Hard Reset: Using the CSS Initial Value

There are times when the CSS cascade becomes a pain in the ass. When we want to revert the value of a property to its default (before we added any other value for the property in the current element or any ancestor up the chain) we have to remember the property but also what the default value is.

CSS provides several mechanisms to handle inheritance. We'll discuss several ways to work with the cascade to make it do what we need.

initial

The newest (to me) way to reset is the initial value.

The `initial` CSS keyword sets and element to its initial value. It is allowed on every CSS property and causes the element for which it is specified to use the initial value of the property.

> On inherited properties, the initial value may be surprising and you should consider using the inherit, unset, or revert keywords instead.
>
> Test the hell out of this keyword.

```
<p style="color:red">
    this text is red
       <em style="color:initial">
          this text is in the initial color (e.g. black)
       </em>
    this is red again
 </p>
```

One thing to be aware is that the initial value may not be the same across browsers. This is specially important when working with fonts as the default font is not necessarily the same in all browsers.

inherit

The `inherit` CSS value uses the value of the parent element's property (defining the parent as the parent element in the document tree, even if it's not the containing block). It is allowed on every CSS property.

In the following CSS fragment the h2 elements that are children of #sidebar will inherit the color of its parent. The stylesheet defines another container (`div#current`) with a different color.

```
/* make second-level headers green */
 h2 { color: green; }

 /* leave those in the sidebar alone so they use the parent's color */
 #sidebar h2 { color: inherit; }

div#current { color: blue; }
```

In the following HTML fragment the sidebar's h2 element will be blue, the color of the parent.

```
<div id="current">
  <sidebar id="#sidebar">
    <h2>Sidebar Title</h2>
</sidebar>
</div>
```

For inherited properties, this reinforces the default behavior, and is only needed to override another rule. For non-inherited properties, the results may be unexpected and you may find that using `initial`, or `unset` on the `all` property works better.

unset

The unset CSS keyword is the combination of the initial and inherit keywords. This keyword resets the property to its inherited value if it inherits from its parent or to its initial value if not. In other words, it behaves like the inherit keyword in the first case and like the initial keyword in the second case.

Color is an inherited property so the following CSS:

```
.foo {
  color: blue;
}
.bar {
  color: green;
}

p {
  color: red;
}
.bar p {
  color: unset;
}
```

Produces the following colors in the HTML file below.

```
<p>This text is red</p>
<div class="foo">
  <p>This text is also red</p>
</div>
<div class="bar">
  <p>This text is green (default inherited value)</p>
</div>
```

revert

This property is only supported in Safari.

a

The revert keyword resets the cascade as if no stylesheets of the specified type were present. For example, if the revert keyword is used in an author stylesheet (the one we write as designers) then it'll assume that no stylesheets are present

and will take the value from the default user agent style sheet or any user stylesheet present.

The order of how the cascade works is show below in order of importance from left to right:

```
user agent > user > author
```

There's no perfect way to handle cascade idiosyncrasies reliably across browsers. The CSS Cascading and Inheritance Level 4 is an editor's draft and, my guess, is that there will be significant work in the spec before it's finalized.

For now these are the tools we have so let's make the best use of them.