|  |  |
| --- | --- |
| Term | Definition or instruction… |
| **Requirement** | -- nodejs runtime javascript  -- npm node package manager  -- Augury system de débogage spécifique  -- CLI (Cmd line interface) outil qui vous permet d’exécuter des scripts depuis l’invite de cmd pour la création, structuration et la production |
| **Description** | -- C’est plus qu’un frameworkJS il se présente comme plateforme s’appuie sur le principe d’organisation par composent c.à.d. leur structure est une arborescence de composent appeler component qui nous permettre de crée (single page app) cross plateforme (web, mobile & desktop …) |
| **lazyLoading** | C’est une technique moderne permettant de charger du code applicatif lorsque celui-ci sera sollicité, plutôt que de charger toute l'application dès son démarrage.  L'idée c'est d'alléger et d'optimiser l'application coté performances |
| Code Example  ng new App --routing --style=scss  cd App  ng g m ./mymodule1 --route mymodule1 --module app.module  ------------------------------output---------------------------  CREATE src/app/mymodule1/mymodule1-routing.module.ts  CREATE src/app/mymodule1/mymodule1.module.ts  CREATE src/app/mymodule1/mymodule1.component.scss  CREATE src/app/mymodule1/mymodule1.component.html  CREATE src/app/mymodule1/mymodule1.component.ts  UPDATE src/app/app-routing.module.ts  ng g c ./mymodule1/component1 --module mymodule1  ------------------------------output---------------------------  CREATE src/app/mymodule1/component1/component1.component.scss  CREATE src/app/mymodule1/component1/component1.component.html  CREATE src/app/mymodule1/component1/component1.component.ts  UPDATE src/app/mymodule1/mymodule1.module.ts  app-routing.module.ts\*  const routes: Routes = [  {  path: 'lazy-path-one',  loadChildren: () => import('./mymodule1/mymodule1.module')  .then(m => m.mymodule1Module)  }]; |
| **Interpolation** | {{ var or function }}  script.ts ==> html tmpl |
| **Propribinding** | One way binding [prop]=”var” | Tooway binding [(prop)]=”var”  script.ts ==> html tmpl |
| **Derective** | il sont des instructions relative à un composant angular  --Derective structurel (ngif, ngfor… ) structuration du document selon des conditions…  --Derective par attribut (ngStyle, ngClass, ngModel …) modifier le comportement des coponnent déjà dans le DOM |
| **ngIf & ngFor** | <**tr \*ngFor="let *item* of collection; let *i*=*index*; last as last"**>  <**/tr**>  <**label \*ngIf="condition;else *other\_content*"**>  ....  </**label>**  <**ng-template #other\_content**>  ....  </**ng-template**> |
| **Pipes** | Personnaliser l’affichage de données sans modifier leurs natures |
| **Globle infos** | --implement OnInit moment de création de component et après exc de constructeur  --observable émetteur des données dans le temps tt les rq http expl  --sass vs css code réutilisable avec moins line de code |