Build et déploiement d’une app Angular

Angular propose la commande ng build pour compiler l’application pour la production. La sortie du build est par défaut dans le dossier “dist/” à la racine.

- **La compilation** génère les fichiers nécessaires pour le navigateur web : index.html, ainsi que les fichiers *.js et *.css

- **La librairie Webpack** compile notre application (la configuration est dans angular.json)

NB : **ng build** utilise par défaut la configuration de production. Mais vous pouvez utiliser la configuration de développement via l’option ng build `--configuration development`

1 Exemple avec ng build

Webpack va compiler notre application :

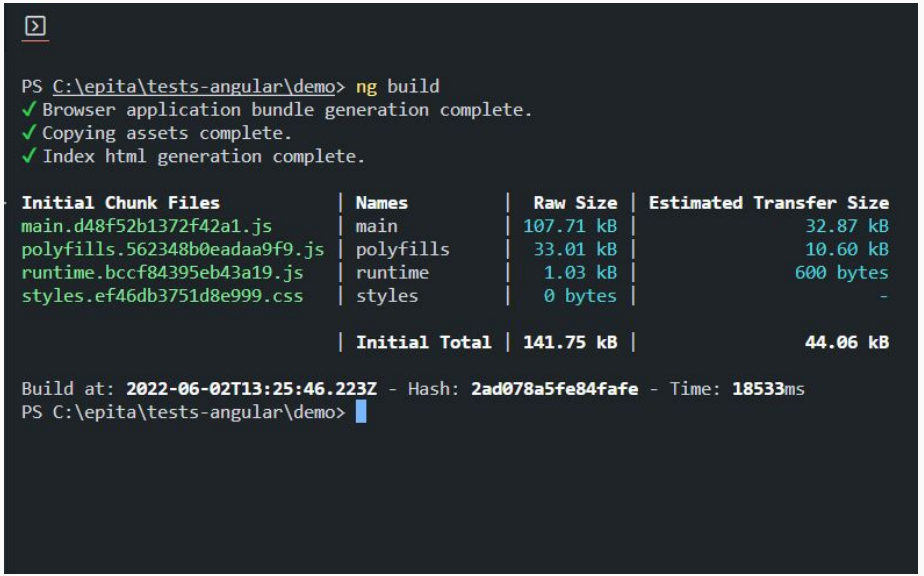
main.js contient le code principal de l'application et tous les imports,

polyfills.js contient le code de retro-compatibilité, par exemple si l'on cible la compilation avant ES2015,

runtime.js est du code utilitaire utilisé par Webpack pour charger les éléments du code principal de l'application,

styles.css contient le CSS minifié de l'application.

https://angular.io/guide/deployment



2 Build et base href

Dans index.html nous retrouvons la balise <base> . Elle définit l'URL de base à utiliser pour recomposer toutes les URLs relatives contenues dans un document. Il ne peut y avoir qu'un élément <base> au sein d'un document.

Dans un environnement local, nous pouvons redéfinir la base pour le build, via la commande **ng build** --base-href dist/demo/

Dans un environnement online, nous pouvons redéfinir la base pour le build, via la commande **ng build** --base-href http://urlapplication.com/

Grâce à l'option --base-href les fichiers chargés depuis le document HTML (*.js et *.css) seront chargés à partir de l'url définie.

https://developer.mozilla.org/fr/docs/Web/HTML/Element/base

3 Base d’une intégration continue

```
npm install
```

1 Installer les dépendances listées dans package.json

```
ng test --watch=false --browser=ChromeHeadless
```

2 Exécuter les tests sans navigateur ni live-reload

```
ng build --base href dist/demo/
```

3 Compiler l’application pour la production

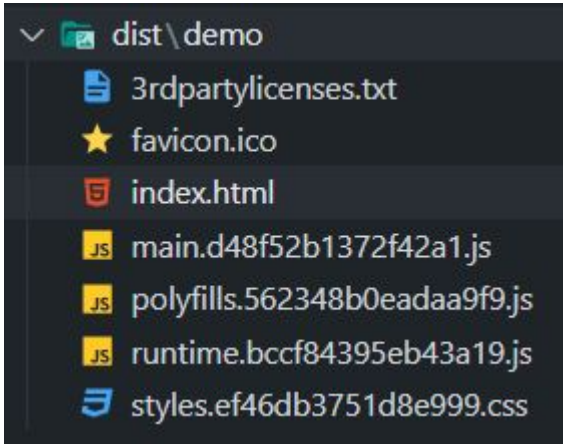
Nous pouvons créer une seule commande
pour effectuer ces tâches, dans packckage.json

```
npm run ci
```

```
"test" : "ng test",
"test-headless" :"ng test --watch=false --browsers=ChromeHeadless",
"build-prod" : "ng build --base-href /dist/movies/",
"ci": "npm install && npm run test-headless && npm run build-prod"
```

https://angular.io/cli/build

4 Le dossier dist



Le contenu du dossier dist/<votre-application> est directement déposable sur un espace de stockage (AWS S3 par exemple). Il suffit alors de pointer une url vers ce dossier. Depuis un client (browser), le fichier index.html sera chargé et l'application sera exécutée.

Pour gérer l'intégration continue en revanche, cela nécessite un environnement nodeJs. Il est alors possible d'utiliser notre propre serveur nodeJs, ou bien un service facilitant l'intégration continue d'une application front-end, comme Netlify ou autre.

Netlify permet en effet de raccorder un repo github, et le serveur exécutera une suite de tâches nécessaires (installation, test, et build), dès qu'il sera notifié d'un push sur la branche que l'on a sélectionnée.

https://angular.io/cli/build https://www.netlify.com/