

Лабораторна робота № 6

Події та директиви.

Виконаємо зміну картинки у шаблоні також за ініціативою користувача. Для цього додаємо метод, що буде виконувати зміну поточного індексу картинки а також збільшимо інтервал його зміни за таймером (рис. 1).

```
35  ngOnInit() {
36      this.curImage = this.images[this.curImageIndex];
37      setInterval( callback: () => {
38          this.changeCurImage( forward: true);
39      }, ms: 10000);
40
41
42      setTimeout( handler: () => {
43          this.faculty = '';
44          this.specialty = '';
45      }, timeout: 2000);
46
47
48      changeCurImage(forward: boolean) {
49          if (forward) {
50              this.curImageIndex++;
51          } else {
52              this.curImageIndex--;
53          }
54          if (this.curImageIndex >= this.images.length) {
55              this.curImageIndex = 0;
56          }
57          if (this.curImageIndex < 0) {
58              this.curImageIndex = this.images.length - 1;
59          }
60          this.curImage = this.images[this.curImageIndex];
61      }
```

Рис. 1. Додавання методу зміни індексу поточної картини до group.component.ts.

Також додаємо у шаблон 2 кнопки із подією click. Подія виділяється круглими скобками, що сигналізує передачу даних (або подій) від шаблону до typescript класу (рис. 2).

```

1 <div class="group">
2   <div class="img-container">
3     <button (click)="changeCurImage(false)"><< prev</button>
4     <button (click)="changeCurImage(true)">next >></button>
5     <div>
6       <img [src]="curImage">
7     </div>
8   </div>

```

Рис. 2. Додавання кнопок у шаблон group.component.html.

Також додаємо трохи стилізації у group.component.scss (рис. 3).

```

19 .img-container {
20   text-align: center;
21   margin-bottom: 20px;
22 }

```

Рис. 3. Додавання стилізації у group.component.scss.

В результаті отримуємо 2 додаткові кнопки, що надають можливість гортати зображення (рис. 4).

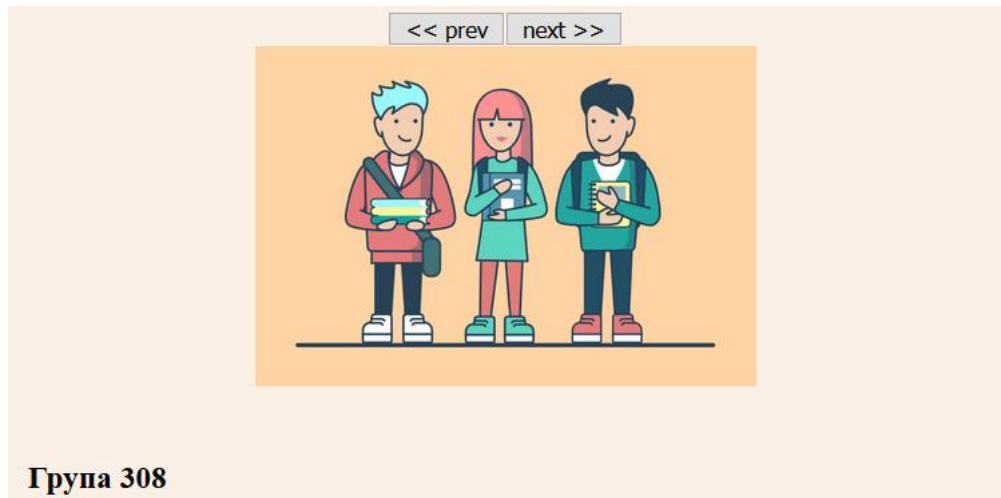


Рис. 4. Зміни на сторінці застосунку.

Реалізуємо також зміну поля номер групи. Для цього поряд із виводом номера групи додаємо поле input, у яке будемо вводити нове значення та кнопку, по натисканню на яку будемо виконувати зміну значення (рис. 5).

```

9      <h3>Група {{ number }}</h3>
10     <div class="edit-number">
11         <input #numb type="text">
12         <button (click)="changeNumber(numb.value)">Змінити</button>
13     </div>
14     <p class="faculty">{{ faculty }}</p>

```

Рис. 5. Додавання елементів до group.component.html.

Також у класі GroupComponent реалізуємо метод, що виконуватиме зміну значення номеру групи (рис. 6).

```

61
62     changeNumber(numb: string) {
63         this.number = numb;
64     }

```

Рис. 6. Метод зміни значення поле номера групи у group.component.ts.

В результаті маємо функціонал зміни значення номеру студентської групи (рис. 7).

Рис. 7. Вигляд сторінки після додавання поля вводу.

У даному випадку також можливо обійтися і без кнопки. Змінимо у шаблоні прив'язку до події input та приберемо кнопку (рис. 8). Тепер зміна відбуватиметься при введенні кожного нового символу у поле вводу.

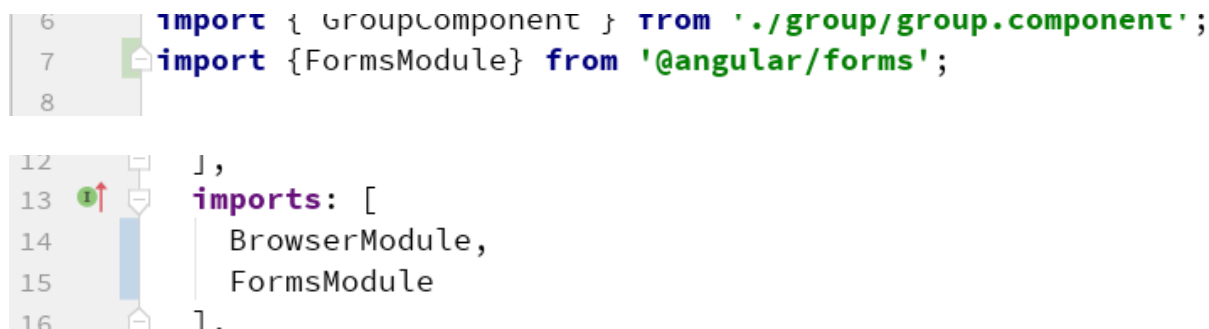
```

9      <h3>Група {{ number }}</h3>
10     <div class="edit-number">
11         <input #numb type="text" (input)="changeNumber(numb.value)">
12     <!-- <button (click)="changeNumber(numb.value)">Змінити</button> -->
13     </div>

```

Рис. 8. Зміна події для реагування на зміни значення.

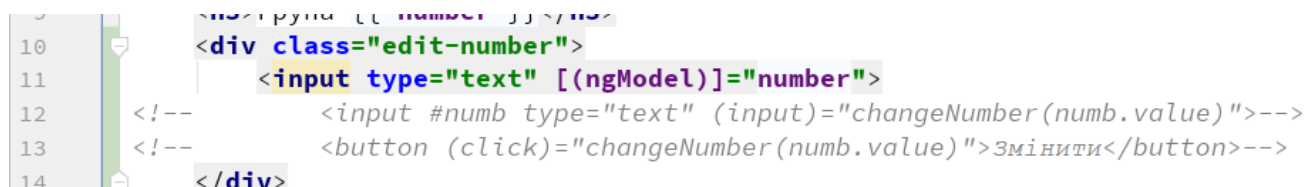
Але в Angular можна реалізувати дану задачу ще простіше – за допомогою механізму двостороннього зв’язування, коли зміни у шаблоні автоматично застосовуються для полів класу, і навпаки, при зміні полів класу автоматично оновляться значення елементів шаблону. Для цього нам перш за все необхідно в `app.module.ts` імпортувати `FormsModule` (рис. 9).



```
6 import { GroupComponent } from '../group/group.component';
7 import { FormsModule } from '@angular/forms';
8
12 ],
13 imports: [
14   BrowserModule,
15   FormsModule
16 ]
```

Рис. 9. Підключення `FormsModule`.

Далі у шаблоні `group.component.html` виконаємо двостороннє зв’язування для поля вводу значення номеру групи (рис. 10).



```
10 <div class="edit-number">
11   <input type="text" [(ngModel)]="number">
12   <!-- <input #numb type="text" (input)="changeNumber(numb.value)">-->
13   <!-- <button (click)="changeNumber(numb.value)">Змінити</button>-->
14 </div>
```

Рис. 10. Використання `[(ngModel)]`.

Тепер навіть можна прибрати із класу `GroupComponent` метод `changeNumber`, оскільки Angular візьме на себе всю роботу по зв’язуванню елементу `input` та змінної `number`.

Тепер трохи познайомимось із директивами. Реалізуємо умовне форматування номера групи, змінимо колір та розмір шрифту для груп 3-го та більш старших курсів. Спочатку додаємо відповідну стилізацію до `group.component.scss` (рис. 11).

```

5  h3 {
6    margin-top: 0px;
7    &.old {
8      color: darkblue;
9      font-size: 1.5em;
10   }
11 }

```

Рис. 11. Зміни у group.component.scss.

Далі у класі GroupComponent реалізуємо метод, що визначатиме необхідність стилізації (рис. 12).

```

61
62  isOld() {
63    return (+this.number.toString()[0]) > 2;
64  }

```

Рис. 12. Метод isOld у group.component.ts.

І, у шаблоні group.component.html для поля номер групи за допомогою директиви ngClass додаємо клас стилізації для груп, що належать до 3-го курсу на старше (рис. 13).

```

9  <h3 [ngClass]="{old: isOld()}">Група {{ number }}</h3>
10 <div class="edit-number">

```

Рис. 13. Використання директиви ngClass.

Далі познайомимось із директивою ngIf, що дозволяє виконувати умовне виведення блоків html-коду у шаблонах. Приховаємо у шаблоні все, окрім номера групи, та додамо посилання «детальніше», за яким буде відображатися вся інша інформація. Для цього спочатку додаємо до класу поле-флаг showInfo (рис. 14).

```

11  спеціалізу = інженерія програмного забезпечення ;
12  showInfo = false;
13

```

Рис. 14. Додавання поля до класу GroupComponent.

Та виконуємо зміни у шаблоні group.component.html (рис. 15).

```

1 <div class="group">
2   <div class="img-container" *ngIf="showInfo">
3     <button (click)="changeCurImage(false)"><< prev</button>
4     <button (click)="changeCurImage(true)">next >></button>
5     <div>
6       <img [src]="curImage">
7     </div>
8   </div>
9
10  <h3 [ngClass]="{old: isOld()}">Група {{ number }}</h3>
11
12  <div *ngIf="showInfo; then btnHide else btnShow"></div>
13  <ng-template #btnHide>
14    <button (click)="showInfo=false">Згорнути</button>
15  </ng-template>
16
17  <ng-template #btnShow>
18    <button (click)="showInfo=true">Детальніше</button>
19  </ng-template>
20
21  <div class="info" *ngIf="showInfo">
22    <div class="edit-number">
23      <input type="text" [(ngModel)]="number">
24      <!-- <input #numb type="text" (input)="chang
25      <!-- <button (click)="changeNumber(numb.valu
26
27    </div>
28    <p class="faculty">{{ faculty }}</p>
29    <p class="specialty">спеціальність: {{ specialty }}</p>
30    <p>Кількість студентів: {{studentsQuantity}}</p>
31    <p>Склад групи: {{students}}</p>
32    <p>Староста:</p><pre>{{starosta | json}}</pre>
33  </div>
34</div>

```

Рис. 15. Використання директиви ngIf.

Переглядаємо отриманий результат та перевіряємо роботу кнопки «детальніше» (рис. 16).

Група 308

Детальніше

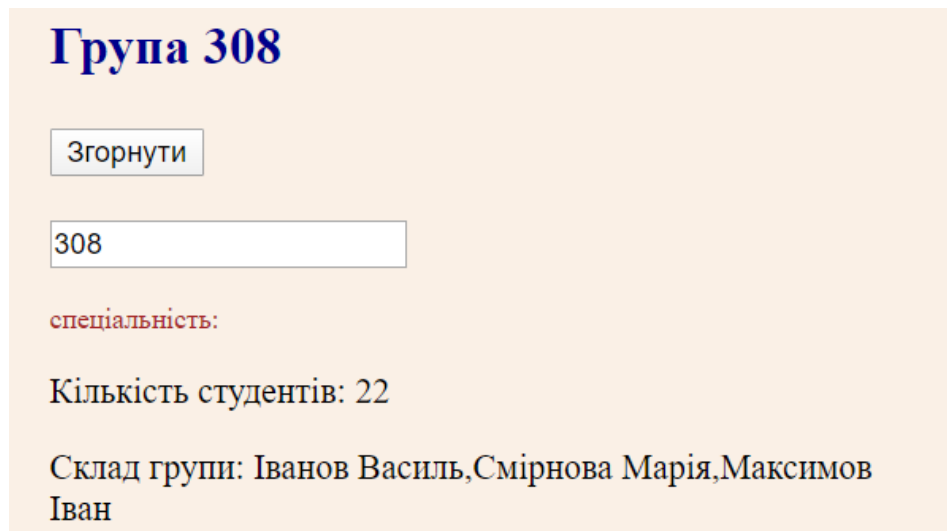


Рис. 16. Зовнішній вигляд сторінки застосунку.

Познайомимось із ще однією директивою, що дозволяє реалізувати циклічний вивід елементів у шаблоні – `ngFor`. Виконаємо визначення масиву студентських груп, та виведемо їх на сторінку. Для початку додаємо створення масиву у `app.component.ts` (рис. 17).

```
8 export class AppComponent {
9   title = 'angular-test-proj';
10  groups = [
11    {
12      number: 301,
13      faculty: `факультет комп'ютерних наук`,
14      specialty: `комп'ютерні науки`
15    },
16    {
17      number: 308,
18      faculty: `факультет комп'ютерних наук`,
19      specialty: `інженерія програмного забезпечення`
20    },
21  ];
22 }
```

Рис. 17. Масив студентських груп.

Використаємо директиву `ngFor` для тегу `app-group` в `app.component.html` (рис. 18).

```
1 <h1>Hello world !!</h1>
2 <app-group *ngFor="let group of groups"></app-group>
```

Рис. 18. Використання ngFor.

Але при перегляді сторінки бачимо, що група 308 виводиться 2 рази, адже ми не передаємо до компоненти group жодних значень із батьківської app, і всі дані беруться із прописаних у класі GroupComponent значень. Переробимо GroupComponent таким чином, щоб він міг приймати вхідні дані та працював із ними. Також приберемо із класу та шаблону компонента group поля студенти та староста, і вивід картинок (рис. 19-20).

```
1  import {Component, Input, OnInit} from '@angular/core';
2
3  @Component({
4    selector: 'app-group',
5    templateUrl: './group.component.html',
6    styleUrls: ['./group.component.scss']
7  })
8  export class GroupComponent implements OnInit {
9    @Input() group;
10
11    showInfo = false;
12
13    constructor() { }
14
15    ngOnInit() {
16    }
17
18    isOld() {
19      return (+this.group.number.toString()[0]) > 2;
20    }
21  }
```

Рис. 19. Зміни у group.component.ts.

```
1  <div class="group">
2    <h3 [ngClass]="{old: isOld()}">Група {{ group.number }}</h3>
3
4    <div *ngIf="showInfo; then btnHide else btnShow"></div>
5    <ng-template #btnHide>
6      <button (click)="showInfo=false">Згорнути</button>
7    </ng-template>
8    <ng-template #btnShow>
9      <button (click)="showInfo=true">Детальніше</button>
10  </ng-template>
```



```

11
12 <div class="info" *ngIf="showInfo">
13   <div class="edit-number">
14     <input type="text" [(ngModel)]="group.number">
15   </div>
16   <p class="faculty">{{ group.faculty }}</p>
17   <p class="specialty">спеціальність: {{ group.specialty }}</p>
18   <p>Кількість студентів: {{group.studentsQuantity}}</p>
19 </div>
20 </div>

```

Рис. 20. Зміни у group.component.html.

Тепер наш компонент GroupComponent вміє приймати вхідні дані. Скористаймося цим, і передамо до нього дані студентських груп із AppComponent (рис. 21).

```

1 <ng>hello world !!</ng>
2 <app-group *ngFor="let group of groups" [group]="group"></app-group>
3

```

Рис. 21. Передача даних у компонент group.

Переглянемо, як змінилась сторінка нашого застосунку (рис. 22).

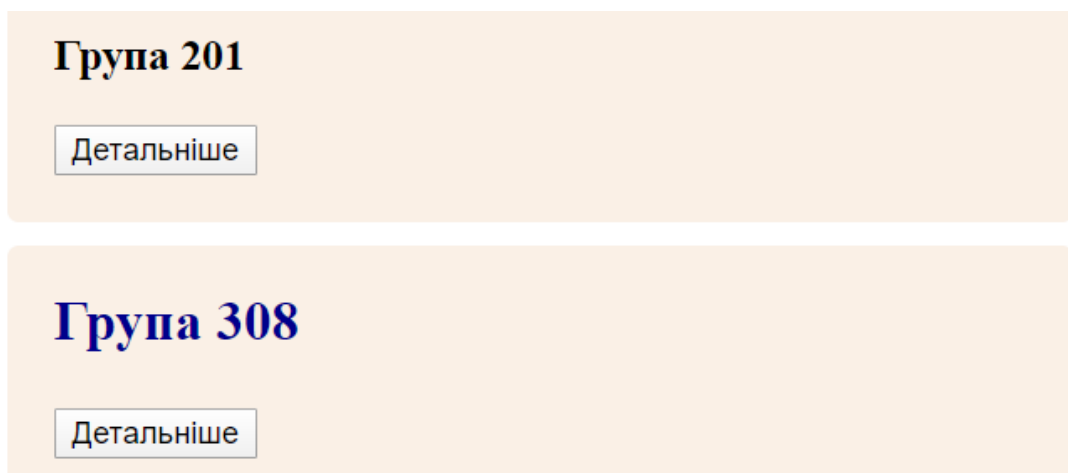


Рис. 22. Перевірка виведення студентських груп.

Дочірні компоненти також можуть не тільки приймати значення від батьківських, а і передавати дані у батьківський компонент. Для цього використовується декоратор @Output та у дочірньому компоненті створюємо обробник подій, який ініціює цю передачу. Так, виконаємо у компоненті app.component виведення дати останньої зміни даних, при тому, що сама зміна буде відбуватися у компоненті group.component.

Спочатку в класі компонента `group` додаємо `@Output` параметр та метод, що ініціює його повернення (рис. 23).

```
1  import {Component, EventEmitter,  
2    Input, OnInit, Output} from '@angular/core';  
  
10  @input() group;  
11  @Output() dataChange = new EventEmitter();  
12  showInfo = false;  
  
23  onChange() {  
24    this.dataChange.emit(new Date());  
25  }
```

Рис. 23. Додавання повернення даних.

У шаблоні `group.component.html` викликаємо повернення даних із компонента при зміні номера групи (рис. 24).

```
13  <div class="edit-number">  
14    <input type="text"  
15      [(ngModel)]="group.number"  
16      (ngModelChange)="onChange()">  
17  </div>
```

Рис. 24. Виклик методу при зміні даних.

У класі `AppComponent` створимо поле для дати останньої зміни та метод, що писатиме туди значення, отримане від компонента `GroupComponent` (рис. 25).

```
9    title = 'angular-test-proj';  
10   changeDate = new Date();  
11   groups = []  
  
25   onDataChange(event) {  
26     this.changeDate = event;  
27   }
```

Рис. 25. Обробка отримання даних.

У шаблоні `app.component.html` виведемо дату останньої зміни та пов'яжемо подію компонента `group` із методом `onDataChange` (рис. 26).

```
1 <h1>Hello world !!</h1>
2 <p>last data change:
3   <span>{{changeDate|date:'dd.MM.yyyy HH:mm:ss'}}</span>
4 </p>

5 <app-group
6   *ngFor="let group of groups"
7   [group]="group"
8   (dataChange)="onDataChange($event)">
9 </app-group>
```

Рис. 26. Зміни у шаблоні app.component.html.

Перевіряємо роботу сторінки.