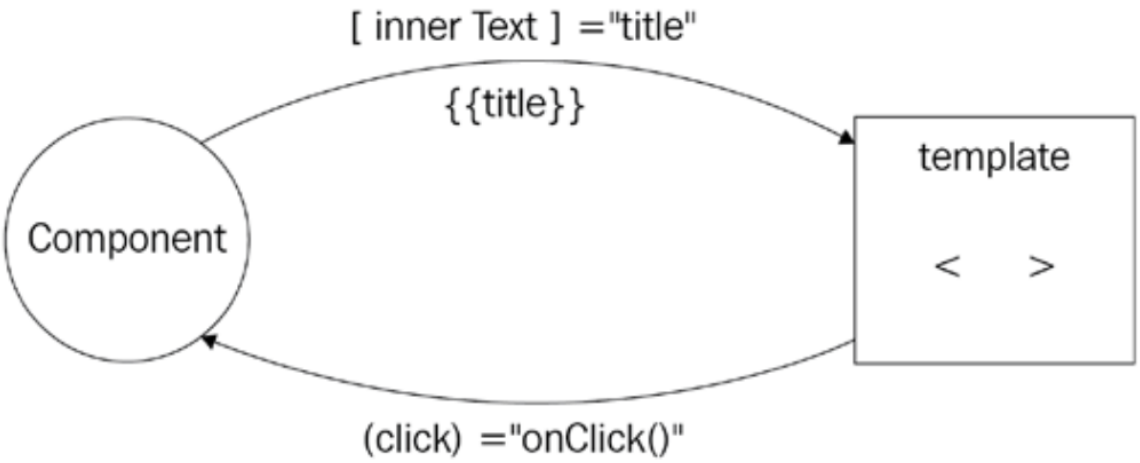
@Rizwan Mushtaq Dhudhaal 👨‍💻

#angular: Communicate between component.ts and template files

There are 2 approaches to communicate between *.ts file and *.html file:

1. One way binding
2. Two way binding

<u>One way binding:</u>



property binding

```
<span>{{ title }}</span>
<span [innerText]="title"></span>
```

- Above line of code is from an html file also called as template file in angular.
- “title” is a property name defined in the class component.ts file.
- There are 2 ways we can use “title” property in template file:
  - {{ title }} -> interpolation
  - [innerText]="title" -> property binding

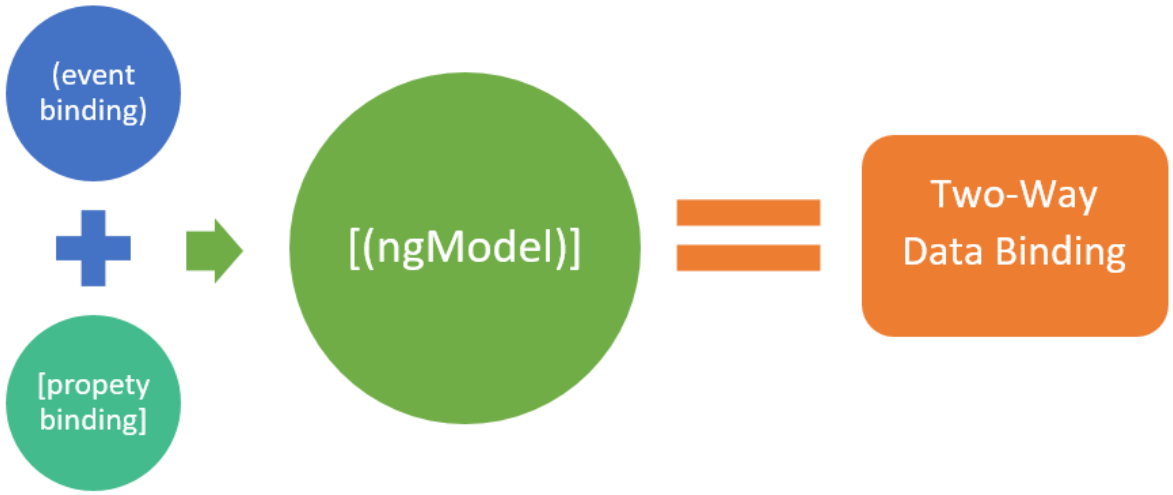
Event binding

```
<button (click)="onClick()">Click me</button>
```

- If I want to communicate from the html file back to the component.ts file, it could happen e.g when the user clicks some button in the template file. We use the above approach called event binding.

- On the left side, there is an event inside the parentheses
- On the right hand side, there is the onClick public method defined in the component.ts file.

Two way binding:



- I would like to explain it using username and password inputs in template file and linking these inputs with properties in *.ts file.

*.ts file:

```
export class LoginComponent {
        username: string;
        password: string;
}
```

*.html file:

```
<div>
        <input type="text" name="username" placeholder="Username"
        [(ngModel)]="username">
</div>
<div>
        <input type="password" name="password" placeholder="Password"
        [(ngModel)]="password">
</div>
```

- In above code, we used ngModel directive to bind public property in *.ts file to template file with input element.

- Now to see effect of two-way binding, I will use interpolation to see the value of username:
   {{ username }}
- As you change the value of the input element, the username property is also updated. This is magic, which is called as two way binding in angular.
- Just to remember:
  The syntax of the ngModel directive is known as a banana in a box, which is a memory rule for you to be able to remember how to type it. We create it in two steps. First, we create the banana by surrounding ngModel in parenthesis (). Then, we put the banana in a box by surrounding it in square brackets [()]. Remember, it's called banana in a box, not box in a banana.