

CSS

CSS FUNDAMENTALS

# Units



**IN A ROCKET**

Learn front-end development at *rocket speed*

**ABSOLUTE  
UNITS**

**RELATIVE  
UNITS**

# ABSOLUTE UNITS

px

pixel

1/96 of 1 inch  
(96px = 1 inch)

pt

point

1/72 of 1 inch  
(72pt = 1 inch)

pc

pica

12pt = 1pc

mm

millimeter

1cm = 10mm

cm

centimeter

10mm = 1cm

in

inch

2.54 cm = 1 inch

Pixels are bad for usability:

- they don't scale,
- they don't help other elements to scale proportionally.

Use them only when you really need an exact and fixed size.

EXAMPLE

```
border: 1px solid #000;
```

**ABSOLUTE  
UNITS**

**RELATIVE  
UNITS**

# RELATIVE UNITS

%

Percentage

Relative to the parent element's value for that property.

em

Em

Relative to the current font-size of the element.

rem

Root em

Relative to the font-size of the root.

ch

Character

Relative to width of the "0".

vw

Viewport width

Relative to the width of viewport.  
 $1vw = 1/100$  of the viewport's width.

vh

Viewport height

Relative to the height of viewport.  
 $1vh = 1/100$  of the viewport's height.

vmin

Smaller dimension

$1vmin = 1/100$  of viewport's smaller dimension.

vmax

Larger dimension

$1vmax = 1/100$  of viewport's larger dimension.

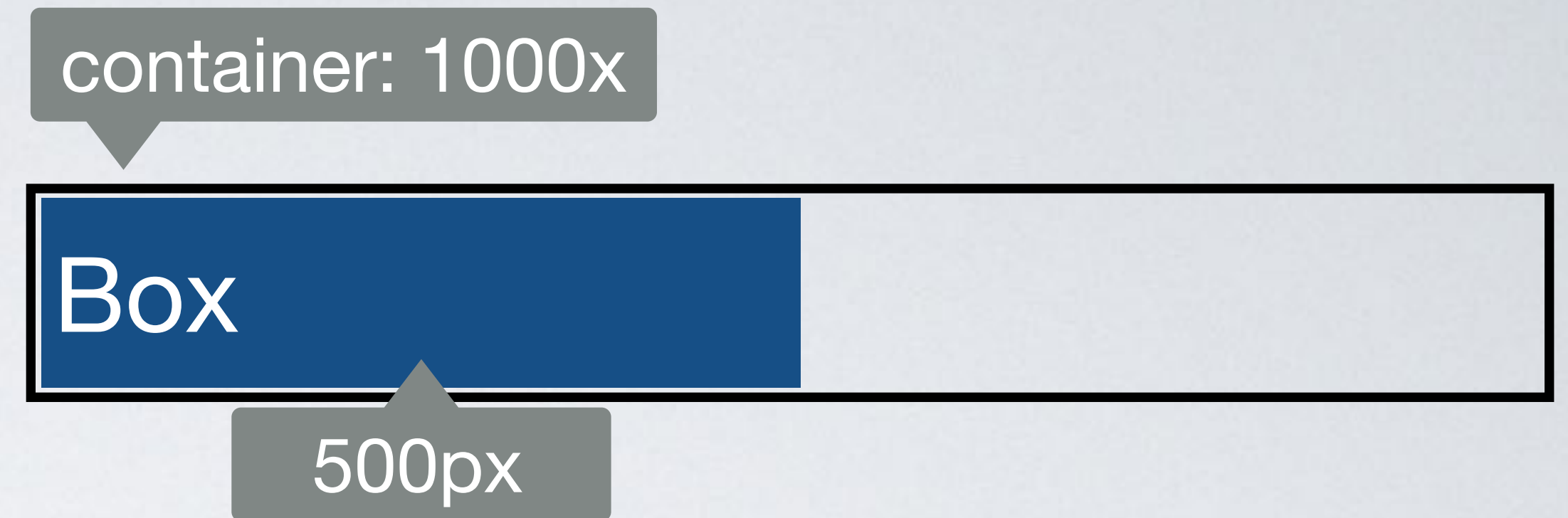
ex

x-height

Relative to the height of a lower-case x.

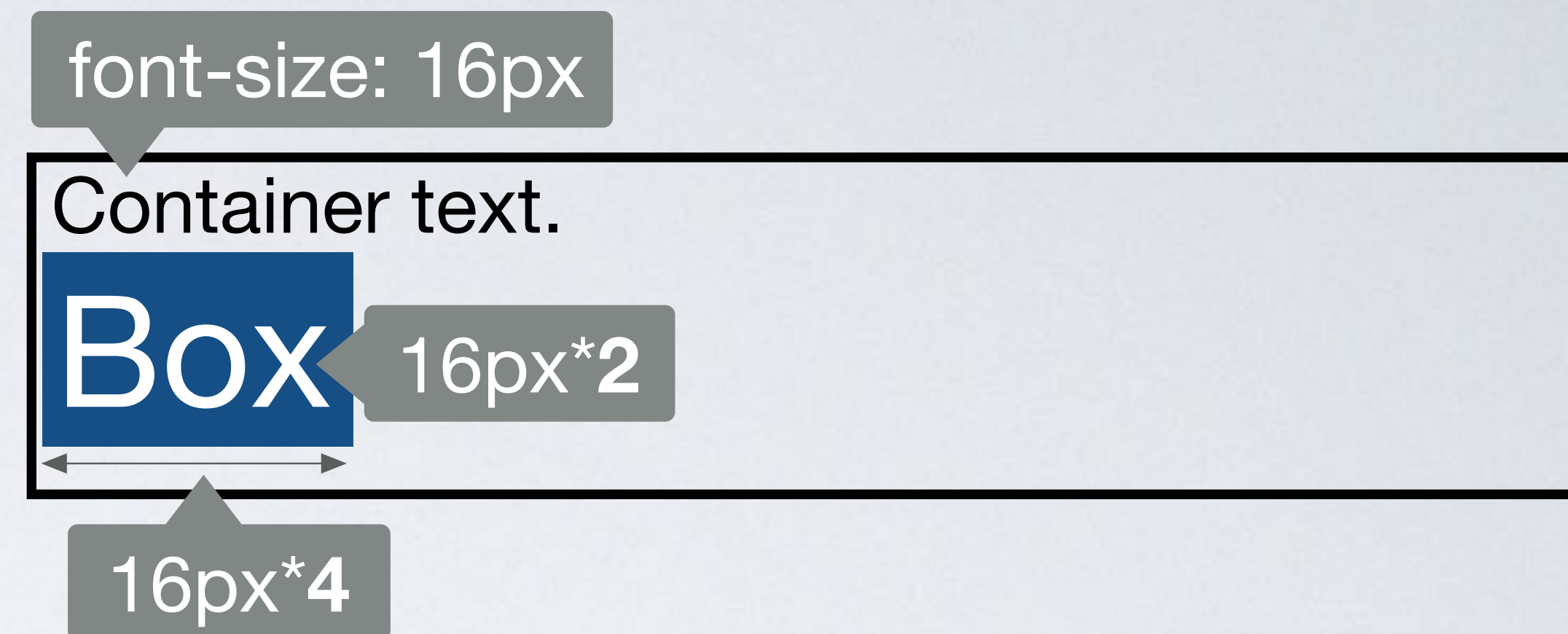
# RELATIVE UNITS / PERCENTAGE

```
.box { width: 50%; }
```



# RELATIVE UNITS / EM

```
.box {  
  font-size: 2em;  
  width: 4em;  
}
```



---

**The em is simply the font size.** In an element with a 16px font, 1em thus means 16px. Expressing sizes, such as margins and paddings, in em means they are related to the font size, and if the user has a big font (e.g., on a big screen) or a small font (e.g., on a handheld device), the sizes will be in proportion.

SOURCE: [CSS units by W3C](#).



# RELATIVE UNITS / REM

```
:root { font-size: 15px; }  
  
.box {  
  font-size: 3rem;  
  width: 6rem;  
}
```



# RELATIVE UNITS / REM vs EM

## EM

```
html {font-size: 100%} 16px

h1 {
  font-size: 2em; 1em(16px)*2 = 32px
  margin-bottom: 1em; 1em(32px)*1 = 32px
}

p {
  font-size: 1em; 1em(16px)*1 = 16px
  margin-bottom: 1em; 1em(16px)*1 = 16px
}
```

## REM

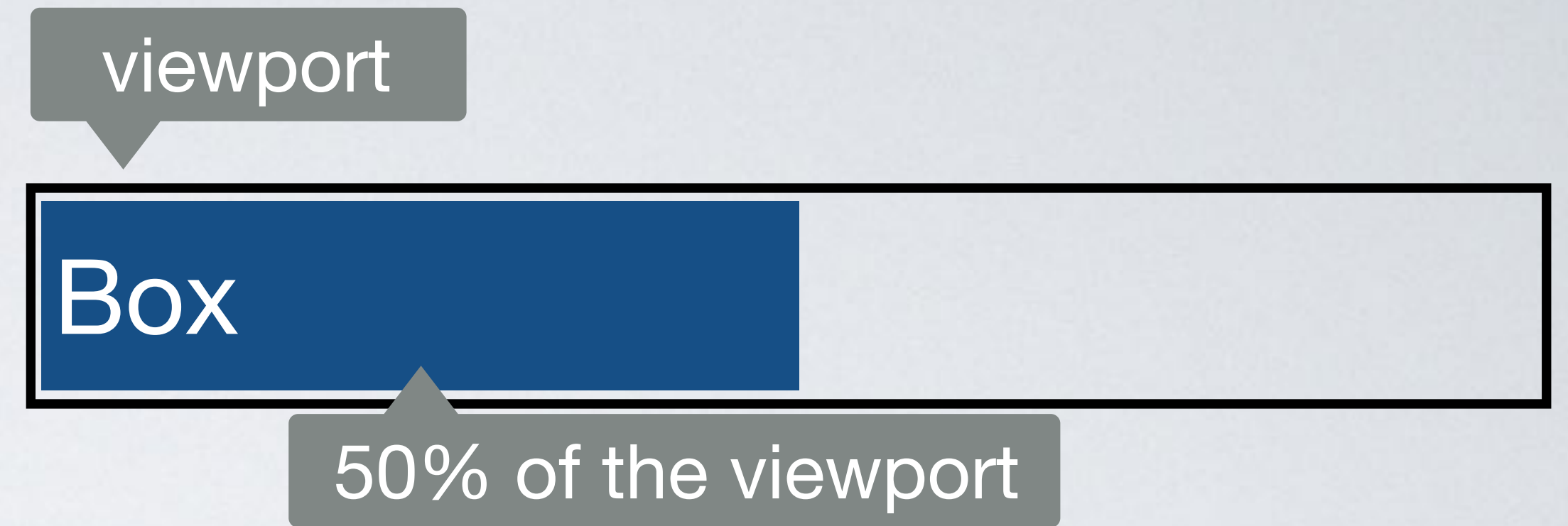
```
html {font-size: 100%} 16px

h1 {
  font-size: 2rem; 1rem(16px)*2 = 32px
  margin-bottom: 1rem; 1rem(16px)*1 = 16px
}

p {
  font-size: 1rem; 1rem(16px)*1 = 16px
  margin-bottom: 1rem; 1rem(16px)*1 = 16px
}
```

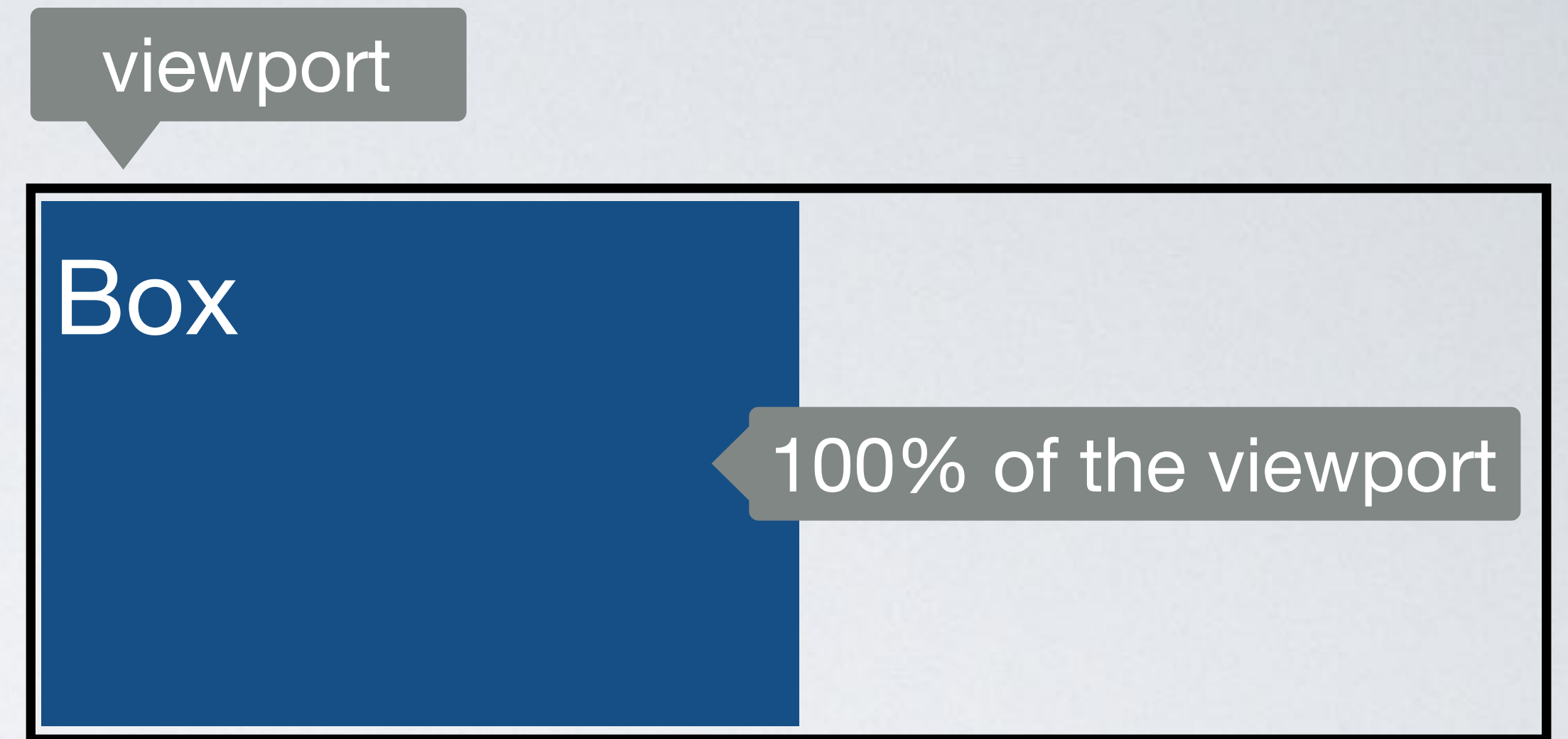
# RELATIVE UNITS / VW

```
.box {  
  width: 50vw;  
}
```



# RELATIVE UNITS / VH

```
.box {  
  height: 100vh;  
}
```



## HOW TO SET HEIGHT FOR FULL PAGE SIZE

PAST

```
html { height: 100%; }  
body { min-height: 100%; }
```

**Problem:** the HTML element does not grow beyond the height of the visible viewport.

NOW

```
body { min-height: 100vh; }
```

**GOOD  
PRACTICE**

# RELATIVE UNITS / VMIN

```
.box {  
  font-size: 10vmin;  
}
```

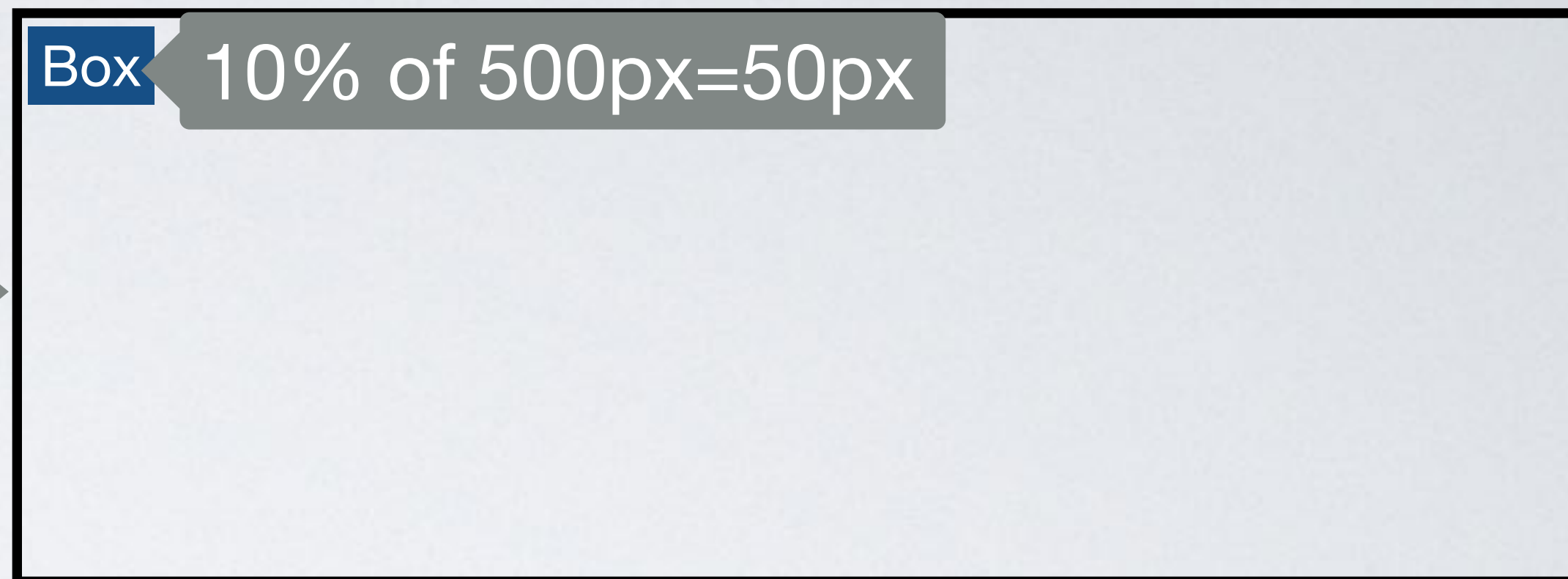
---

1vmin = 1vw or 1vh, whichever is smaller.

height: 500px

width: 1000px

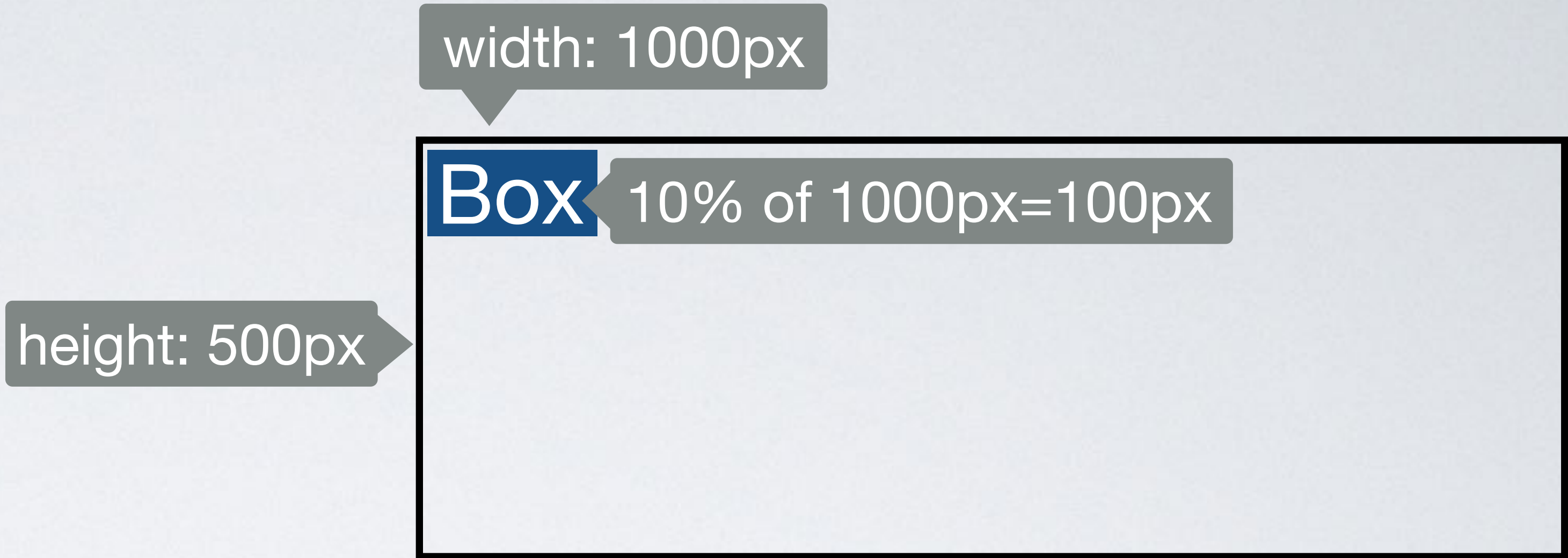
Box 10% of 500px=50px



# RELATIVE UNITS / VMAX

```
.box {  
  font-size: 10vmax;  
}
```

1vmax = 1vw or 1vh, whichever is larger.



Do I need to scale an element when the viewport size changes?



What do I want it to scale relative to?



**ABSOLUTE  
UNITS**



**RELATIVE  
UNITS**



CSS

CSS FUNDAMENTALS

# Units



**IN A ROCKET**

Learn front-end development at *rocket speed*