Web Component estudos

Passos:

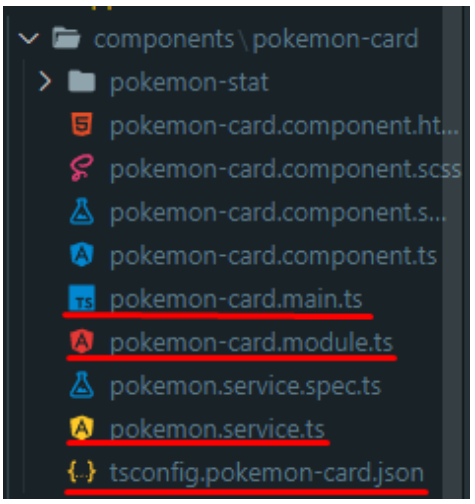
Fazer virar um web component e depois armazenar ele em algum lugar (os arquivos Dist)

Se estivermos no **tsconfig.json** com o target menor que ES2015 temos que atualizar, pois com versões menores que a ES2015 não tem suporte a Web Components

Internet Explorer não tem suporte para web components.

Angular elements para criar o web component. npm i @angular/elements
jscat para juntar os arquivos de build em apenas 1. npm i -D jscat (Essa dependência junta todos os arquivos JavaScript do build que fica em dist em apenas 1 arquivo).
É bom colocarmos tudo do component no mesmo diretório inclusive o service do mesmo. O único problema é que devemos pensar de forma diferente quando estamos criando web component, pois não queremos deixar nada muito encapsulado em questão de eventos porque queremos comunicar para cima, como isso vai ser disponibilizado para outras aplicações não importa o que vamos fazer dentro do nosso component o que importa é o poder que damos para o component de cima (components pai, para a aplicação que está usando esse component), então se temos um input que a gente lida com ele dentro do nosso component, chama o serviço, salva no local storage o legal é não fazer isso o legal é as vezes da oinput que vai tratar um dado de alguma coisa e já soltar um evento com esse novo valor para que assim deixemos a aplicação de quem está usando nosso web component que lide com esse dado, então a gente tem uma certa mudança na forma de pensar em como a gente está construindo a aplicação com Angular se a gente tiver algum web component, pois temos que conseguir que ele faça sentido na nossa aplicação e que possamos disponibilizar ele para que possa fazer sentindo para outras pessoas usarem em suas aplicações.

Para tudo dar certo cada component tem que ter um module.ts, main.ts, tsconfig.json, service.ts, pois cada projeto precisa disso e os web components são tratados como projetos dentro do nosso projeto. E também temos de configurar o angular.json com as configurações extras do web component do projeto e criar um script de build somente do nosso web component.



Precisamos importar o modulo do web component no app.module.ts, pois se não quando dermos um ng serve ele não vai funcionar porque esse componente não está sendo passado no nosso componente raiz.

```
@NgModule({
  declarations: [AppComponent],
  imports: [BrowserModule, HttpClientModule, PokemonCardModule],
  providers: [],
  bootstrap: [AppComponent],
})
export class AppModule {}
```

Quando damos build no nosso projeto por padrão ele cria uma pasta dist e gera os arquivos de build lá, podemos alterar isso em angular.json na options do nosso web component para os arquivos de build irem para outro destino, e também temos que notar que no angular.json temos que criar um novo project, pois novamente frisando o web component é interpretado como um projeto dentro do nosso projeto.

```
"pokemon-card": {
  "projectType": "application",
  "root": "",
  "sourceRoot": "src",
  "architect": {
    "build": {
      "builder": "@angular-devkit/build-angular:browser",
      "options": {
        "outputPath": "dist/pokemon-card",
```

aot: true faz com que tudo sege buildado antes da aplicação rodar e não enquanto a aplicação roda.

```
"aot": true,
```

commonChunk: false é para ele não ter um arquivo de common com algumas coisas da aplicação separadas nesse arquivo.
assets serve para nós pegarmos algum arquivo necessários no nosso projeto como: logo, imagem, icone, etc. temos que passar o arquivo nesse assets.

```
"commonChunk": false,
"assets": [],
```

Se possível sempre deixar os styles no nosso component para não precisarmos pegar de fora, pois isso gera conflitos que teríamos que renomear os styles e tudo ia ser jogado no build para apenas um arquivo e pode ter conflito se o nome for igual.

Vamos ter também que criar um script para que faça o build para nós e também junte nossos arquivos de build em apenas 1.

Para isso vamos no package.json e em script adicionamos uma propriedade de build com o nome e ng build [nome do nosso web component] e usamos o jscat para juntar todos os arquivos em um arquivo só, então ele basicamente pega e da um join nesses arquivos para nós e passamos o diretório dos arquivos e os nomes deles e passamos para qual arquivo que vão se tornar.

```
"build:pokemon-card": "ng build pokemon-card  &&  jscat ./dist/pokemon-card/runtime.js ./dist/pokemon-card/polyfills.js ./dist/pokemon-card/main.js > dist/pokemon-card/pokemon-card.js"
```

Agora pra dar build no nosso web component é só executar o comando: **npm run build:pokemon-card**.
--prod não é mais utilizado nas versões 13 em diante do Angular, pois o build por padrão já é o build de prod.