# Wednesday: Submit Form

## Submit Form

### Submitting forms using ngSubmit

At the bottom of our form, we have a submit button that's supposed to add a new goal to our array but if we click it right now, nothing happens. Let's make it work the way it is supposed to. We want it to add a goal to the goals array when this button is clicked.

To do this, we need to add use the `ngSubmit` directive from angular forms module that we imported earlier. Let's add it to our form:

*src/app/goal-form/goal-form.component.html*

```
<div class="container-fluid">
  <h2 class="text-center">Create a new Goal</h2>
  <hr>
  <form (ngSubmit)='submitGoal()' #goalForm='ngForm'>
....
```

We have added the `(ngSubmit)` event attribute which calls the `submitGoal()` function once the event is emitted. Let's now write the code to be executed when the `submitGoal()` function is called.

*src/app/goal-form/goal-form.component.ts*

```
import { Component, OnInit, Output, EventEmitter } from'@angular/core';
...
exportclass GoalFormComponent implements OnInit {

  newGoal = new Goal(0,"","",newDate());
@Output() addGoal = new EventEmitter<Goal>();

  submitGoal(){
this.addGoal.emit(this.newGoal);
  }
...
```

At the top, we have imported the `Output` and `EventEmitter` functions from `@angular/core`. We have then created the `addGoal` event emitter object which is of type `Goal` that will emit an event to the parent component. We have then created the `submitGoal()` function in which we use the emit method and pass in the new goal object we want to create.

Since the `addGoal` event is being emitted to a parent component, we need to make sure the parent component is informed of this event being emitted. We want the goal component to be the parent component in this case, so let's write the code that will handle this event.

*src/app/goal/goal.component.html*

```
...
<div class="col-md-6">
      <app-goal-form (addGoal)="addNewGoal($event)"></app-goal-form>
 </div>
...
```

In the template, we catch the `addGoal` event being emitted and define that it should call the `addNewGoal()` function which takes in an event placeholder. Let's now define this function that is supposed to be executed by our parent component.

*src/app/goal/goal.component.ts*

```
....

export class GoalComponent implements OnInit {

  goals: Goal[] = [
    new Goal(1, 'Watch finding Nemo', 'Find an online version and watch merlin find his son',new Da
te(2019,9,14)),
    new Goal(2,'Buy Cookies','I have to buy cookies for the parrot',new Date(2019,6,9)),
    new Goal(3,'Get new Phone Case','Diana has her birthday coming up soon',new Date(2019,1,12)),
    new Goal(4,'Get Dog Food','Pupper likes expensive snacks',new Date(2019,11,18)),
    new Goal(5,'Solve math homework','Damn Math',new Date(2019,2,14)),
    new Goal(6,'Plot my world domination plan','Cause I am an evil overlord',new Date(2019,3,14)),
  ];
....
  addNewGoal(goal){
    let goalLength = this.goals.length;
    goal.id = goalLength+1;
    goal.completeDate = new Date(goal.completeDate)
    this.goals.push(goal)
  }
....
```

We have created the `addNewGoal()` function that takes a goal object as an argument. We first need to change the `id` property of the goal. We get the length of the array of goals and store it in the variable `goalLength` we then add one to the `goalLength` and set that as the new `id` for the goal. We then set the `completeDate` property of the goal object to a `Date` Object. Lastly, we push the new goal to our array of goals.

If our server is still running, we can now add a new goal to the array and it will display on our application.