



Interaction entre composants parent et fils

Plan



Composant Parent – Composant Fils

Propriété d'entrée « input property »

Propriété de sortie « output property »

Exemple d'application

Accès du composant parent aux éléments du composant fils



Composant Parent – Composant Fils

(1/2)



Un composant peut appeler, au niveau de son template, un autre composant via son selector.

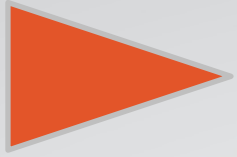
```
@Component({  
  selector: 'app-child',  
  templateUrl: './child.component.html',  
  styleUrls: ['./child.component.css']  
})  
export class ChildComponent {  
  .....  
}
```

Composant fils

Template

```
< app-child> </ app-child>
```

Composant parent



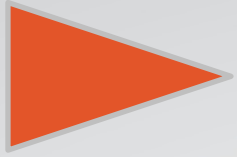
Composant Parent – Composant Fils

(2/2)



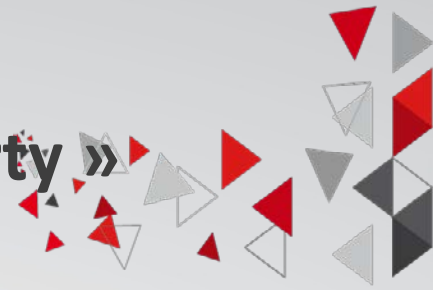
Un composant parent peut interagir avec son composant Fils via:

1. Des propriétés d'entrées et des propriétés de sorties appelées respectivement « input property » et « output property ».
2. Variable locale
3. Le décorateur `@ViewChild()`
4. Des services



Propriété d'entrée « input property »

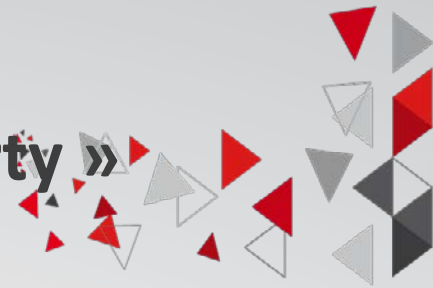
(1/5)



- Un composant fils peut recevoir des informations depuis son composant parent, via une propriété d'entrée « @Input() » .
- Un composant fils peut avoir plusieurs propriétés d'entrées.
- La valeur de la propriété d'entrée est celle envoyée par le parent.
- Pour agir sur la valeur de la propriété d'entrée, le composant fils intercepte la valeur envoyée et effectue le changement désiré à travers le **setter** de cette propriété.



Propriété d'entrée « input property » (2/5)



- La déclaration de la propriété d'entrée se fait au niveau du composant fils comme suit:

```
class childComponent{  
  @Input() property1: Type;  
}
```

- **Input** est à importer depuis **@angular/core**

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

Propriété d'entrée « input property » (3/5)

Le composant parent envoie les informations vers le composant fils à travers « property binding »

```
<app-child [property1]="currentEtudiant"></app-child>
```

currentEtudiant: la
valeur à passer depuis le
composant parent vers
le composant fils à
travers property1



Propriété d'entrée « input property » (4/5)



Le composant fils modifie la valeur de la propriété d'entrée via le setter de cette propriété. Le setter permet d'intercepter et modifier la valeur de la propriété d'entrée.

```
export class childComponent {  
  private prop1 = "";  
  @Input()  
  set property1(p: string) {  
    this.prop1 = p.trim();  
  }  
  get prop1(): string { return this.prop1; }  
}
```

```
{{ prop1 }}  
{{ property1 }}
```




Propriété d'entrée « input property » (5/5)



La méthode hook **ngOnChanges** est lancée automatiquement à chaque fois où la valeur de la propriété d'entrée est modifiée au niveau du composant parent. Le composant fils peut tracer les changements de la valeur de la propriété d'entrée via cette méthode.

```
import {Input, SimpleChange, OnChanges } from '@angular/core';  
...  
export class childComponent implements OnChanges {  
  ....  
  ngOnChanges(changes: {[propKey: string]: SimpleChange}) {  
    Console.log (changes);  
    .....  
  }  
}
```



Propriété de sortie « output property » (1/3)

- Le composant fils expose des émetteurs d'évènements «EventEmitter» dans certaines situations vers le composant parent.
- Le composant parent reste à l'écoute de son composant fils et interagit avec en cas d'un EventEmitter lancé.
- La propriété EventEmitter du fils est appelé « output property » et décrite par « @Output() »
- La propriété de sortie et EventEmitter à importer depuis **@angular/core**

```
import { Output, EventEmitter } from '@angular/core';
```

Propriété de sortie « output property » (2/3)

La déclaration de la propriété de sortie « output property », se fait au niveau du composant fils comme suit:

Type: type d'informations envoyées vers le composant parent quand l'événement est emis

```
class childComponent{  
  @Output() requested = new EventEmitter<Type>();  
  
  oneRequest(){  
    //traitement  
    this.requested.emit(variable);  
    //traitement  
  }  
}
```

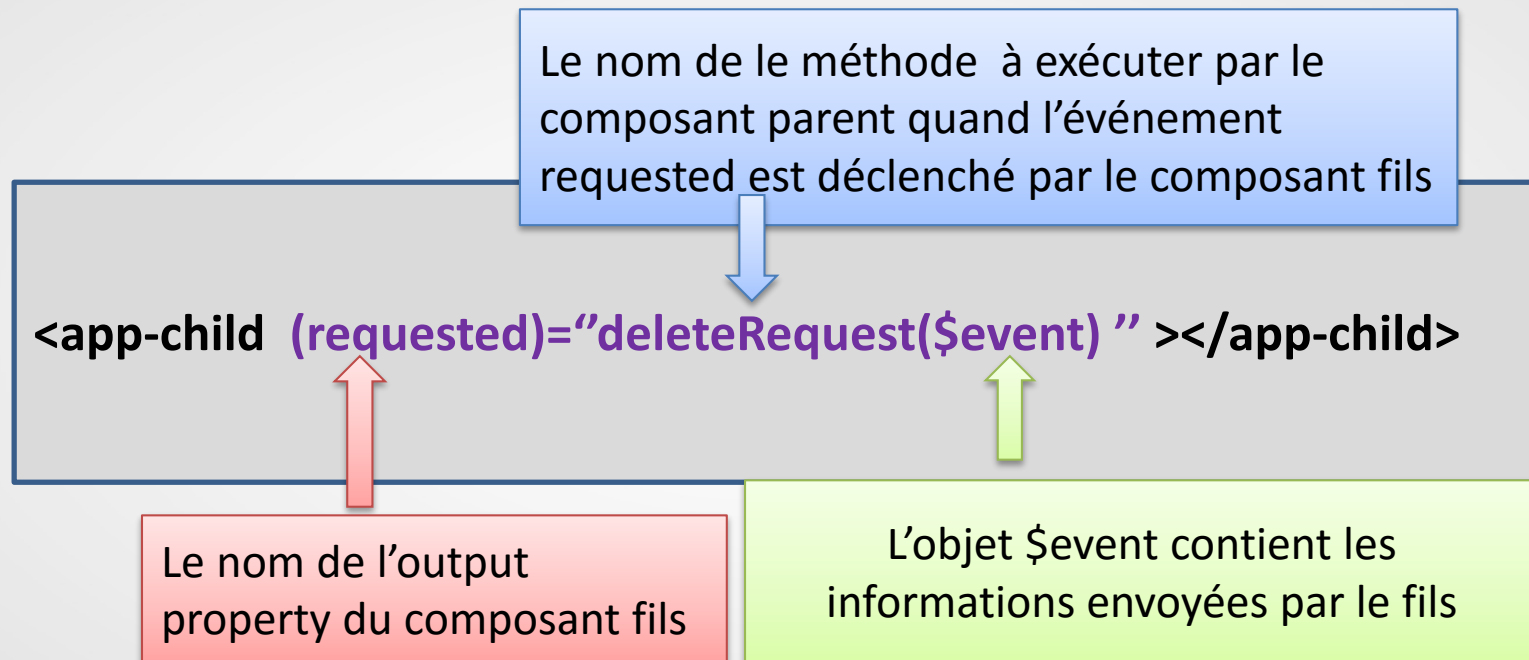
Déclaration de la propriété de sortie

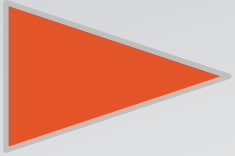
Emission de la propriété de sortie quand la méthode oneRequest est exécutée

variable: l'information envoyée vers le composant parent ayant le type déclaré lors de la déclaration de l'output property

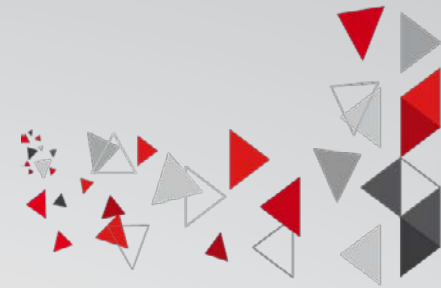
Propriété de sortie « output property » (3/3)

Le composant parent reste à l'écoute des événements émis par le composant fils.





Exemple d'application (1/3)



- 1 Soit un composant parent appelé **homeComponent** qui appelle un composant fils appelé **rePrefixComponent** `< app-re-prefix> </ app-re-prefix>`
- 2 Le composant parent envoie vers le composant fils la valeur à laquelle le préfixe « re » sera rajouté

re-prefix.component.ts

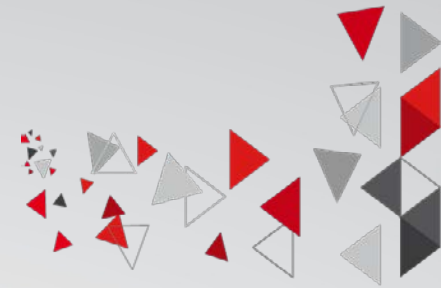
```
@Input() initialValue : string;
```

home.component.html

```
Valeur: <input type="text" [(ngModel)]="firstVal" ngModel>  
<app-re-prefix [initialValue]="firstVal"></app-re-prefix>
```



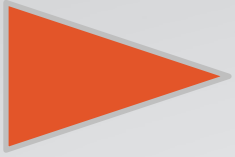
Exemple d'application (2/3)



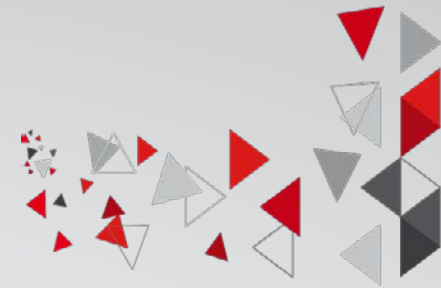
- 3 Le composant fils intercepte la valeur reçue et lui ajoute le préfixe « re » et 4 renvoie la nouvelle valeur vers le composant parent.

re-prefix.component.ts

```
private prefixedValue = "";  
@Output() prefixed = new EventEmitter<string>();  
  
@Input()  
set initialValue(val:string){  
  this.prefixedValue="re"+val;  
  
this.prefixed.emit(this.prefixedValue);  
}
```



Exemple d'application (3/3)



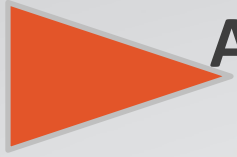
- 5 Une fois le composant parent reçoit la nouvelle valeur, il l'affiche.

home.component.ts

```
newVal:string="";  
getNewVal(val:string){  
    this.newVal=val;  
}
```

home.component.html

```
<app-re-prefix [initialValue]="firstVal"  
(prefixed)="getNewVal($event)"></app-re-prefix>  
{{newVal}}
```



Accès du composant parent aux éléments du composant fils (1/6)



- Le composant parent n'a pas le droit d'accéder aux propriétés et méthodes de son composant fils.
- Mais, cette restriction peut être résolue moyennant deux méthodes :
 1. En utilisant les variables références de template (accès limité au niveau du template)
 2. En utilisant le décorateur `@ViewChild()` (accès à partir de la classe du composant)

Accès du composant parent aux éléments du composant fils (2/6)

Pour accéder aux éléments du composant fils depuis le composant parent via les variables références de template, il faut créer une variable de template qui pointe sur le composant fils.

```
<app-child #child></app-child>
```


Au niveau du template du composant parent

```
<button  
(click)="child.start()">Start</button>  
<div>{{child.propertyA}}</div>
```


```
<app-child #child></app-child>
```

Au niveau du composant Fils

```
Class childComponent{  
  start(){.....}  
  propertyA : number = 10;  
  
}
```



Accès du composant parent aux éléments du composant fils (3/6)



Quand le composant parent veut lire ou écrire dans l'une des propriétés de son composant fils ou bien exécuter une action de son composant fils, ce dernier doit être injecté dans le composant parent avec ViewChild().

```
import { childComponent } from './child.component';  
@ViewChild(childComponent)  
private exempleComponent : ChildComponent;
```

Accès du composant parent aux éléments du composant fils (4/6)

Au niveau du composant Fils


```
Class childComponent{  
start(){.....}  
propertyA : number = 10;  
}
```

Au niveau du template parent

```
<button (click)="startp()">  
calculer</button>  
{{val}}
```

Au niveau de la classe du composant parent

```
import { childComponent } from './child.component';  
...  
Class parentComponent{  
...  
val:number;  
@ViewChild(childComponent)  
private childComponent : childComponent;  
startp(){  
this.childComponent.start();  
this.val=this.childComponent.propertyA;  
}  
...}
```



Accès du composant parent aux éléments du composant fils (5/6)

- ViewChild() à importer depuis @angular/core.

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

- Le composant parent accède aux méthodes/propriétés du composant fils à partir de la classe du composant.
- Le composant fils doit être appelé au niveau du template du composant parent.
=> Le composant fils devient disponible quand la vue du parent est initialisée.

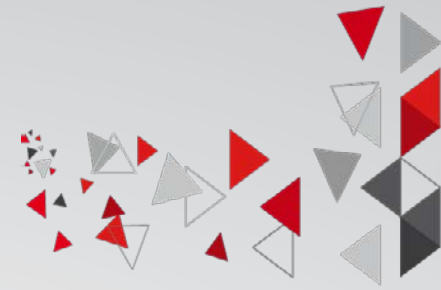
Accès du composant parent aux éléments du composant fils (6/6)

- Pour accéder au composant fils depuis le composant parent une fois la vue de ce dernier est initialisée, il faut s'assurer que la vue du parent a été bien initialisée.
- Pour ce faire, le composant parent doit implémenter l'interface **AfterViewInit** et la méthode **ngAfterViewInit**.

```
export class parentComponent implements AfterViewInit {  
  .....  
  ngAfterViewInit() {  
    .....  
  };  
}
```



Références



<https://angular.io/>
Septembre 2020)

(dernière

consultation