**BROWSER SUPPORT**

When new browsers are developed (or an existing browser is updated), they often include new features that weren't previously available. The result, however, is a multitude of browsers with varying functionalities.

To avoid inconsistencies across browsers when creating a website, developers must ensure that newer HTML or CSS features are supported in each browser. The following resource is an up-to-date record of CSS properties across many versions of many browsers:

* "[Can I use](http://caniuse.com/)"

For example, if you search "Can I use" for the CSS property filter (a property that can add a visual effect to an image), you'll see that it is supported in most browsers, except for Internet Explorer 11 and Opera Mini.

When a feature is searched using the resource above, the search results include a *support matrix*. The support matrix outlines which browsers (and browser versions) support the feature. The bottom of the pane also includes information about known issues and bugs.

**Global Support vs Unprefixed Support**

In the top-right corner of "Can I use", there is information on "global support" and "unprefixed support". What do these terms mean and what is the difference between them?

As stated earlier, every browser has its own implementation of many newer CSS rules. To distinguish their own implementation, browsers add a prefix to the CSS property. The prefix is known as a *vendor prefix*. For example, the -mozvendor prefix refers to Mozilla Firefox's implementation. A full list of vendor prefixes can be found [here](https://developer.mozilla.org/en-US/docs/Glossary/Vendor_Prefix).

The following code demonstrates how the transition property is implemented across all browsers using vendor prefixes.



The "global support" value mentioned earlier represents the percentage of supported browsers for the specified feature if all necessary prefixes are used (as is the case in the previous example). The "unprefixed support" value represents the percentage of supported browsers for the feature if no prefixes are used (if just the transition declaration were written in the previous example).

To identify exactly which CSS properties need vendor prefixes, you can use tools like [this one](http://pleeease.io/play/) to generate all of the necessary vendor prefixes for you.

**POLYFILLS**

Even with vendor prefixes, what if a user is using a browser that doesn’t support newer browser functionalities? Developers have created libraries called *polyfills* to support such users.

Polyfills:

1. Detect the user's browser.
2. Collect information about which features are supported by the browser.
3. Return the collected information to your website.

The collected information allows you to write alternative CSS for browsers that are missing certain features. Your website may not look as visually appealing as it would on a newer browser, but it will function.

One example of a polyfill is Modernizr. To use Modernizr:

1. [Navigate to the website](https://modernizr.com/) and click "Download."
2. Click the + next to any features you want to polyfill.
3. Click "Build."
4. Click "Download" next to the "Build" option in the resulting pop-up. This will prompt you to download a JavaScript file (the polyfill code).
5. Place the downloaded **.js** file (JavaScript file) into the corresponding folder in your website's directory.
6. Link the JavaScript file using a <source> tag in your **index.html** file.
7. Use CSS to target elements that have the detected feature using .feature-name. To target elements that don't have the detected feature use .no-feature-name. The code feature-name is intended to represent the actual CSS feature.

An example of alternative CSS rules from the Modernizr documentation:

