处理不同的文本方向

mdn web docs

到目前为止，我们在 CSS 学习中遇到的许多属性和值都与屏幕的物理尺寸有关。例如，我们在盒子的顶部、右侧、底部和左侧创建边框。这些物理尺寸非常巧妙地映射到水平查看的内容，默认情况下，网络倾向于支持从左到右的语言（例如英语或法语），而不是从右到左的语言（例如阿拉伯语）。

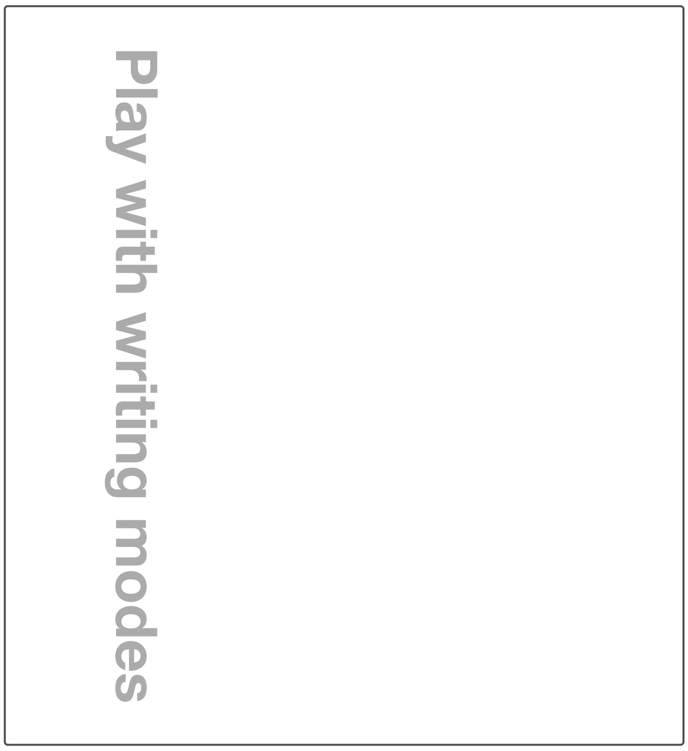
然而近年来，为了更好地支持内容的不同方向性，CSS 不断发展，包括从右到左以及从上到下的内容（例如日语）——这些不同的方向性被称为**书写模式**。随着您学习的进展并开始使用布局，了解书写模式将对您非常有帮助，因此我们现在将介绍它们。

先决条件：	基本的计算机知识、[安装的基本软件、使用文件](#)的基本知识、HTML 基础知识（学习 [HTML 简介](#)）以及 CSS 的工作原理（学习 [CSS 第一步](#)）。
客观的：	了解书写模式对现代 CSS 的重要性。

什么是写作模式?

CSS 中的书写模式是指文本是水平运行还是垂直运行。该 [`writing-mode`](#) 属性使我们可以从一种书写模式切换到另一种书写模式。您不需要使用使用垂直书写模式的语言来执行此操作——您也可以出于创造性目的更改布局部分的书写模式。

在下面的示例中，我们使用 `writing-mode: vertical-rl`. 文本现在垂直运行。垂直文本在图形设计中很常见，可以为您的网页设计添加更有趣的外观和感觉。


Play with writing modes

Interactive editor

```
h1 {
  writing-mode: vertical-rl;
}
```

```
<h1>Play with writing modes</h1>
```

Reset

该属性的三个可能值 [`writing-mode`] 是：

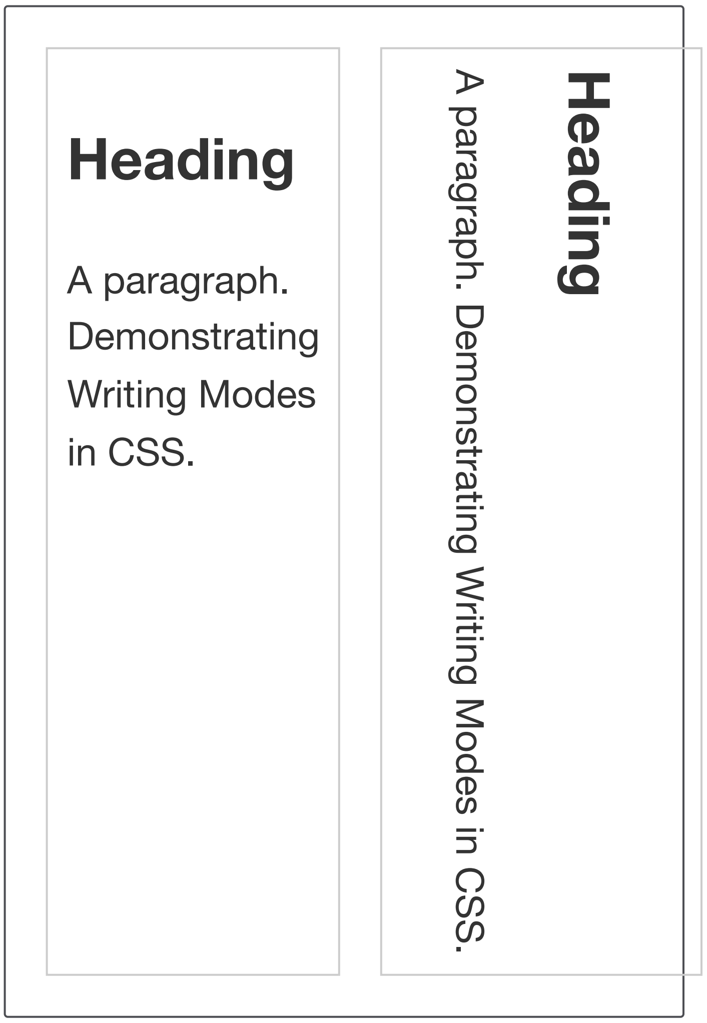
- `horizontal-tb`： 从上到下的块流向。句子水平运行。
- `vertical-rl`： 从右到左的块流向。句子垂直运行。
- `vertical-lr`: Left-to-right block flow direction. Sentences run vertically.

So the `writing-mode` property is in reality setting the direction in which block-level elements are displayed on the page — either from top-to-bottom, right-to-left, or left-to-right. This then dictates the direction text flows in sentences.

Writing modes and block and inline layout

We have already discussed block and inline layout, and the fact that some things display as block elements and others as inline elements. As we have seen described above, block and inline is tied to the writing mode of the document, and not the physical screen. Blocks are only displayed from the top to the bottom of the page if you are using a writing mode that displays text horizontally, such as English.

If we look at an example this will become clearer. In this next example I have two boxes that contain a heading and a paragraph. The first uses `writing-mode: horizontal-tb`, a writing mode that is written horizontally and from the top of the page to the bottom. The second uses `writing-mode: vertical-rl`; this is a writing mode that is written vertically and from right to left.


Heading
A paragraph. Demonstrating Writing Modes in CSS.
Heading
A paragraph. Demonstrating Writing Modes in CSS.

Interactive editor

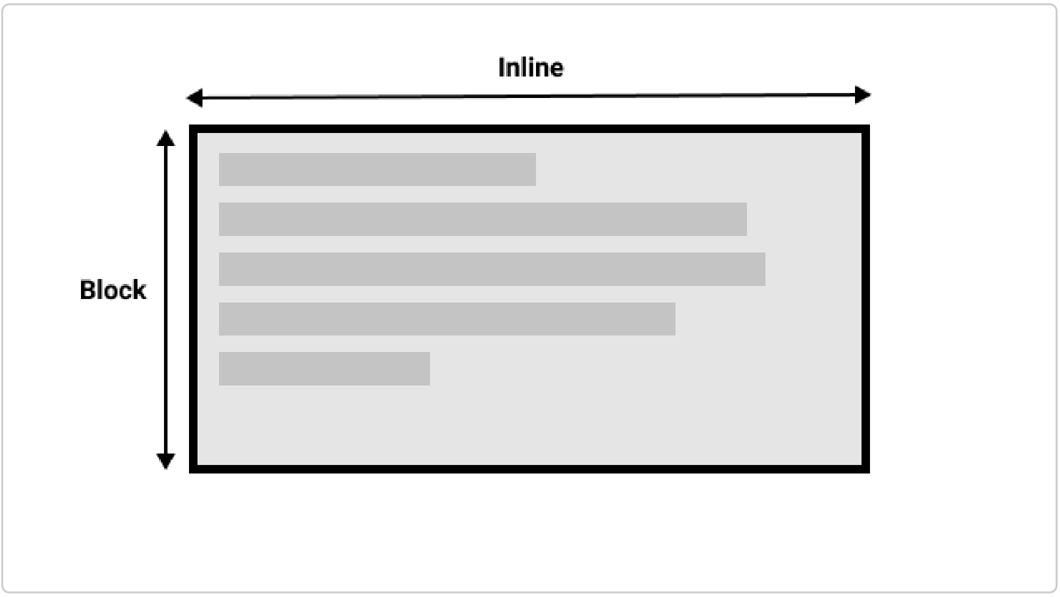
```css
.horizontal {
  writing-mode: horizontal-tb;
}

.vertical {
  writing-mode: vertical-rl;
}
```

```html
<div class="wrapper">
  <div class="box horizontal">
    <h2>Heading</h2>
    <p>A paragraph. Demonstrating
Writing Modes in CSS.</p>
  </div>
  <div class="box vertical">
    <h2>Heading</h2>
    <p>A paragraph. Demonstrating
Writing Modes in CSS.</p>
  </div>
</div>
```

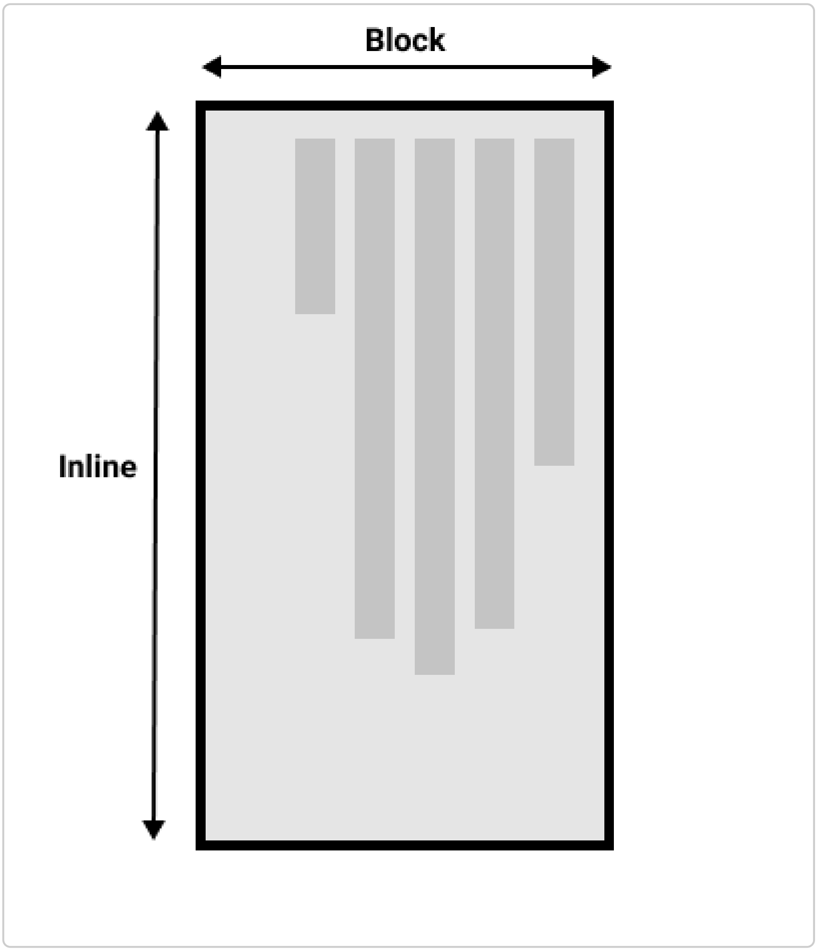
When we switch the writing mode, we are changing which direction is block and which is inline. In a `horizontal-tb` writing mode the block direction runs from top to bottom; in a `vertical-rl` writing mode the block direction runs right-to-left horizontally. So the **block dimension** is always the direction blocks are displayed on the page in the writing mode in use. The **inline dimension** is always the direction a sentence flows.

Reset

This figure shows the two dimensions when in a horizontal writing mode.



This figure shows the two dimensions in a vertical writing mode.



Once you start to look at CSS layout, and in particular the newer layout methods, this idea of block and inline becomes very important. We will revisit it later on.

Direction

In addition to writing mode we also have text direction. As mentioned above, some languages such as Arabic are written horizontally, but right-to-left. This is not something you are likely to use in a creative sense — if you want to line something up on the right there are other ways to do so — however it is important to understand this as part of the nature of CSS. The web is not just for languages that are displayed left-to-right!

Due to the fact that writing mode and direction of text can change, newer CSS layout methods do not refer to left and right, and top and bottom. Instead they will talk about *start* and *end* along with this idea of inline and block. Don't worry too much about that right now, but keep these ideas in mind as you start to look at layout; you will find it really helpful in your understanding of CSS.

Logical properties and values

The reason to talk about writing modes and direction at this point in your learning is that we have already looked at a lot of properties that are tied to the physical dimensions of the screen, and these make more sense when in a horizontal writing mode.

Let's take a look at our two boxes again — one with a `horizontal-tb` writing mode and one with `vertical-rl`. I have given both of these boxes a `width`. You can see that when the box is in the vertical writing mode, it still has a width, and this is causing the text to overflow.

What we really want in this scenario is to essentially swap height with width in accordance to the writing mode. When we're in a vertical writing mode we want the box to expand in the block dimension just like it does in the horizontal mode.

To make this easier, CSS has recently developed a set of mapped properties. These essentially replace physical properties — things like `width` and `height` — with **logical**, or **flow relative** versions.

The property mapped to `width` when in a horizontal writing mode is called `inline-size` — it refers to the size in the inline dimension. The property for `height` is named `block-size` and is the size in the block dimension. You can see how this works in the example below where we have replaced `width` with `inline-size`.

Logical margin, border, and padding properties

In the last two lessons we have learned about the CSS box model, and CSS borders. In the margin, border, and padding properties you will find many instances of physical properties, for example `margin-top` , `padding-left` , and `border-bottom` . In the same way that we have mappings for width and height there are mappings for these properties.

The `margin-top` property is mapped to `margin-block-start` — this will always refer to the margin at the start of the block dimension.

The `padding-left` property maps to `padding-inline-start` , the padding that is applied to the start of the inline direction. This will be where sentences start in that writing mode. The `border-bottom` property maps to `border-block-end` , which is the border at the end of the block dimension.

You can see a comparison between physical and logical properties below.

If you change the writing mode of the boxes by switching the `writing-mode` property on `.box` to `vertical-rl` , you will see how the physical properties stay tied to their physical direction, whereas the logical properties switch with the writing mode.

You can also see that the h2 has a black `border-bottom` . Can you work out how to make that bottom border always go below the text in both writing modes?

There are a huge number of properties when you consider all of the individual border longhands, and you can see all of the mapped properties on the MDN page for Logical Properties and Values.

Logical values

We have so far looked at logical property names. There are also some properties that take physical values of `top`, `right`, `bottom`, and `left`. These values also have mappings, to logical values — `block-start`, `inline-end`, `block-end`, and `inline-start`.

For example, you can float an image left to cause text to wrap round the image. You could replace `left` with `inline-start` as shown in the example below.

Change the writing mode on this example to `vertical-rl` to see what happens to the image. Change `inline-start` to `inline-end` to change the float.

Here we are also using logical margin values to ensure the margin is in the correct place no matter what the writing mode is.

> **Note:** Currently, only Firefox supports flow relative values for `float`. If you are using **Google Chrome** or **Microsoft Edge**, you may find that the image did not float.

Should you use physical or logical properties?

The logical properties and values are newer than their physical equivalents, and therefore have only recently been implemented in browsers. You can check any property page on MDN to see how far back the browser support goes. If you are not using multiple writing modes, then for now you might prefer to use the physical versions. However, ultimately we expect that people will transition to the logical versions for most things, as they make a lot of sense once you also start dealing with layout methods such as flexbox and grid.

Test your skills!

You've reached the end of this article, but can you remember the most important information? You can find some further tests to verify that you've retained this information before you move on — see Test your skills: Writing modes and logical properties.

Summary

The concepts explained in this lesson are becoming increasingly important in CSS. An understanding of the block and inline direction — and how text flow changes with a change in writing mode — will be very useful going forward. It will help you in understanding CSS even if you never use a writing mode other than a horizontal one.

In the next article, we'll take a good look at overflow in CSS.