File

`src/app/admin/components/edit-hero/edit-hero.component.ts`

Description

Component to edit a hero

Implements

`OnInit`

Metadata

selector	`app-edit-hero`
styleUrls	`./edit-hero.component.scss`
templateUrl	`./edit-hero.component.html`

Index

Properties

- Public correctData
- Public dialog
- Public editHeroForm
- Public hero
- Public idHero
- Public message

Methods

- getErrorMessage
- getInfoHero
- ngOnInit
- openSnackBar
- SearchHeroDialog
- submit

Constructor

`constructor(formBuilder: FormBuilder, _snackBar: MatSnackBar, dialog: MatDialog, _HeroService: HeroService)`

Defined in src/app/admin/components/edit-hero/edit-hero.component.ts:37

Constructor in which we inject our services and different elements

Parameters :

Name	Type	Optional
formBuilder	FormBuilder	No
_snackBar	MatSnackBar	No
dialog	MatDialog	No
_HeroService	HeroService	No

Methods

getErrorMessage

`getErrorMessage(dato)`

Defined in src/app/admin/components/edit-hero/edit-hero.component.ts:188

function to control error messages

Parameters :

Name	Optional
dato	No

Returns : "This information is required" | "The maximum of characters is 30" | "" | "You must enter at leas...

message

getInfoHero

`getInfoHero(idHero)`

Defined in src/app/admin/components/edit-hero/edit-hero.component.ts:156

Get the info about the hero

Parameters :

Name	Optional
idHero	No

Returns : void

ngOnInit

`ngOnInit()`

Defined in src/app/admin/components/edit-hero/edit-hero.component.ts:52

Start when the component inits

Returns : void

openSnackBar

`openSnackBar(message: string, action: string)`

Defined in src/app/admin/components/edit-hero/edit-hero.component.ts:232

function to open snackBars

Parameters :

Name	Type	Optional
message	string	No
action	string	No

Returns : void

SearchHeroDialog

`SearchHeroDialog()`

Defined in src/app/admin/components/edit-hero/edit-hero.component.ts:211

Open modal to search hero

Returns : void

submit

`submit()`

Defined in src/app/admin/components/edit-hero/edit-hero.component.ts:168

for submit form

Returns : `void`

Properties

Public correctData

Type : `boolean`

Defined in src/app/admin/components/edit-hero/edit-hero.component.ts:37

variable to check if the function was ok

Public dialog

Type : `MatDialog`

Defined in src/app/admin/components/edit-hero/edit-hero.component.ts:45

Public editHeroForm

Type : `FormGroup`

Defined in src/app/admin/components/edit-hero/edit-hero.component.ts:25

to add FormGroup

Public hero

Type : `Hero`

Defined in src/app/admin/components/edit-hero/edit-hero.component.ts:21

variable to store the hero to edit

Public idHero

Type : `number`

Default value : `0`

Defined in src/app/admin/components/edit-hero/edit-hero.component.ts:29

variable for id Hero

Public message

Type : `string`

Defined in src/app/admin/components/edit-hero/edit-hero.component.ts:33

variable to save message info

```
import { Component, OnInit } from '@angular/core';
import { Hero } from 'src/app/models/hero';
import { FormGroup, FormBuilder, Validators } from '@angular/forms';
import { HeroService } from 'src/app/services/hero.service';
import { MatDialog } from '@angular/material';
import { selectHeroComponent } from 'src/app/components/modals/select-hero-dialog/select-hero-dialog.component';
import { MatSnackBar } from '@angular/material';
/**
 * Component to edit a hero
 */
@Component({
  selector: 'app-edit-hero',
  templateUrl: './edit-hero.component.html',
  styleUrls: ['./edit-hero.component.scss']
})

export class EditHeroComponent implements OnInit {
  /**
   * variable to store the hero to edit
   */
  public hero: Hero;
  /**
   * to add FormGroup
   */
```

```
public editHeroForm: FormGroup
/**
 * variable for id Hero
 */
public idHero: number = 0
/**
 * variable to save message info
 */
public message: string
/**
 * variable to check if the function was ok
 */
public correctData: boolean

/**
 * Constructor in which we inject our services and different elements
 */
constructor(
  private formBuilder: FormBuilder,
  private _snackBar: MatSnackBar,
  public dialog: MatDialog,
  private _HeroService: HeroService,
) { }

/**
 * Start when the component inits
 */
ngOnInit() {
  this.hero = new Hero(0, '', '', 0, 0, 0, 0, 0, 0, '', '', '', '', '', '', '', '', '', '', '', '', '', null)

  this.editHeroForm = this.formBuilder.group({
    heroName: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    image: [
      '',
      [

        Validators.minLength(1),
        Validators.maxLength(300)
      ]
    ],
    intelligence: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    strength: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    speed: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    durability: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    power: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    combat: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    fullName: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    placeOfBirth: ['', [Validators.minLength(1), Validators.maxLength(30)]],
    publisher: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    alignment: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    firstApperance: [
      '',
      [

        Validators.minLength(1),
        Validators.maxLength(300)
      ]
```

```
    ],
    gender: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    race: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    height: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    weight: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    eyeColor: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    hairColor: [
      '',
      [Validators.minLength(1), Validators.maxLength(30)]
    ],
    work: [
      '',
      [Validators.minLength(1), Validators.maxLength(300)]
    ],
    biography: [
      '',
      [

        Validators.minLength(1),
        Validators.maxLength(300)
      ]
    ]
  })
}

/**
 * Get the info about the hero
 * @param {number}idHero
 */
getInfoHero(idHero) {
  this._HeroService.getHeroById(idHero).subscribe(res => {
    this.hero = res
  }, err => {
    console.log(err)
  })

}

/**
 * for submit form
 */
submit() {
  this._HeroService.modifyHero(this.idHero, this.hero).subscribe(
    res => {
      this.message = 'Create correctly';
      this.correctData = true;
      this.openSnackBar('MODIFY CORRECTLY', 'Close')
    },
    err => {
       console.error(err);
        this.message = 'create hero failed';
        this.openSnackBar('MODIFY HERO FAILED', 'Close')}
  )

}

/**
 * function to control error messages
 * @param {string} dato
 * @returns message
 */
getErrorMessage(dato) {
  var result: string
  if (this.editHeroForm.controls[dato].hasError('required')) {
    return (result = 'This information is required')
  } else if (this.editHeroForm.controls[dato].hasError('minlength')) {

    return (result = 'You must enter at least 1 characters')

  } else if (this.editHeroForm.controls[dato].hasError('maxlength')) {
    if (dato === 'image' || dato === 'firstAppearance' || dato === 'biography' || dato === 'work') {
      return (result = 'The maximum of characters is 300')
```

```
      } else {

        return (result = 'The maximum of characters is 30')
      }
    } else {
      return (result = '')
    }
  }

  /**
   * Open modal to search hero
   */
  SearchHeroDialog(): void {
    const dialogRef = this.dialog.open(selectHeroComponent, {
      data: {
        action: 'edit'
      }
    });
    dialogRef.afterClosed().subscribe(result => {
      console.log("The dialog was closed");
      if (result) {
        this.idHero = result;
        this.getInfoHero(result)
      }
    });
  }


  /**
   * function to open snackBars
   *  @param {string} message
   *  @param {string} action
   */
  openSnackBar(message: string, action: string) {
    this._snackBar.open(message, action, {
      duration: 8000,
      panelClass: ['blue-snackbar']
    })
  }

}
```

```
<div class="adminDetail">
  <div id="form">
    <div class="title">
      <h1>Edit Hero</h1>
      <button
        type="button"
        class="UnfollowB"
        mat-raised-button
        (click)="SearchHeroDialog()"
      >
        <span>SEARCH HERO</span>
      </button>
    </div>
    <form [formGroup]="editHeroForm" (ngSubmit)="submit()">
      <mat-form-field appearance="legacy">
        <mat-label>heroName</mat-label>
        <input
          matInput
          type="text"
          formControlName="heroName"
          placeholder="heroName"
          [(ngModel)]="hero.heroName"
        />
        <mat-error *ngIf="!editHeroForm.controls['heroName'].valid">{{
          getErrorMessage("heroName")
        }}</mat-error>
      </mat-form-field>

      <mat-form-field appearance="legacy">
        <mat-label>image</mat-label>
        <input
          matInput
          type="text"
          formControlName="image"
          placeholder="image"
          [(ngModel)]="hero.image"
        />
        <mat-error *ngIf="!editHeroForm.controls['image'].valid">{{
          getErrorMessage("image")
        }}</mat-error>
      </mat-form-field>

      <mat-form-field appearance="legacy">
        <mat-label>fullName</mat-label>
        <input
```

```
    matInput
    type="text"
    formControlName="fullName"
    placeholder="fullName"
    [(ngModel)]="hero.fullName"
  />
  <mat-error *ngIf="!editHeroForm.controls['fullName'].valid">{{
    getErrorMessage("fullName")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>placeOfBirth</mat-label>
  <input
    matInput
    type="text"
    formControlName="placeOfBirth"
    placeholder="placeOfBirth"
    [(ngModel)]="hero.placeOfBirth"
  />
  <mat-error *ngIf="!editHeroForm.controls['placeOfBirth'].valid">{{
    getErrorMessage("placeOfBirth")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>publisher</mat-label>
  <input
    matInput
    type="text"
    formControlName="publisher"
    placeholder="publisher"
    [(ngModel)]="hero.publisher"
  />
  <mat-error *ngIf="!editHeroForm.controls['publisher'].valid">{{
    getErrorMessage("publisher")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>alignment</mat-label>
  <input
    matInput
    type="text"
    formControlName="alignment"
    placeholder="alignment"
    [(ngModel)]="hero.alignment"
  />
  <mat-error *ngIf="!editHeroForm.controls['alignment'].valid">{{
    getErrorMessage("alignment")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>firstApperance</mat-label>
  <input
    matInput
    type="text"
    formControlName="firstApperance"
    placeholder="firstApperance"
    [(ngModel)]="hero.firstApperance"
  />
  <mat-error *ngIf="!editHeroForm.controls['firstApperance'].valid">{{
    getErrorMessage("firstApperance")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>gender</mat-label>
  <input
    matInput
    type="text"
    formControlName="gender"
    placeholder="gender"
    [(ngModel)]="hero.gender"
  />
  <mat-error *ngIf="!editHeroForm.controls['gender'].valid">{{
    getErrorMessage("gender")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>race</mat-label>
  <input
    matInput
    type="text"
    formControlName="race"
```

```
    placeholder="race"
    [(ngModel)]="hero.race"
  />
  <mat-error *ngIf="!editHeroForm.controls['race'].valid">{{
    getErrorMessage("race")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>height</mat-label>
  <input
    matInput
    type="text"
    formControlName="height"
    placeholder="height"
    [(ngModel)]="hero.height"
  />
  <mat-error *ngIf="!editHeroForm.controls['height'].valid">{{
    getErrorMessage("height")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>weight</mat-label>
  <input
    matInput
    type="text"
    formControlName="weight"
    placeholder="weight"
    [(ngModel)]="hero.weight"
  />
  <mat-error *ngIf="!editHeroForm.controls['weight'].valid">{{
    getErrorMessage("weight")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label> eyeColor</mat-label>
  <input
    matInput
    type="text"
    formControlName="eyeColor"
    placeholder="eyeColor"
    [(ngModel)]="hero.eyeColor"
  />
  <mat-error *ngIf="!editHeroForm.controls['eyeColor'].valid">{{
    getErrorMessage("eyeColor")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>hairColor</mat-label>
  <input
    matInput
    type="text"
    formControlName="hairColor"
    placeholder="hairColor"
    [(ngModel)]="hero.hairColor"
  />
  <mat-error *ngIf="!editHeroForm.controls['hairColor'].valid">{{
    getErrorMessage("hairColor")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>work</mat-label>
  <input
    matInput
    type="text"
    formControlName="work"
    placeholder="work"
    [(ngModel)]="hero.work"
  />
  <mat-error *ngIf="!editHeroForm.controls['work'].valid">{{
    getErrorMessage("work")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>biography</mat-label>
  <input
    matInput
    type="text"
    formControlName="biography"
    placeholder="biography"
    [(ngModel)]="hero.biography"
  />
```

```
  <mat-error *ngIf="!editHeroForm.controls['biography'].valid">{{
    getErrorMessage("biography")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>intelligence</mat-label>
  <input
    matInput
    type="number"
    placeholder="intelligence"
    [(ngModel)]="hero.intelligence"
    formControlName="intelligence"
    min="1"
    max="100"
  />
  <span matSuffix></span>
  <mat-error *ngIf="!editHeroForm.controls['intelligence'].valid">{{
    getErrorMessage("intelligence")
  }}</mat-error>
</mat-form-field>
<mat-form-field appearance="legacy">
  <mat-label>strength</mat-label>
  <input
    matInput
    type="number"
    placeholder="strength"
    [(ngModel)]="hero.strength"
    formControlName="strength"
    min="1"
    max="100"
  />
  <span matSuffix></span>
  <mat-error *ngIf="!editHeroForm.controls['strength'].valid">{{
    getErrorMessage("strength")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>speed</mat-label>
  <input
    matInput
    type="number"
    placeholder="speed"
    [(ngModel)]="hero.speed"
    formControlName="speed"
    min="1"
    max="100"
  />
  <span matSuffix></span>
  <mat-error *ngIf="!editHeroForm.controls['speed'].valid">{{
    getErrorMessage("speed")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>durability</mat-label>
  <input
    matInput
    type="number"
    placeholder="durability"
    [(ngModel)]="hero.durability"
    formControlName="durability"
    min="1"
    max="100"
  />
  <span matSuffix></span>
  <mat-error *ngIf="!editHeroForm.controls['durability'].valid">{{
    getErrorMessage("durability")
  }}</mat-error>
</mat-form-field>

<mat-form-field appearance="legacy">
  <mat-label>power</mat-label>
  <input
    matInput
    type="number"
    placeholder="power"
    [(ngModel)]="hero.power"
    formControlName="power"
    min="1"
    max="100"
  />
  <span matSuffix></span>
  <mat-error *ngIf="!editHeroForm.controls['power'].valid">{{
    getErrorMessage("power")
  }}</mat-error>
```

```
      </mat-form-field>

      <mat-form-field appearance="legacy">
        <mat-label>combat</mat-label>
        <input
          matInput
          type="number"
          placeholder="combat"
          [(ngModel)]="hero.combat"
          formControlName="combat"
          min="1"
          max="100"
        />
        <span matSuffix></span>
        <mat-error *ngIf="!editHeroForm.controls['combat'].valid">{{
          getErrorMessage("combat")
        }}</mat-error>
      </mat-form-field>

      <div *ngIf="!correctData">
        <p class="mensajeError">{{ this.message }}</p>
      </div>
      <br />
      <button type="submit" class="modifyHero" [disabled]="!editHeroForm.valid">
        MODIFY HERO
      </button>
    </form>
  </div>
</div>
```

./edit-hero.component.scss

```
// button
$bg: #272727;
$fg: #f3d403;
$border-width: 0.2rem;
$corner-size: 3rem;
$dur: 0.3s;

.UnfollowB {
  margin: 0 0.5vw 3vw 0.5vw;
  outline: none;
  font-family: "B612";
  letter-spacing: 0.02rem;
  cursor: pointer;
  background: transparent;
  border: $border-width solid currentColor;
  padding: 0.1rem 0.5rem;
  font-size: 1rem;
  color: $fg;
  position: relative;
  transition: color $dur;

  &:hover {
    color: #cc4224;
    &::before {
      width: 0;
    }
    &::after {
      height: 0;
    }
  }
  &:active {
    border-width: $border-width / 2;
  }
  span {
    position: relative;
    z-index: 2;
  }
  &::before,
  &::after {
    content: "";
    position: absolute;
    background: $bg;
    z-index: 1;
    transition: all $dur;
  }
  &::before {
    width: calc(100% - #{$corner-size});
    height: calc(101% + #{$border-width * 2});
    top: -$border-width;
    left: 50%;
    transform: translateX(-50%);
  }
  &::after {
    height: calc(100% - #{$corner-size});
```

```
    width: calc(101% + #{$border-width * 2});
    left: -$border-width;
    top: 50%;
    transform: translateY(-50%);
  }
}

.adminDetail {
  border: 2.5px solid #00a23d;
  box-shadow: 0 0 21px 9px rgba(8, 17, 10, 0.63);
  padding: 0.5em;
  margin: 0.5em;
  background-color: #272727;
  color: white;
  margin-bottom: 3em;
}

#form .mat-form-field {
  margin-left: 12px;
}

.modifyHero {
  outline: none;
  background-color: #f3d403;
  border: none;
  border-radius: 10px;
  border: 2.5px solid #00a23d;
  padding: 9px;
  font-size: 0.8rem;
  color: black;
}

.modifyHero:hover {
  background-color: #cc4224;
  color: rgb(223, 221, 221);
}

.title {
  display: flex;
  flex-direction: column;
}

#form ::ng-deep .mat-form-field-appearance-legacy .mat-form-field-label {
  color: #709e82;
}

@media (min-width: 992px) {
  .title {
    display: flex;
    flex-direction: row;
  }

  .adminDetail {
    margin: 3em;
    padding: 2em;
  }
}
```

Legend
Html element
Component
Html element with directive

result-matching ""

No results matching ""