Working with text beyond English

Most of the time we don't have to worry about alignment and text direction when working with web content. When the language works the same as English we have no problem. We start having issue becomes when we work with languages other than English in the same document. It gets more complicated when our primary language uses a different writing mode than the one we're used to. Think about vertical Kanji or right-to-left Hebrew or Arabic text mixed with your traditional English or latin content.

We'll discuss tools that CSS gives you to control how languages other than English appear on web pages. We'll briefly talk about some CSS terminology that makes our CSS DRY by making multiple rules unnecessary.

Why is this important? People

Towards the end of his presentation at Fronteers 2016 Bruce Lawson states that *lack of awareness and locally relevant content* is the most common barrier to Internet adoption in Asia.

Later in Bruce's presentation states that **50% of websites are in English, a language spoken by 10% of speakers in the survey countries. By way of contrast, only 2% of websites worldwide are in Mandarin and less than 0.1% in Hindi.** (Data verified with information from: [w3techs](#))

While English is the predominant language of the web it is not a universal language, particularly in emerging markets and rural areas of developed nations. There is also an assumption that the rules of western communication, possession of high end hardware and even cultural norms will be equally applicable across the world.

That is not true.

> WWW is the World Wide Web not Wealthy Western Web
>
> — Bruce Lawson

What would the web look in their language? What changes would we have to make if we want our content to be understood on countries where people don't read English? Again, where are your customers coming from? is it where we expect them from?

Why is this important? Development

Most of us do our work in English but that may not always be the case. We might be called to create content in languages other than English or to mix content in English and other languages, some of which may be laid out differently than your main content.

We can use writing modes to play with the way our standard content is laid out on the page. Along with transitions we can rotate our text and make it look

different than the standard left to right, top to bottom.

Look at this pen from Jen Simmons; pay special attention to the way she styled the h1 element and how she changed direction without actually having to use transitions or animations.

See the Pen Writing Mode Demo — Headline by Jen Simmons (@jensimmons) on CodePen.

The code that changes the direction of the h1 element is this:

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

The full list of values for `writing-mode` are:

```css
h1 {
  writing-mode: horizontal-tb;
}

h1 {
  writing-mode: vertical-rl;
}

h1 {
  writing-mode: vertical-lr;
}

h1 {
  writing-mode: sideways-rl;
}

h1 {
  writing-mode: sideways-lr;
}
```

Understanding writing modes may help you better understand Grid and Flexbox.

Did you ever wonder why Flexbox attributes use start and end in their names? By doing so the layout will work the same regardless of where the start and end of our content are.

Now that we know what we're working we need to define a few more concepts before we can look at different writing systems and how we can use them on the web.

Figure 1: Three concepts to understand about CSS Writing Modes

- **block direction**
- **inline direction**
- **character direction**

Block direction

Block direction describes how blocks of text are laid out, whether it's horizontally or vertically. Most languages wil use a vertical block direction but not all of them.

Inline direction

Inline direction controls the way the text flows, left to right or right to left... yes, even in vertical layouts. The `dir` global HTML attribute controls the inline direction of the content. That's why you see (or should see) `html` tags like this for United States English:

```
<html lang='en-us' dir='ltr'>
```

And like this for Chilean Spanish:

```
<html lang='es-cl' dir='ltr'>
```

Character direction

While studying this, and reading lots of articles by Jen Simmons, I found Character direction, the most confusing. It describes where the top of a character is.

For example: if you write a capital Y, where is the top of the character? Most languages will use top to bottom but not all.

Different writing systems

There are four main writing systems. We'll discuss them along with any changes we need to make to our CSS to acommodate for their differences.

Latin based systems

Most languages in the world will use a Latin based writing system with top to bottom and left to right. This is also the default for CSS so we don't need to set it as the default.

When working with multiple languages in the same document we can reset it with something like this"

```
div .english {
    writing-mode: horizontal-tb;
}
```

Arabic-based systems

Arabic-based systems (including Arabic and Henrew) that use a top to bottom, right to left system (also known as RTL).

technically Arabic systems have a top to bottom blockdirection and a right to let inline direction.

Take the following example website for the united nations in Eglish and Arabic and see how the text is laid out on the Arabic site (figure 2), both the text and the layout read from right to left and look like a mirror of the English site.

Figure 2: United Nations site in English

To use Arabic as the default language for a page, use the following html element:

```
<html lang='ar' dir='rtl'>
```

to control the direction of specific elements within the pge you can use the `direction` CSS attribute, like this:

```
.arabic-content {
  direction: rtl;  /* Right to Left */
}
```

Han based systems

Han based systems include, among others, the CKJ languages (Chinese, Korean and Japanese). The interesting part is that these languages can be written with vertical or horizontal display blocks.

   If you write your Japanese in a top to bottom, left to right mode then you don't have to make changes. The default writing mode will work.

   If you write your content in a vertical layout you can use the following CSS:

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

And let the browser handle the content layout for you.

   You can mix mix vertical and horizonal layouts in the same page like in this page from Vogue Japan:

Figure 4:
Vogue
Japan
page
using
vertical
and
horizontal
layouts

You can set the writing mode for the vertical portion of the article and let the rest use the default horizontal, left to right layout.

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

Or you can change the default to a vertical orientation, and then set specific elements to horizontal-tb, like this:

```css
html {
    writing-mode: vertical-rl;
}

h2, .photocaptions, section {
    writing-mode: horizontal-tb;
}
```

If you mix western and CJK languages. Look for the English words in the Japanese text below:

Figure
5:
Example
of
vertical
Japanse
and
English
text

Mongolian writing systems

OK, this was beyond what I expected but not too complex not to use it. There is one more writing system to add to the mix. We'll probably never use it… it's Mongolian.

Text in Mongolian runs vertically down the page just like Han-based systems. There differences are two:

- Block-level elements stack from left to right
- The character direction is "upside down". The top of the Mongolian characters point to the right

Creative uses

We can use vertical writing directions to create headings like the Octavia Buttler example we discussed earlier. You can use it anyway you want to.