Angular : Successeur de Angular.js qui était une librairie. C'est un framework qui supporte ton projet dès le départ. Son but est d'apporter tous les outils nécessaires à la construction d'une SPA.

CLI : Une ligne de commande qui nous aide lors du développement dans Angular.

Composants : Il est responsable de l'affichage d'une partie de la page. Il gère donc :
- le rendu / la vue (HTML/CSS)
- les intéractions (JS → TS).

Rendu
- HTML
- CSS

Intéraction
- TS / Classe

1 Classe TS
- Selector ✓
- templateUrl ✓
- styleUrls ✓

@Component

Interpolation : Permet d'afficher une propriété de la classe TS dans le template HTML : {{ variable }}
{{ fonction() }}
{{ 2 + 3 + var }}

Event Binding : Permet de lier à un événement d'un élément une fonction de la classe TS : (event) = "fonction()"
$event : représente le détail de l'événement.

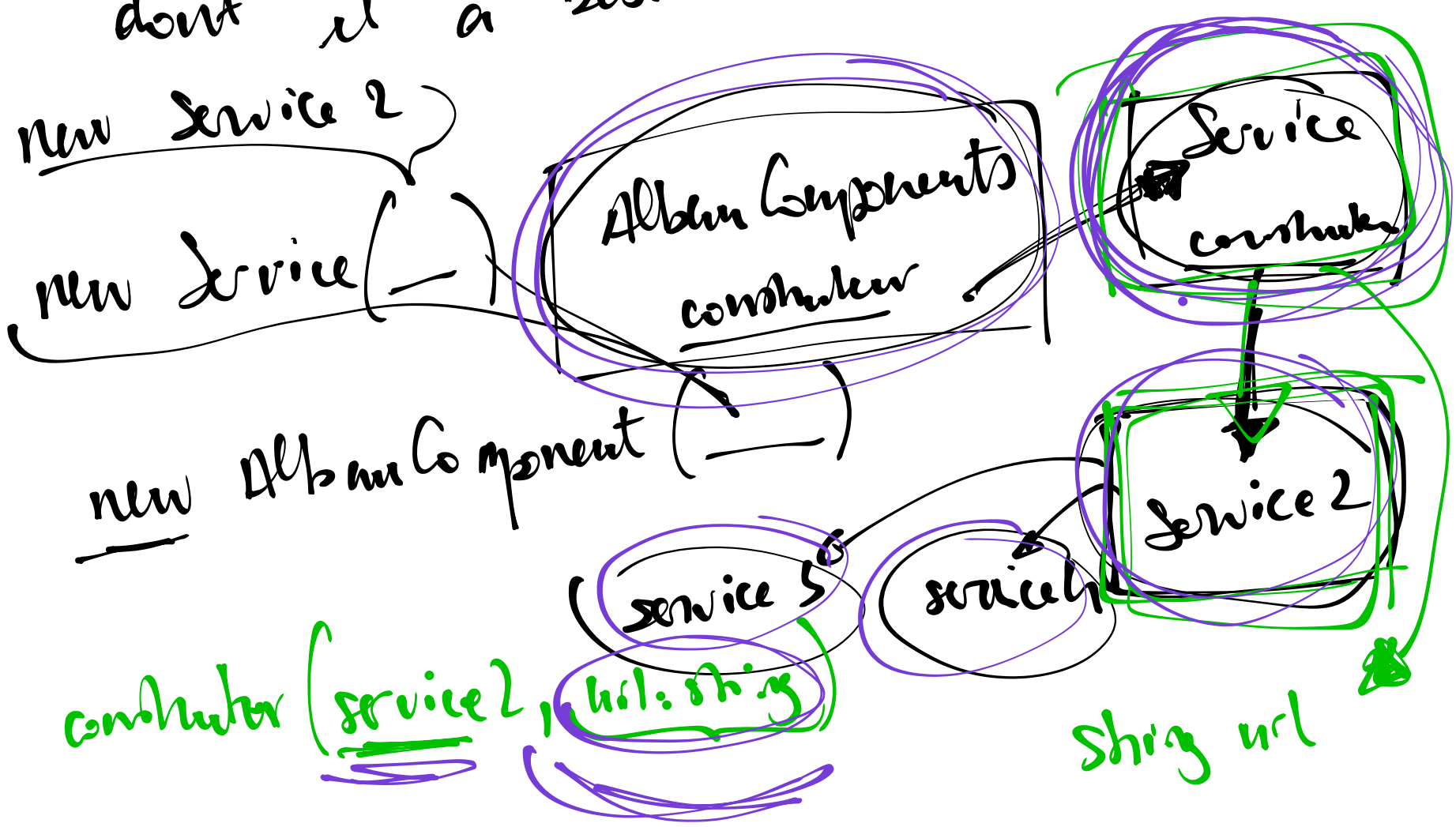
Property Binding : Permet de lier à un attribut d'un élément une expression Javascript :

@Input : Permet d'indiquer qu'une propriété d'un composant est accessible depuis l'extérieur du composant via les attributs.

@Output : Permet d'indiquer qu'une propriété de type EventEmitter pours être écoutée depuis l'extérieur du composant

Service : En principe les composants ne sont chargés que de la gestion de l'affichage et des interactions.
Un service est une classe qui va contenir de la logique métissable.
On utilise l'**injection de dépendance** pour obtenir un service au sein d'un composant ou d'un autre service

Injection de Dépendance : Un objet qui en utilise un autre ne doit pas le construire lui-même. Il doit le demander dans son constructeur. C'est celui qui a créé l'objet qui doit lui passer l'objet dont il a besoin.



Directive : C'est un code qui va se greffer à un élément HTML pour enrichir / modifier son apparence ou son comportement.

- **Directive d'attribut** : attribut qui donne un comportement particulier à un élément.
- **Directive structurelle** : attribut qui a un effet sur l'existence même d'un élément
- **Composant** : permet d'injecter du code HTML, CSS et de composant passés.

Template Driven Forms : Ce sont de formulaires créés et configurés majoritairement dans le template. Le TS n'intervient qu'au moment de gérer la soumission.

(Forms Module) (ngSubmit)

- NgForm : se greffe au formulaire et lui apporte des ajts/comportements supplémentaire
- Ng Model : idem mais sur un champ particulier.

Reactive Forms: un formulaire créé et configuré directement dans la classe TS: on a accès au formulaire à tout moment: plus de validation

ReactiveFormsModule (ngSubmit)

- FormGroup: fait le lien entre un form et la configuration du formulaire dans le TS.
- FormControlName: idem pour les champs.

Routage: Permet de faire l'association entre une URL et une action de notre app. Dans Angular, on associe une URL et un composant à afficher.

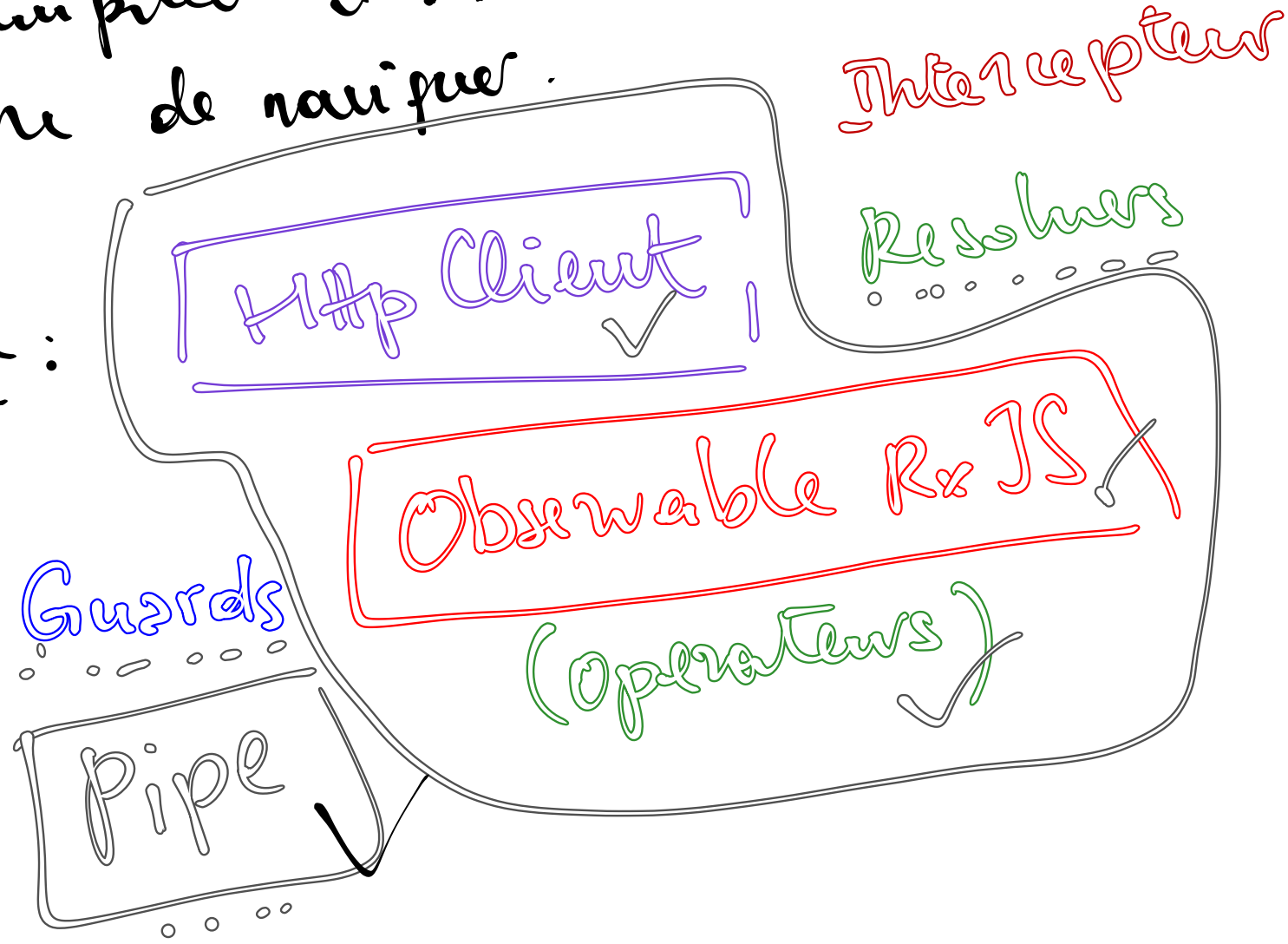
RouterModule

- routerLink: Permet de naviguer (comme href) sans recharger la page.
- router-outlet: Emplacement dans lequel Angular affichera le composant qui correspond à l'URL.

- **ActivatedRoute** : C'est un service qui permet d'étudier / de travailler sur l'URL actuelle.
  - **Snapshot** : image figée à un instant T de la route. Ne tient pas compte des futures évolutions de l'URL.

- **Router** : C'est un service qui permet de manipuler l'URL du navigateur et donc de naviguer.

A Suivre :



Requêtes HTTP. C'est une demande envoyée à un serveur distant. Elle possède :

- Verbe / méthode : GET, POST, DELETE, PUT, PATCH.
- URL : l'adresse de la ressource que l'on veut atteindre.
- Body (corps) : C'est les données qu'on envoi dans la requête
- Headers (En têtes) : informations supplémentaires.

HttpClient
HttpClientModule

HttpClient : C'est un service qui dispose de méthodes qui ont les mêmes noms que les verbes HTTP et qui permet de faire des requêtes HTTP.
Le résultat d'un appel est rendu sous forme d'un Observable

Observable : Un observable est un objet qui va émettre une ou plusieurs valeurs à des moments indéfinis.

Observable (suite) : On peut "l'observer" et faire une action lorsqu'une valeur est émise.

design pattern Observer

Pour insuire une réaction, il faut **Souscrire** auprès de l'Observable en lui donnant un objet qui possèdera :

- next : la fonction à éxécuter lorsqu'une valeur sera émise.
- error : la fonction à éxécuter si une erreur se produit
- Complete : la fonction à éxécuter lorsque l'observable n'a plus de valeurs.

Librairie RxJS : C'est la librairie utilisée par Angular pour la gestion de **l'asynchrone**. C'est une librairie qui existe aussi dans d'autres langages. Elle s'appelle en réalité **ReactiveX**. Elle donne un cadre à la programmation réactive.

Pipe (RxJS): Le pipe permet d'intervenir sur la valeur émise par un Observable avant que la souscription ne la reçoive. Pour intervenir, on utilise des operateurs qui sont fournis par la librairie.

Operateur RxJS: C'est une fonction qui permet d'agir sur une valeur émise par un Observable avant la souscription et qui se place donc dans le Pipe.

- map: permet de transformer la valeur émise.
- filter: permet de stoper la valeur émise en fonction d'un critère.
- SwitchMap: remplace la valeur par une valeur qui se trouve dans un observable.

Pipe (angular) : C'est un outil que l'on peut utiliser dans un template HTML afin d'appliquer une transformation sur une valeur avant de l'afficher :

- uppercase
- lowercase
- currency
- date
- async