I. Programmation Angular

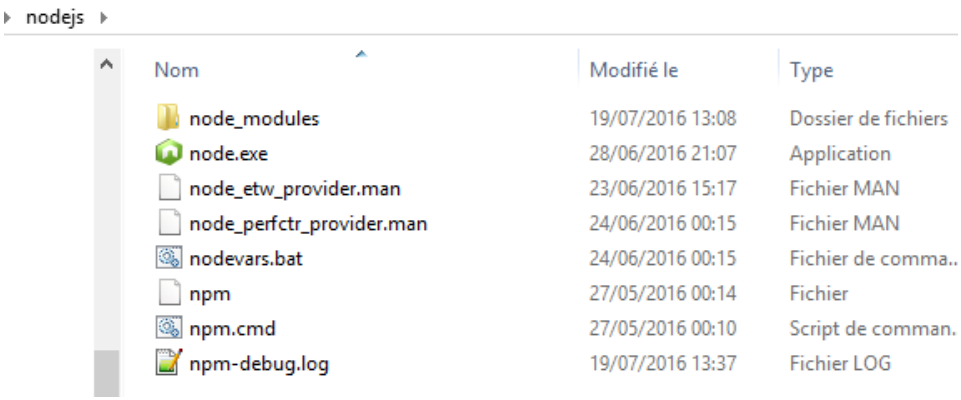
Table des matières

II. Plateforme de développement

1.Prérequis et installation

2. Installation de node.js

download node.js



Disponibilité de npm pour gérer la packages

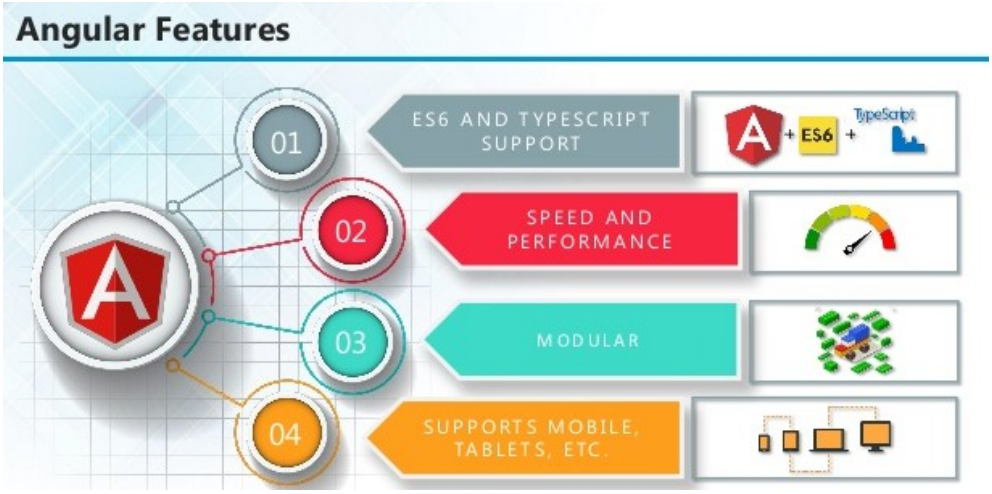
```
>npm
Usage: npm <command>
where <command> is one of:
 access, add-user, adduser, apihelp, author, bin, bug cache, completion, config, ddp, dedupe, deprecate, d
dist-tags, docs, edit, explore, faq, find, find-dupe    help, help-search, home, i, info, init, install, isslink,
list, ll, ln, login, logout, ls, outdated, ownpack, ping, prefix, prune, publish, r, rb, rebuild,  repo, restart,
rm, root, run-script, s, se, search,  show, shrinkwrap, star, stars, start, stop, t, tag,test, tst, un, uninstall,
unlink, unpublish, unstar, update, upgrade, v, version, view, whoami
npm <cmd> -h     quick help on <cmd>
npm -l           display full usage info
npm faq          commonly asked questions
npm help <term>  search for help on <term>
npm help npm     involved overview

Specify configs in the ini-formatted file:
    C:\Users\stagiaire\.npmrc
or on the command line via: npm <command> --key value
Config info can be viewed via: npm help config
```

upgrade

```
>npm install -g update-v8
```

Installation de angular

```
>npm install -g @angular/cli
```



Architecture

Commande ng

```
>ng | more
ng build <options...>
  Builds your app and places it into the output path (dist/ by default).
  --target (String) (Default: development) Defines the build target.
    aliases: -t <value>, -dev (--target=development), -prod (--target=production), --target <value>
  --environment (String) Defines the build environment.
    aliases: -e <value>, --environment <value>
  --output-path (Path) Path where output will be placed.
    aliases: -op <value>, --outputPath <value>
  --aot (Boolean) Build using Ahead of Time compilation.
    aliases: -aot
  --sourcemaps (Boolean) Output sourcemaps.
    aliases: -sm, --sourcemap, --sourcemaps
  --vendor-chunk (Boolean) Use a separate bundle containing only vendor libraries.
```

3.Création d'un projet

```
>ng new exo2
  create exo2/e2e/app.e2e-spec.ts (286 bytes)
  create exo2/e2e/app.po.ts (208 bytes)
  create exo2/e2e/tsconfig.e2e.json (235 bytes)
  create exo2/karma.conf.js (923 bytes)
  create exo2/package.json (1288 bytes)
  create exo2/protractor.conf.js (722 bytes)
  create exo2/README.md (1020 bytes)
  create exo2/tsconfig.json (363 bytes)
  create exo2/tslint.json (3012 bytes)
```

```
  create exo2/.angular-cli.json (1239 bytes)
  create exo2/.editorconfig (245 bytes)
  create exo2/.gitignore (529 bytes)
  create exo2/src/assets/.gitkeep (0 bytes)
  create exo2/src/environments/environment.prod.ts (51 bytes)
  create exo2/src/environments/environment.ts (387 bytes)
  create exo2/src/favicon.ico (5430 bytes)
  create exo2/src/index.html (291 bytes)
  create exo2/src/main.ts (370 bytes)
  create exo2/src/polyfills.ts (2405 bytes)
  create exo2/src/styles.css (80 bytes)
  create exo2/src/test.ts (642 bytes)
  create exo2/src/tsconfig.app.json (211 bytes)
  create exo2/src/tsconfig.spec.json (283 bytes)
  create exo2/src/typings.d.ts (104 bytes)
  create exo2/src/app/app.module.ts (316 bytes)
  create exo2/src/app/app.component.html (1141 bytes)
  create exo2/src/app/app.component.spec.ts (986 bytes)
  create exo2/src/app/app.component.ts (207 bytes)
  create exo2/src/app/app.component.css (0 bytes)
 Installing packages for tooling via npm.
 >node-gyp rebuild > build_log.txt 2>&1 || exit 0
```

Installation des dépendances

```
 exo2>npm install
 npm notice created a lockfile as package-lock.json. You should commit this file.
 npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.1.3
 (node_modules\fsevents):
 npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@
 1.1.3: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})
 up to date in 27.425s
```

Lancer un serveur de dev

```
 >ng serve
 .............


  29% building modules 162/163 modules 1 active ..._modules\core-js\modules\_is-a
  29% building modules 162/164 modules 2 active ...odules\core-js\modules\_to-iob
  29% building modules 162/165 modules 3 active ...es\core-js\modules\_array-incl
  29% building modules 163/165 modules 2 active ...es\core-js\modules\_array-incl
  29% building modules 164/165 modules 1 active ...es\core-js\modules\_array-incl
  29% building modules 165/166 modules 1 active ...core-js\modules\_to-absolute-i
 Date: 2018-02-05T21:25:23.687Z
 Hash: 113d5280fe3be5e9c68b
```

```
Time: 13824ms
chunk {inline} inline.bundle.js (inline) 5.79 kB [entry] [rendered]
chunk {main} main.bundle.js (main) 19.5 kB [initial] [rendered]
chunk {polyfills} polyfills.bundle.js (polyfills) 551 kB [initial] [rendered]
chunk {styles} styles.bundle.js (styles) 33.7 kB [initial] [rendered]
chunk {vendor} vendor.bundle.js (vendor) 7.41 MB [initial] [rendered]

webpack: Compiled successfully.
```

Code généré

```
Contenu de src\app>
app.component.css
app.component.html
app.component.spec.ts
app.component.ts
app.module.ts
```

- app.component.ts— the component class code, written in TypeScript.
- app.component.html— the component template, written in HTML.
- app.component.css— the component's private CSS styles.

- angular-cli.json: a configuration file for Angular CLI.
- package.json: a standard file for Node-based projects.
- protractor.conf.js: Protractor is a tool for running end-to-end tests for Angular projects.
- karma.conf.js: Karma is a test runner for JavaScript applications.
- tsconfig.json: includes setting for the TypeScript compiler.

Un composant peut contenir un autre composant

```
AppComponent : NavBar,SideBar,ContentArea
```

Création d'un composant

Elément du composant

- selector
- templateUrl
- styleUrls

```
>ng generate component product
src\app>ng generate component moncomposant
create src/app/moncomposant/moncomposant.component.html (31 bytes)
create src/app/moncomposant/moncomposant.component.spec.ts (670 bytes)
create src/app/moncomposant/moncomposant.component.ts (293 bytes)
create src/app/moncomposant/moncomposant.component.css (0 bytes)
update src/app/app.module.ts (422 bytes)
```

Code associé au moncomposant.ts : annotation (@Component)

```
import { Component, OnInit } from '@angular/core';
```

```
@Component({
  selector: 'app-moncomposant',
  templateUrl: './moncomposant.component.html',
  styleUrls: ['./moncomposant.component.css']
})
export class MoncomposantComponent implements OnInit {
  constructor() { }
  ngOnInit() {
  }
}
```

Visuel du composant

```
<h1>
      moncomposant works!
</h1>
```

Intégration dans la page HTML du site

```
<div style="text-align:center">
  <h1>
    Welcome to {{ title }}!
  </h1>
  <app-moncomposant></app-moncomposant>
</div>
```

Equivalence

```
<p>{{title}}</p>
<p [innerHtml]="title"></p>
```

Ajouter une donnée dans le composant

```
export class MoncomposantComponent implements OnInit {

  message = "Coucou";
  constructor() { }

  ngOnInit() {
  }

}
```

Utilisation dans le rendu du composant

```
<h1>
      moncomposant marche bien!
    Message = {{ message }}!
</h1>
```

Ajouter un traitement dans le composant

```
export class MoncomposantComponent implements OnInit {
```

```
        message = "Coucou";
        constructor() { }
        ngOnInit() {
        }
        modifierMessage() {
                this.message = "nouveau coucou";
        }
    }
```

Utilisation dans le rendu

```
    <h1>
        moncomposant marche bien!
      Message = {{ message }}!
        <button (click)="modifierMessage()">Changer de message</button>
    </h1>
```

app.module.ts

```
    import { BrowserModule } from '@angular/platform-browser';
    import { NgModule } from '@angular/core';

    import { AppComponent } from './app.component';
    import { MoncomposantComponent } from './moncomposant/moncomposant.component';

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

Template et style du composant

template au lieu de templateUrl :moncomposant.component.ts

```
    import { Component, OnInit } from '@angular/core';

    @Component({
      selector: 'app-moncomposant',
      //templateUrl: './moncomposant.component.html',
      template: '<h2>COUCOU</h2>',
      styleUrls: ['./moncomposant.component.css']
    })

    export class MoncomposantComponent implements OnInit {

        message = "Coucou";
```

```
        constructor() { }

        ngOnInit() {
        }

        modifierMessage() {
                this.message = "nouveau coucou";
        }
    }
```

Prise en compte dans le rendu : moncomposant.component.html

```
    <h1>
        moncomposant marche bien!
      Message = {{ message }}!
        <button (click)="modifierMessage()">Changer de message</button>
    </h1>
```

moncomposant.component.css

```
    h2 {
        background : yellow;
        color : blue
    }
```

Sur plusieurs lignes

```
     template: `
    <h2>COUCOU</h2>`,
```

4. class AppComponent

Ajoute un objet dans AppComponent directement

app.component.ts

```
    import { Component } from '@angular/core';

    @Component({
      selector: 'app-root',
      templateUrl: './app.component.html',
      styleUrls: ['./app.component.css']
    })
    export class AppComponent {
      title = 'app';
      // Ajouter un objet
     personne = {
        nom: 'TOTO1',
        prenom: 'toto1',
        age: 1
          getAll : function () {
                return this.nom + "::" +this.prenom + "::" +this.age
          }
      };
    }
```

Prise en compte dans le rendu principal : app.component.html

```
<h1>
  Titre = {{ title }}<br/>
  Personne = {{ personne }}<br/>
  Personne/nom = {{ personne.nom }}<br/>
  Personne/getAll = {{ personne.getAll() }}<br/>
</h1>
```

Affichage

```
Titre = app
Personne = [object Object]
Personne/nom = TOTO1
Personne/getAll = TOTO1::toto1::1
```

5. ngFor et Iterables

AppComponent

```
export class AppComponent {
  title = 'app';
  data = ["TOTO1", "TOTO2", "TOTO3", "TOTO4", "TOTO5"];
}
```

Utilisation dans le rendu

```
<ul>
<li *ngFor="let d of data">{{ d }}</li>
</ul>
```

Affichage

```
TOTO1
TOTO2
TOTO3
TOTO4
TOTO5
```

6. Création de services

```
src\app>ng g service monservice
  create src/app/monservice.service.spec.ts (398 bytes)
  create src/app/monservice.service.ts (116 bytes)
```

Code du service : monservice.service.ts

```
import { Injectable } from '@angular/core';

@Injectable()
export class MonserviceService {

  constructor() { }

}
```

Ajouter données + méthode

```
import { Injectable } from '@angular/core';

@Injectable()
export class MonserviceService {

  constructor() { }
  ladata = ["TOTO1", "TOTO2", "TOTO3", "TOTO4", "TOTO5"];
  getData ()
  {
      return this.data;
  }
}
```

Prise en compte du service : app.component.ts

```
export class AppComponent {
  title = 'app';
  ladata = []
  constructor(private dataService:MonserviceService) {
      console.log (dataService)
      this.ladata = dataService.getData()
  }
}
```

app.module.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppComponent } from './app.component';
import { MoncomposantComponent } from './moncomposant/moncomposant.component';
import { MonserviceService } from './monservice.service';

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

7.Déployer l'application

Création du répertoire dist

```
>ng build
…..
```

```
Date: 2018-02-06T06:34:50.310Z
Hash: 63fbe8d60368dd4e7399
Time: 21411ms
chunk {0} polyfills.f20484b2fa4642e0dca8.bundle.js (polyfills) 59.4 kB [initial][rendered]
chunk {1} main.c67aa9c05c7e6c27c80c.bundle.js (main) 157 kB [initial] [rendered]
chunk {2} styles.9c0ad738f18adc3d19ed.bundle.css (styles) 79 bytes [initial] [rendered]
chunk {3} inline.65f787396c0d67336529.bundle.js (inline) 1.45 kB [entry] [rendered]
```

Installation du gestionnaire de package

```
exo2>npm i -g angular-cli-ghpages
C:\Users\stagiaire\AppData\Roaming\npm\angular-cli-ghpages -> C:\Users\stagiaire
\AppData\Roaming\npm\node_modules\angular-cli-ghpages\bin\angular-cli-ghpages
C:\Users\stagiaire\AppData\Roaming\npm\ngh -> C:\Users\stagiaire\AppData\Roaming
\npm\node_modules\angular-cli-ghpages\bin\angular-cli-ghpages
+ angular-cli-ghpages@0.5.2
added 39 packages in 15.785s
```

Remonter tout sur Git

```
> git init
> git add .
> git commit -m "First commit"
> git remote add origin remote-repository-url
> git push origin master
```

8.Directive

Directive simple

Création de la directive

```
exo2>ng g directive madirective
  create src/app/madirective.directive.spec.ts (244 bytes)
  create src/app/madirective.directive.ts (151 bytes)
  update src/app/app.module.ts (585 bytes)
```

Code généré

```
import { Directive,  ElementRef } from '@angular/core';

@Directive({
  selector: '[appMadirective]'
})
export class MadirectiveDirective {

    constructor(private el: ElementRef) {
      el.nativeElement.style.backgroundColor = 'yellow';
    }
}
```

Prise en compte dans le rendu principal

```
<div style="text-align:center">
  <app-moncomposant></app-moncomposant>
  <h1>
      Titre = {{ title }}<br/>
      appMadirective <div appMadirective>AAAAA</div>
  </h1>
</div>
```

Directive avec événements

Si l'attribut est sur un bouton, gérer le click

```
import { Directive,  ElementRef } from '@angular/core';

@Directive({
  selector: '[appMadirective]'
})
export class MadirectiveDirective {

      traitement()
      {
            alert("Ok")
      }

    constructor(private el: ElementRef) {
      el.nativeElement.style.backgroundColor = 'yellow';
      el.nativeElement.onclick = this.traitement
    }
}
```

Prise en compte dans le rendu principal

```
<div style="text-align:center">
  <app-moncomposant></app-moncomposant>
  <h1>
    Titre = {{ title }}<br/>
      appMadirective <button appMadirective>Je clicke</button>
  </h1>
</div>
```

9. Gestion de pipe

Création du composant pipe

```
exo2>ng g pipe monpipe
  create src/app/monpipe.pipe.spec.ts (191 bytes)
  create src/app/monpipe.pipe.ts (203 bytes)
  update src/app/app.module.ts (648 bytes)
```

Code généré

```
import { Pipe, PipeTransform } from '@angular/core';

@Pipe({
```

```
  name: 'monpipe'
})
export class MonpipePipe implements PipeTransform {

  transform(value: number, args?: any): any {
    return value * 2;
  }
}
```

Prise en compte de app.module.ts

```
import { MonpipePipe } from './monpipe.pipe';

@NgModule({
  declarations: [
    AppComponent,
    MoncomposantComponent,
    MadirectiveDirective,
    MonpipePipe
  ],
  imports: [
    BrowserModule
  ],
  providers: [MonserviceService],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Prise en compte dans le rendu principal

```
<div style="text-align:center">
  <app-moncomposant></app-moncomposant>
  <h1>
      Titre = {{ title }}<br/>
      Le double de 10 = {{ 10 | monpipe}}<br/>
  </h1>
</div>
```

Affichage

```
Titre = app
Le double de 10 = 20
```

10. Orienté objet

Angular utilise les langages Javscript(es5) et TypeScript.

Classe et implémentation d'interface

```
export class ProductComponent implements OnInit {
      constructor() { }
      ngOnInit() {
            }
}
```

Angular ne travaille pas sur DOM directement

```
$("#myElement").text("something")
```

11. Gestion de modules

Les librairies de Angular sont des NgModules. Les modules permettent d'organiser l'application.

Il existe des Modules disponibles pour divers usage :
- FormsModule
- HttpClientModule
- RouterModule
- Ionic
- AngularFire2

Le module englobe tout. Il contient components, directives, and pipes. Seul ceux qui sont public seront accessibles à d'autres modules lors de leur importation.

Création d'un module

```
C:\wamp\www\angularv4\exo2>ng g module monmodule
  create src/app/monmodule/monmodule.module.ts (193 bytes)
```

Options:
- app
- flat
- module
- spec
- routing

Module vierge

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';

@NgModule({
  imports: [
    CommonModule
  ],
  declarations: []
})
export class MonmoduleModule { }
```