



# Angular 2+

## Développer des applications Web

Support de cours

Réf. T44A-046





# Module 10

## Mise en place d'un service asynchrone

# Contenu du module

- Création d'un service asynchrone
- Le service devient Injectable
- Création d'un Observable
- La méthode **emit**
- L'envoi de données avec la méthode emit
- Abonnement aux données du service

# Service asynchrone

- Si notre service communique avec une API
  - Nous devons attendre une réponse de l'API
- Et donc les données arriveront après le chargement du composant

# Service asynchrone

- Le Service va émettre un objet de type **subject**
- Un abonnement à un **subject** est nécessaire
  - Avec l'objet **subscription** depuis le composant av

# Observer / Observable

- un **Observable** est un objet qui émet des informations auxquelles on souhaite réagir.
- Emet un code qui sera exécuté à chaque fois que **l'Observable** émet une information
- Pour observer **l'observable** on utilise la fonction **subscribe()** :
  - Grace à un objet de type **Subscription**


# Observer / Observable

- Le service va emmètre un **subject**
- Le component va prendre un abonnement **subscription**
- Méthaphore abonnement à **Sciences & vie**

# Observer / Observable

- Avec l'utilisation des observers et Observables **RxJS**
- Le service devient injectable

```
import { HttpClient } from '@angular/common/http';  
import { Subject } from 'rxjs';  
import { Injectable } from '@angular/core';  
  
@Injectable()  
export class PersonneService{
```

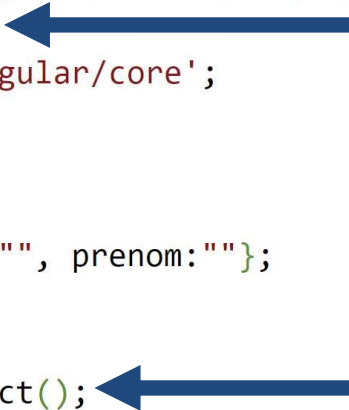




# Observer / Observable

- Mise en place d'un **Subject** : un Observable
- Le **subject** va transporter les données du service

```
import { HttpClient } from '@angular/common/http';  
import { Subject } from 'rxjs';  
import { Injectable } from '@angular/core';  
  
@Injectable()  
export class PersonneService {  
  private personne = { nom: "", prenom: "" };  
  
  // le transporteur  
  personneSubject = new Subject();
```




# Observer / Observable

- **Création** de la méthode emitPersonneSubject()
- Permet d'envoyer les données au moment où elles arrivent

```
@Injectable()
export class PersonneService{
    private personne = { nom : "", prenom: ""};
    // le transporteur
    personneSubject =new Subject();
    constructor(private httpClient:HttpClient){
    }

    emitPersonneSubject(){
        this.personneSubject.next(this.personne);
    }
}
```




# Observer / Observable

- **Création** de la méthode emitPersonneSubject()
- Permet d'envoyer les données au moment où elles arrivent

```
@Injectable()
export class PersonneService{
    private personne = { nom : "", prenom: "" };
    // le transporteur
    personneSubject = new Subject();
    constructor(private httpClient:HttpClient){
    }


    emitPersonneSubject(){
        this.personneSubject.next(this.personne);
    }
}
```



# Observer / Observable

- A chaque action la méthode **emit...** est appelé pour envoyer les données

```
loadFire(){  
  let url = 'https://gestion-equipe.firebaseio.com/client.json';  
  this.httpClient.get<any>(url)  
    .subscribe(  
      (response)=>{  
        if ( response != undefined){  
          this.personne = response;  
        }  
        this.emitPersonneSubject();  
      },  
      (error)=>{  
        console.log(error);  
      });  
}
```



# Observer / Observable

- Mise en place de l'abonnement

```
ngOnInit(){  
  // mis en place de l'abonnement  
  this.personneService.personneSubject.subscribe(  
    (personne:any) =>{  
      this.personne = personne;  
      console.log('Component root :Je mets jour');  
    }  
  );  
  this.personneService.loadFire();  
}
```



# Observer / Observable

- Pour des besoins plus complexes création d'un objet **subscription**
- Pensez à l'import **Subscription** from **rxjs**

```
export class AppComponent implements OnInit{  
  personne:any= {nom:'',prenom:''};  
  personneSubscription:Subscription;  
  
  ngOnInit(){  
    this.personneSubscription =  
    this.personneService.personneSubject.subscribe(  
      (personne:any) =>{  
        this.personne = personne;  
        console.log('Component root :Je mets jour');  
      }  
    );  
    this.personneService.loadFire();  
  }  
}
```

