Basic Web Components and TS info

Custom elements

- The name of your custom element must contain a **dash (-)**. For example, `<file-reader>`, and `<skype-login>` are valid names for custom elements, while `<skype_login>`, and `<skypelogin>` are not. This is necessary in order to allow the HTML parser differentiate between a custom element and an inbuilt HTML element.
- A custom element can't be registered more than once. A `DOMException` error will be thrown if you do so.
- A custom element can't be self-closing. For example, you can't write a custom element like this: `<skype-login />`. It should always be written like this: `<skype-login></skype-login>`.

Template

- allows you to reuse pieces of code

LifeCycleHooks

- **constructor():** Here, you can attach event listeners and initialize state.
- **connectedCallback():** Called whenever the custom element is inserted into the DOM.
- **disconnectedCallback():** Called whenever the custom element is removed from the DOM.

- **attributeChangedCallback(attrName, oldVal, newVal):** Called whenever an attribute is added, removed or updated. Only attributes listed in the **observedAttributes**property are affected.
- **adoptedCallback():** Called whenever the custom element has been moved into a new document.

https://dev.to/bennypowers/lets-build-web-components-part-1-the-standards-3e85

Overall this was a good article and hit all the key pieces for building a web component with examples. we should be able to use this as a reference as we build

https://webdesign.tutsplus.com/articles/how-to-create-your-own-html-elements-with-web-components--cms-21524

- found a lead on how to add in styling from CSS
  - with the exception of style hooks you might specifically create in order to allow external CSS targeting

- good information on browser support because of how old the article is

https://auth0.com/blog/web-components-how-to-craft-your-own-custom-components/

- so far the best tutorial I've seen

https://dev.to/thepassle/web-components-from-zero-to-hero-4n4m

- This tutorial is also really well done

https://github.com/mateusortiz/webcomponents-the-right-way

- pretty much all the resources on web-components gathered into one pace

https://blog.bitsrc.io/web-components-without-a-framework-lets-build-a-modal-a450a4536340

- didn't think it was that great

https://www.webcomponents.org/introduction

- not that great either but provides a quick overview

TypeScript Resources

https://medium.freecodecamp.org/learn-typescript-in-5-minutes-13eda868daeb

https://philandrews.io/post/how-to-get-your-team-to-buy-into-typescript

https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes.html

TypeScript Plan

1. Merge tsc into the build tool chain.
2. Tell your team they are now using typescript.
3. Fire any dissenters on the spot as a warning to the others. Lavish the loyal ones with praise.
4. Inform management that it was in fact your team's decision to experiment with typescript.
5. Resume normal development activities.
6. Crush your enemies.