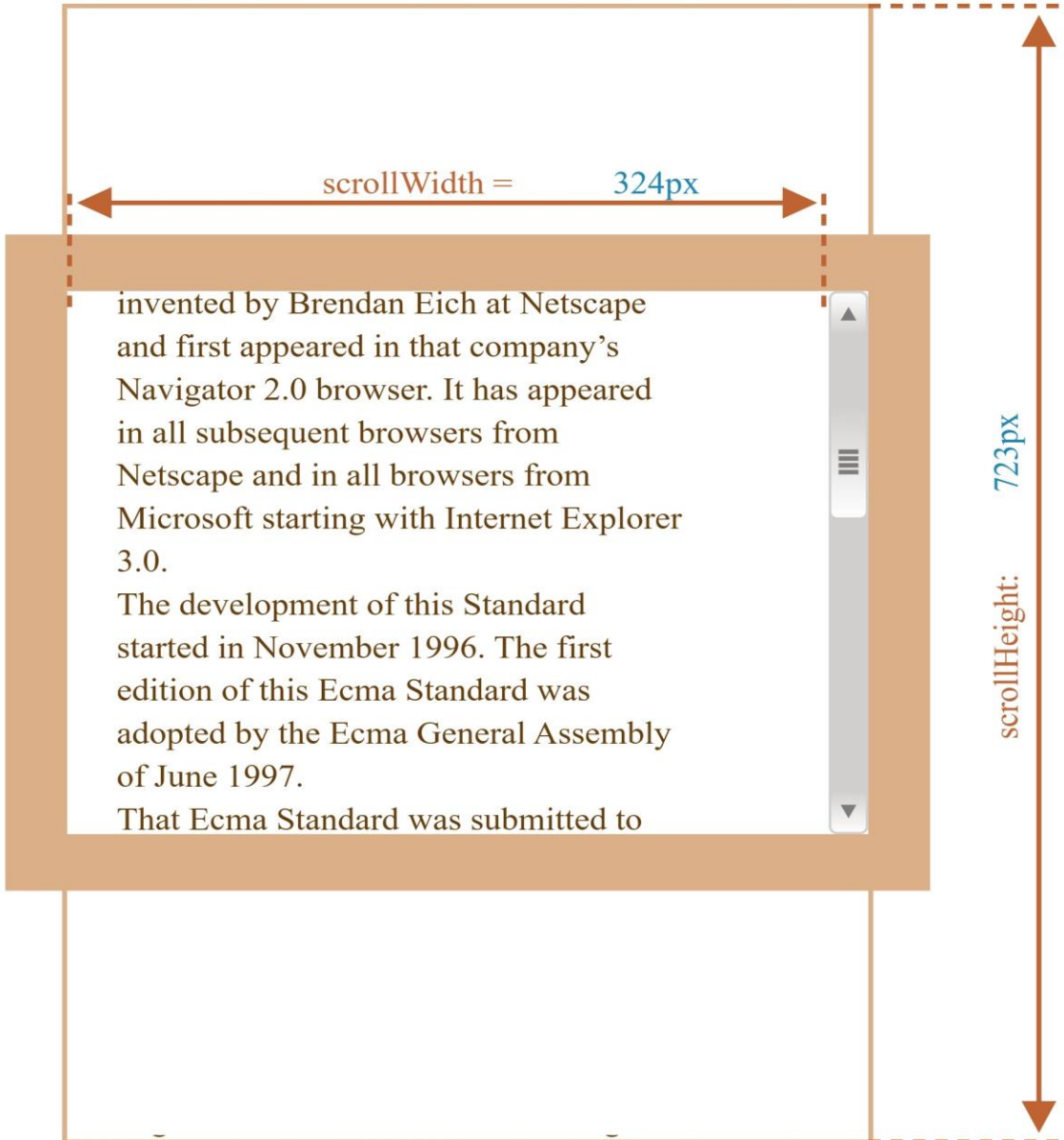
clientTop / clientLeft

(relative coordinates of the inner side from the outer side)



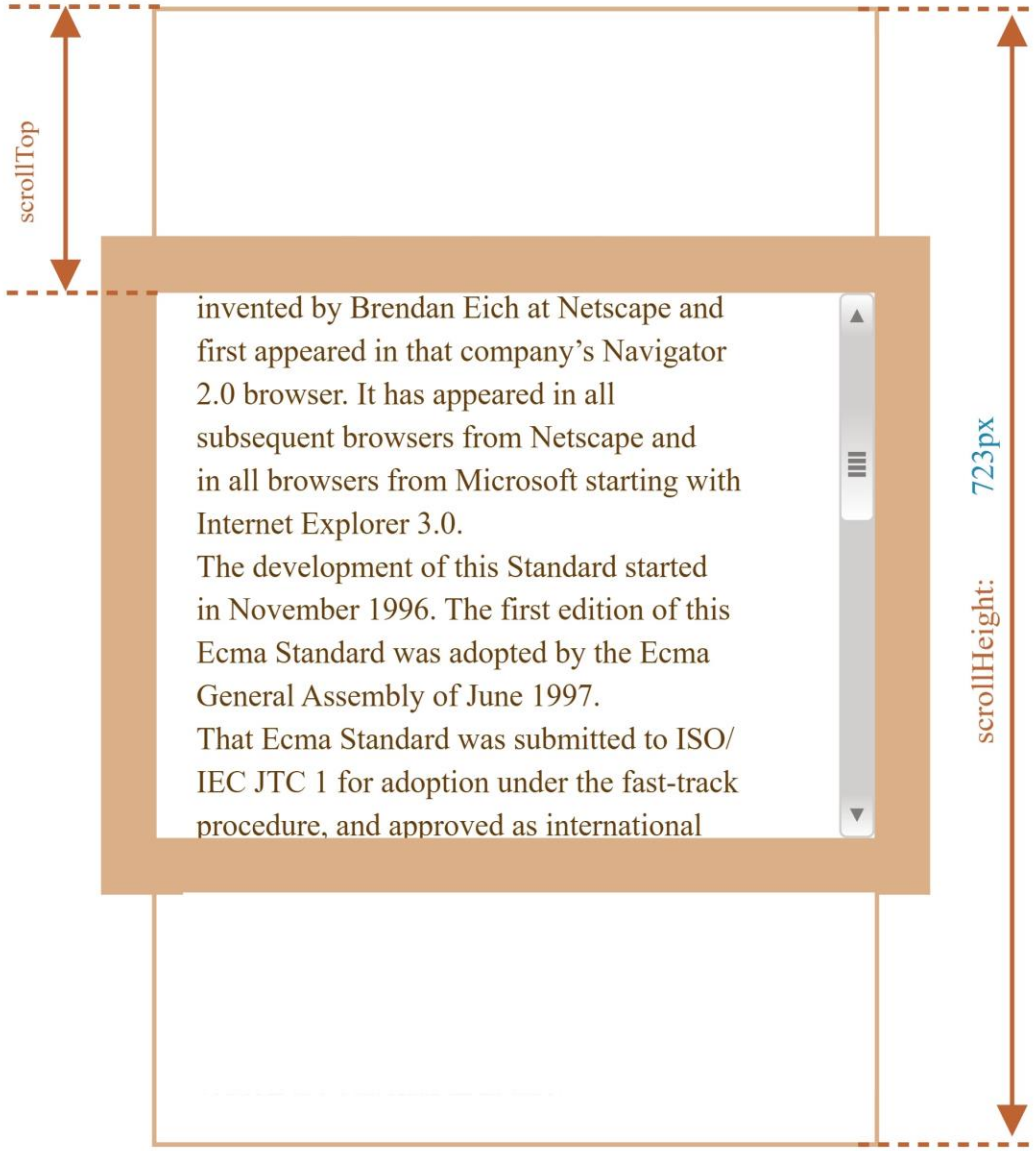
scrollWidth / scrollHeight

(these properties are like clientWidth / clientHeight, but they also include the scrolled out / hidden parts)



scrollLeft / scrollTop

(properties scrollLeft / scrollTop are the width / height of the hidden, scrolled out part of the element)



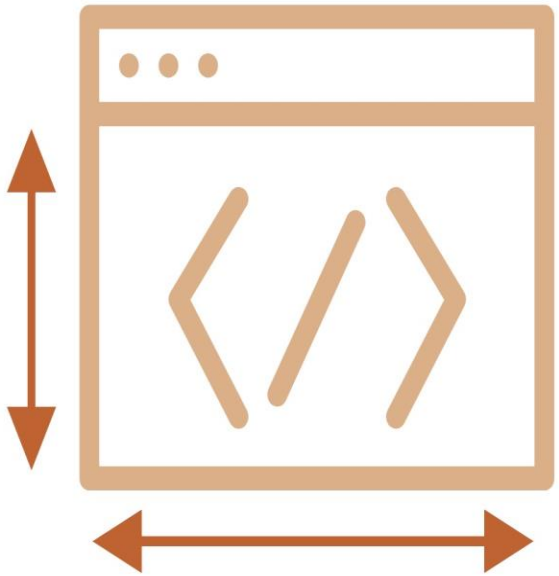
- In other words, scrollTop is "how much is scrolled up

  scrollLeft/scrollTop can be modified
- Most of the geometry properties here are read-only,
  but scrollLeft/scrollTop can be changed, and the browser will scroll the element.
  Setting scrollTop to 0 or a big value, such as 1e9 will make the element scroll to the very top/bottom respectively.

WINDOW sizes and scrolling

clientWidth / clientHeight vs innerWidth / innerHeight

documentElement.clientHeight



documentElement.clientWidth

- To get window width / height, we can use the clientWidth / clientHeight of document.documentElement
- **window**.innerWidth / **window**.innerHeight

If there exists a scrollbar, and it occupies some space, clientWidth / clientHeight provide the width/height without it. In other words, they return the width / height of the visible part of the document, available for the content

window.innerWidth / innerHeight includes the scrollbar

WINDOW scrolling

For historical reasons, both properties exist, but they are the same

- window.pageXOffset --> window.scrollX
- window.pageYOffset --> window.scrollY

window scrolling:

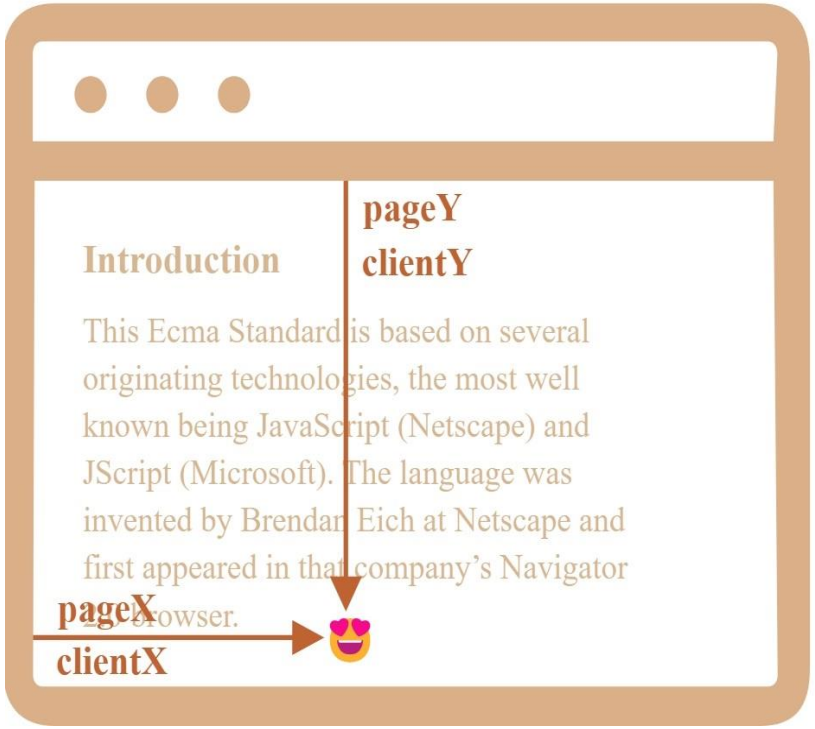
- scrollTo(x, y) - from current position, without px
- scrollBy(pageX, pageY) - absolute values
- scrollIntoView(true / false) - if true element will be aligned with the window top else ...

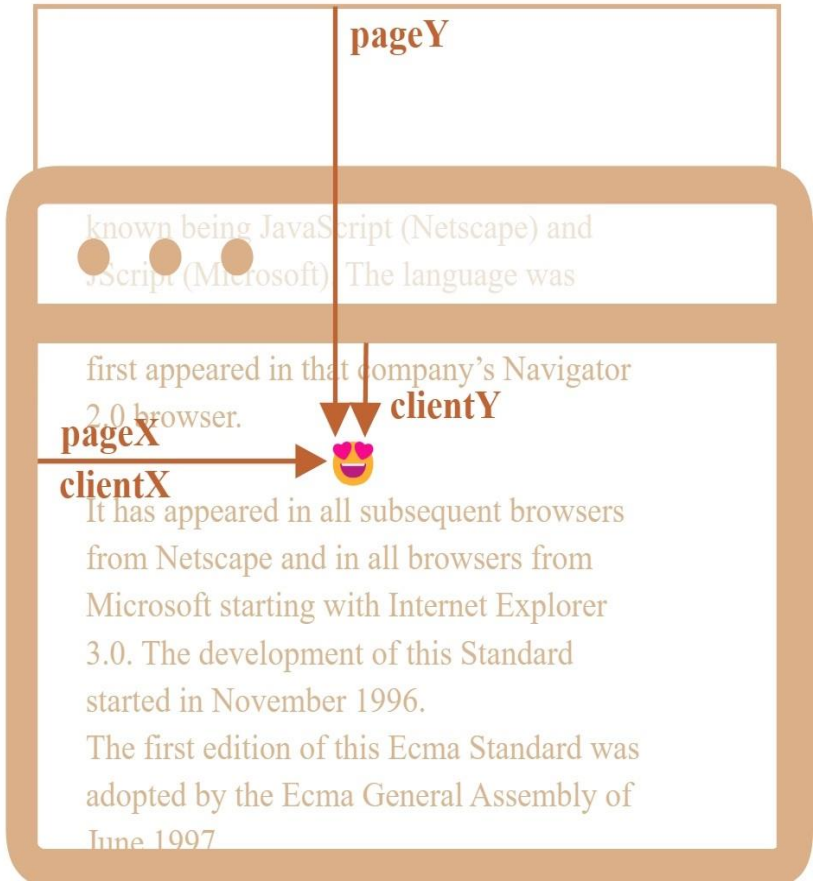
To make the document unscrollable: document.body.style.overflow = "hidden"

COORDINATES

(to move elements around we should be familiar with coordinates)

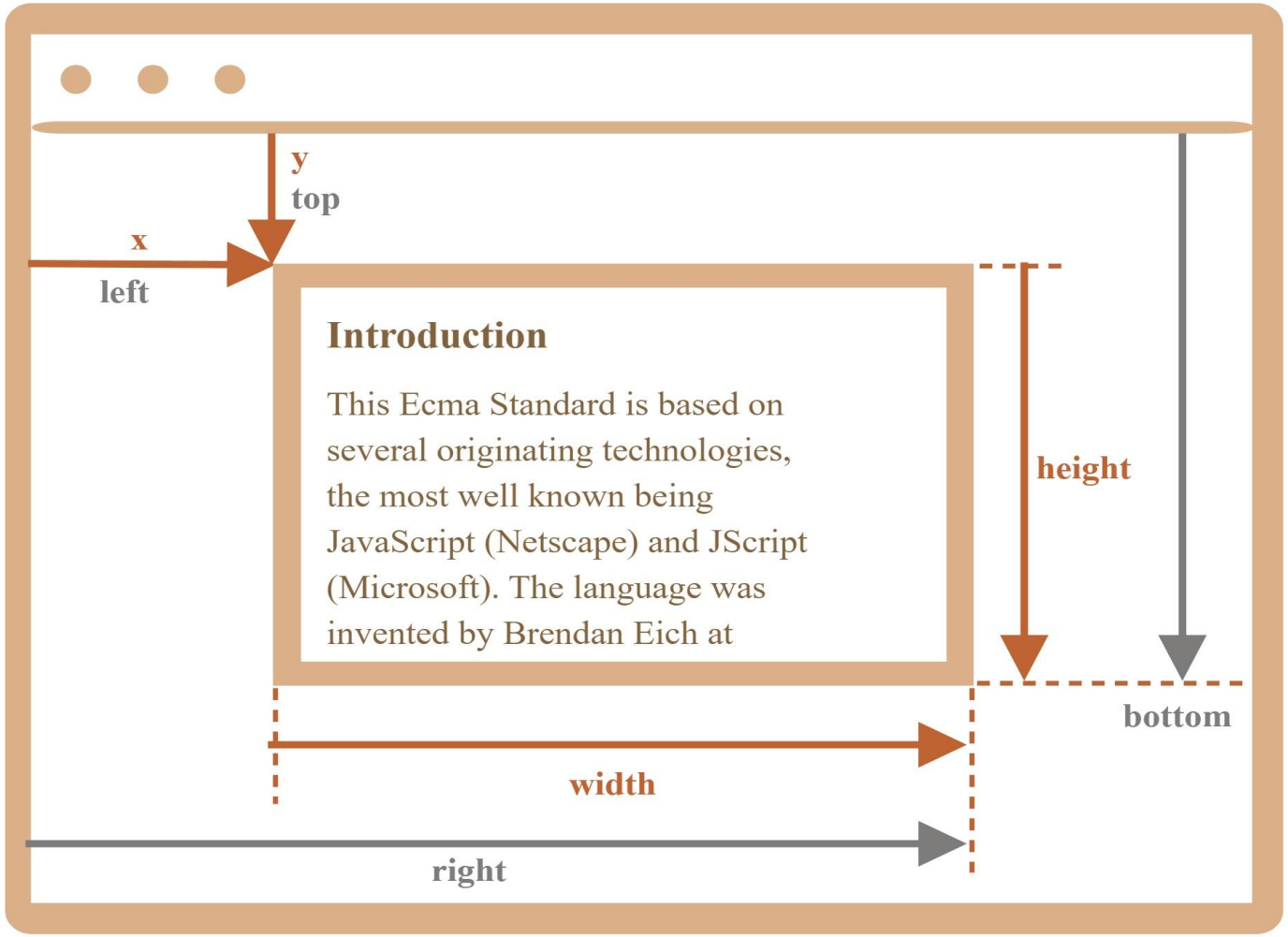
- clientX / clientY - form top and left of the window
- pageX / pageY – from very begining (+ scroll)
- offsetX / offsetY – from parent element
- screenX / screenY - screenLeft / screenTop





ELEMENT coordinates: elem.getBoundingClientRect()

- left = x
- top = y
- right = x + width
- bottom = y + height



Why does top/left exist if there's x/y?

Mathematically, a rectangle is uniquely defined with its starting point (x,y) and the direction vector (width,height). So the additional derived properties are for convenience.

Technically it's possible for width/height to be negative, that allows for "directed" rectangle, e.g. to represent mouse selection with properly marked start and end.

Negative width/height values mean that the rectangle starts at its bottom-right corner and then "grows" left-upwards.