---

Title: INTRO TO HTML5 TEMPLATES

Date: 2015-5-20

Author: Tyler

---

HTML5 template is NOT a new data binding tool and its not a new MV\* framework; it’s part of the html spec. Since it’s in the spec you may be curious about the process of getting into the w3 spec, and what exactly that means.

1. Developers gripe about html/css missing a feature, W3 doesn’t listen. If the w3 listened to every complaint we would have millions of weird features.
2. People build libraries to solve the problem. The W3 starts to watch but takes no action. If the problem is legitimate then more and more libraries will be created to solve the problem.
3. One library or framework rises to the top, W3 learns from it, and creates a new spec.
4. Once in the w3 spec all browsers should support this feature, and it should function the same across all browsers.

Templating is old hat for most programming languages- consider for example razor syntax in c# or erb in ruby. If you are living in the webs front-end you are probably familiar with mustach.js or one of its less popular friends like hogan.js, handlesbars.js, Walrus, DoT.js, or Dust.js. If you are familiar with any JS front end framework you probably understand how templating works.

The general idea behind templating is that you have something that you would like implemented multiple times with slight variants. A template lets you write it once, and switch the variants every time. For example, If you have a list of projects you worked on in your portfolio and the only thing that changes per item is the picture, title, and description. You COULD copy and paste 1 working block of code for each item, or you could use a template.

If you have avoided using any of the current js templates, and are curious how they work, they give the ability to build small little views that you can reuse. The advantage they have over copy and pasting code is that if a change is required to each set, AFTER you copy and paste, you need to do some text editor kung fu to not end up rewriting everything.

Lets jump into some code!!! The following is an example of what a simple template that displays a persons picture, their name, and a card that holds all the styling for that card.

```

<template id='profileCardTemplate'>

<div class='profileCard'>

<img src=”” alt='Profile Image'>

<h4 class='personName'></div>

</div>

</template>

```

The first thing you may notice is the new <template> tag is now syntactically accurate. Templates can be placed anywhere in , or , and will not render until called upon by javascript. This also means that whatever is in this section won’t load/run until its called upon.

In order to instantiate a template you need to activate it from javascript. The easiest way to do so is by replicating the code and appending it to your dom

```

var profileCard = document.querySelector('#profileCardTemplate');

profileCard.content.querySelector('img').src = 'profilePicture.jpg';

var clone = document.importNode(profileCard.content, true);

document.body.appendChild(clone);

```

There are a ton of great features in templates, but there are two I would like to call out:

1. It encourages lazy loading of not just heavy assets like a list of user profile pictures, but also the dom elements associated to said list. The way it does this is by not adding the item to the DOM until after its instantiated. So if you javascript doesn’t run until the page is loaded, you have easily built a lazy loading.
2. Another thing to consider with html5 templates is that script tags nested inside these templates will not run until are instantiated. In the following example the console.log will only occur when the link is clicked

```

<a onclick="generateTemplate()">DO IT!!!</a>

<script>

var generateTemplate = function(){

var content= document.querySelector('template').content;

var clone = document.importNode(content, true);

document.body.appendChild(clone);

}

</script>

<template>

<script> console.log("woot woot");</script>

</template>

```

With any new addition to the front end web designers tool belt, one needs to be wary of the browser support. As you can see from Can I Use its pretty great for our lovely friend IE.

If your project needs to support, I.E. there are shims and polyfils available, so coupled with modernizr you should be able to build a production app with html5 templates.

As the HTML5 Spec continues to grow there are tons of cool features that get added every release. HTML5 templates are one of my favorite I have seen in a while, and I encourage everyone tool look into them. If possible, make your page a little lighter and pull out a javascript library!