**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

# «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки Кафедра інформатики та програмної інженерії

Звіт по лабораторній роботі № 1

Робота з компонентами. Взаємодія між компонентами. Прив’язка до подій дочірнього компоненту. Життєвий цикл компоненту.

з дисципліни: «Реактивне програмування»

Студент: Головня Олександр Ростиславович Група: ІП-11 Дата захисту роботи: Викладач: доц. Полупан Юлія Вікторівна Захищено з оцінкою:

Київ, 2024

**Зміст**

[«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО» 1](#_Toc178633737)

[**Вправа 1: Створення додатку з назвою «Components1»** 3](#_Toc178633738)

[**Вправа 2: Стилі та шаблони компонента.** 4](#_Toc178633739)

[**Вправа 3: Підключення зовнішніх файлів.** 4](#_Toc178633740)

[**Вправа 4: Робота з компонентами.** 5](#_Toc178633741)

[**Вправа 5: ng-content.** 7](#_Toc178633742)

[**Вправа 6: Взаємодія між компонентами. Передача даних у дочірній компонент через Input.** 8](#_Toc178633743)

[**Вправа 7: Прив'язка до сетера.** 10](#_Toc178633744)

[**Вправа 8: Прив'язка до подій дочірнього компонента.** 12](#_Toc178633745)

[**Вправа 9: Двостороння прив'язка.** 14](#_Toc178633746)

[**Вправа 10: Життєвий цикл компоненту.** 16](#_Toc178633747)

[**Вправа 11: Реалізація всіх методів.** 18](#_Toc178633748)

[**Висновок:** 20](#_Toc178633749)

[**Список використаних джерел:** 22](#_Toc178633750)

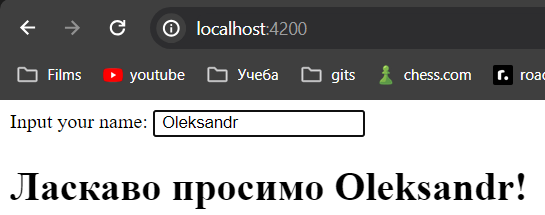
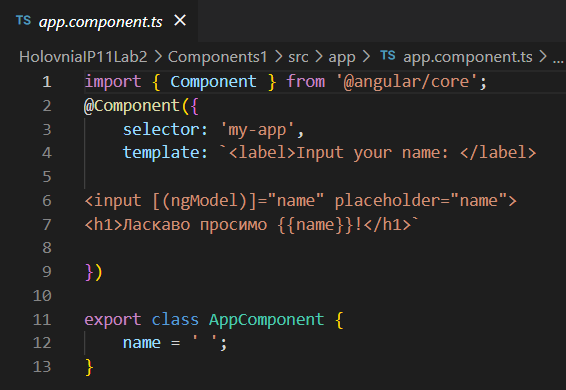
**Мета: Навчитися працювати з компонентами в Angular.**

**Завдання: Створити п’ять Angular-додатків під назвою:**

1. **Components1 (вправи 1-6). Виконати відповідні вправи;**
2. **Components2 (вправи 7-8). Виконати відповідні вправи;**
3. **Components3 (вправа 9). Виконати відповідні вправи;**
4. **Components4 (вправа 10). Виконати відповідні вправи;**
5. **Components5 (вправа 11). Виконати відповідні вправи;**
6. **Зробити звіт по роботі.**
7. **Angular-додатки Components1 та Components5 розгорнути на платформі**
8. **Firebase у проектах з ім’ям «ПрізвищеГрупаLaba2-1» та «ПрізвищеГрупаLaba2-5», наприклад «KovalenkoIP01Laba2-1» та «KovalenkoIP01Laba2-5»**

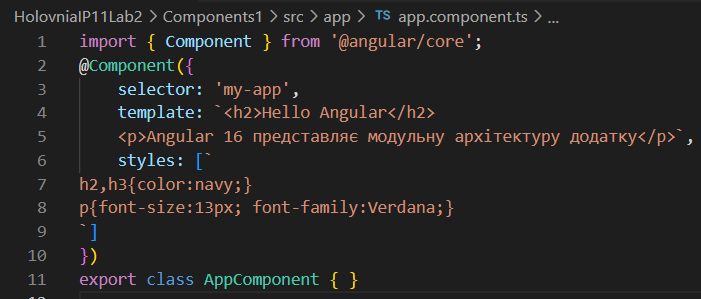
**Вправа 1: Створення додатку з назвою «Components1»**

Із минулого комп’ютерного практикуму запозичимо проект HelloApp:

  
Рисунок 1.1. Застосунок, запозичений з попереднього практикуму.

**Вправа 2: Стилі та шаблони компонента.**

Для стилізації компонента додамо йому атрибут styles:



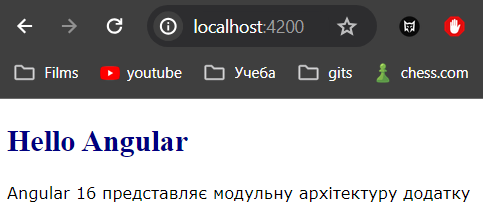
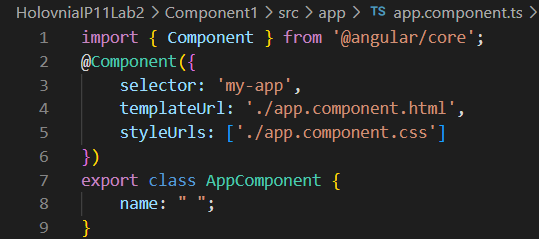
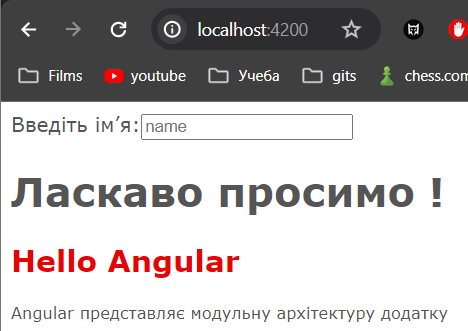
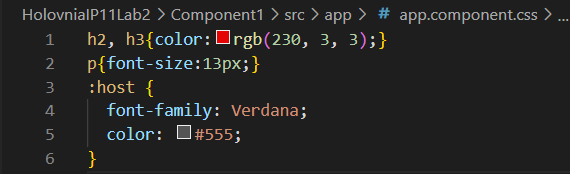
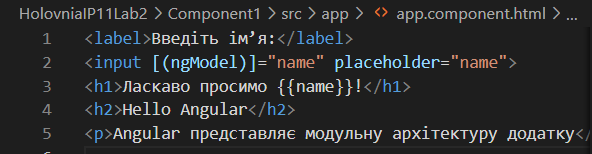


Рисунок 1.2. Приклад додавання стилів до компоненти.

**Вправа 3: Підключення зовнішніх файлів.**

Якщо стилі або шаблон займають доволі багато місця, є сенс їх винести у окремі файли. Таким чином код компоненту стане чистішим і зрозумілішим.

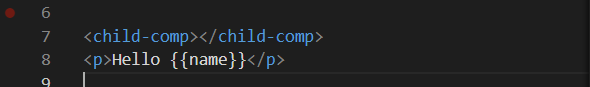


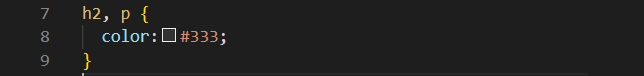
  
Рисунок 1.3.Результат винесення стилів та шаблону в окремі файли.

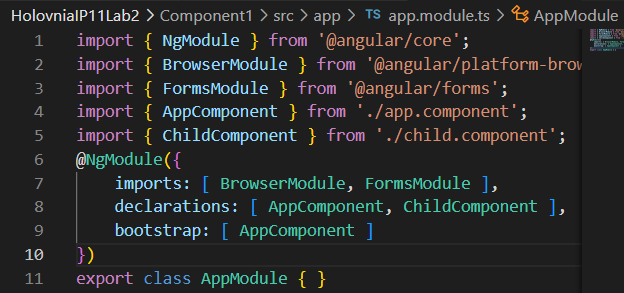
**Вправа 4: Робота з компонентами.**

Окрім основної компоненти, можна також визначити допоміжні. Вони міститимуться всередині основної компоненти, але матимуть власні стилі.

Змінимо код компоненту AppComponent:







Створимо дочірній компонент ChildComponent:

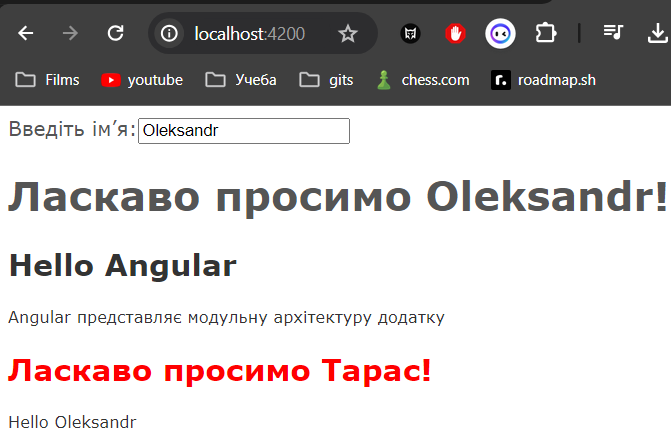


Рисунок 1.4. Використання допоміжних компонент.

Як бачимо, дочірній компонент має власні стилі та простір імен: зовнішня і внутрішня змінні name мають різні значення.

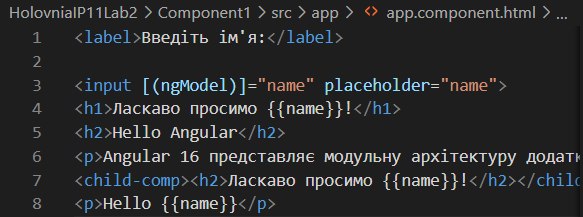
**Вправа 5: ng-content.**

Також є можливість передавати код HTML всередину дочірнього компоненту. Для цього існує елемент ng-content, куди підставляються передані значення.

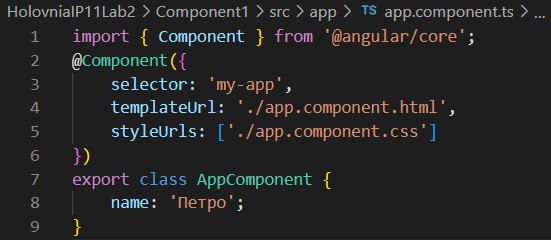
Змінимо компонент ChildComponent:

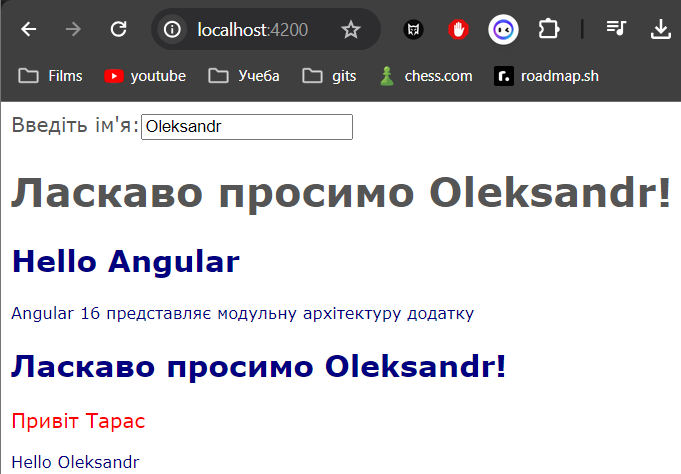


А також внесемо зміни у файли AppComponent:







  
Рисунок 1.5. Приклад передачі у дочірній компонент.

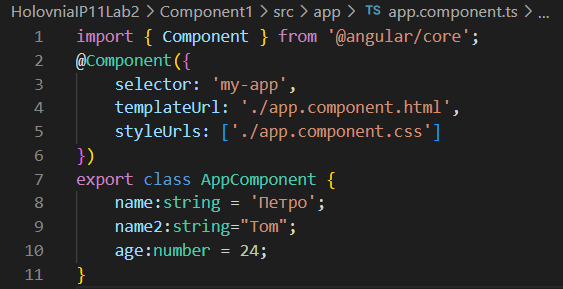
Як бачимо, всередину дочірнього компоненту було передано зовнішнє значення змінної name.

**Вправа 6: Взаємодія між компонентами. Передача даних у дочірній компонент через Input.**

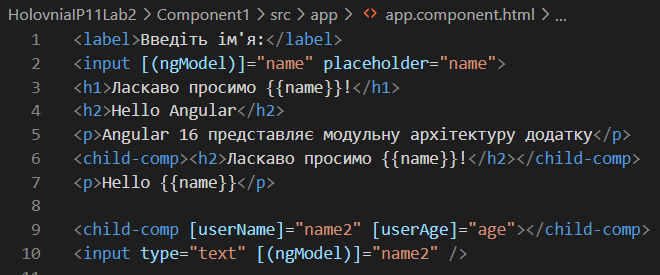
. У компоненті ChildComponent імпортуємо декоратор Input та змінимо код на наступний:



Після чого у головній компоненті визначимо такі моделі, як name, name2 та age:



І у шаблоні передамо ці параметри таким чином:



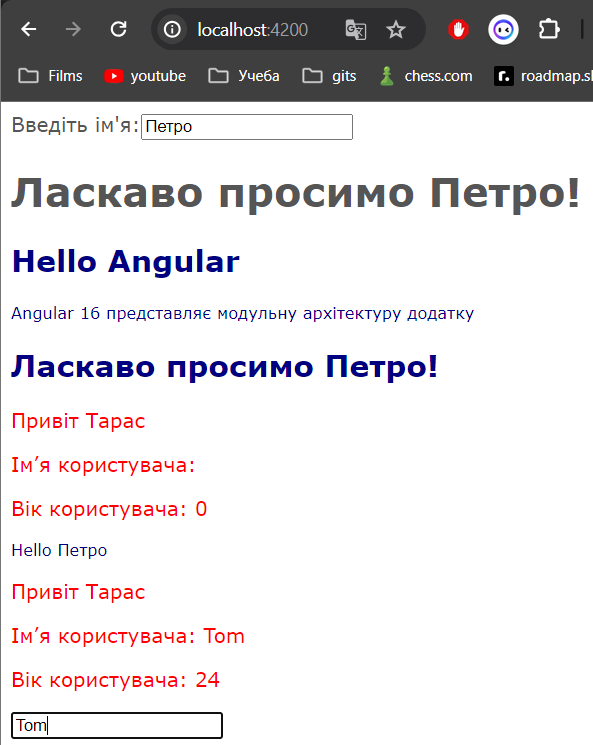
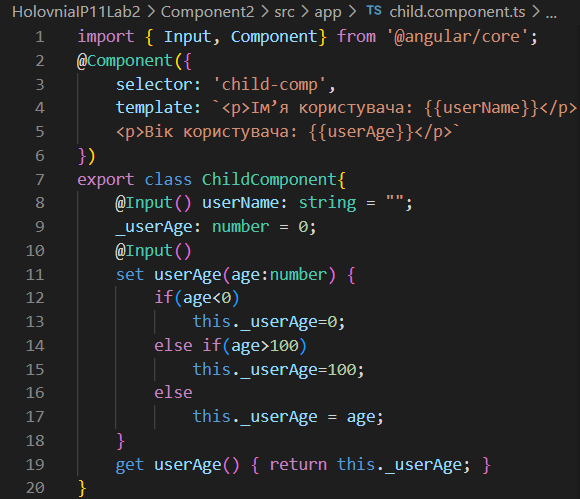


Рисунок 1.6. Результат передачі параметрів через Input.

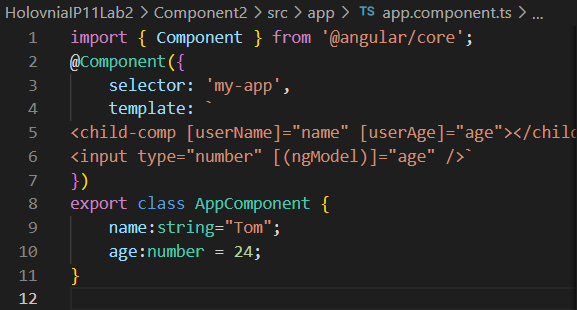
**Вправа 7: Прив'язка до сетера.**

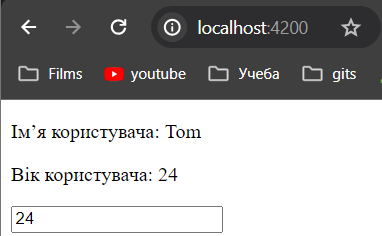
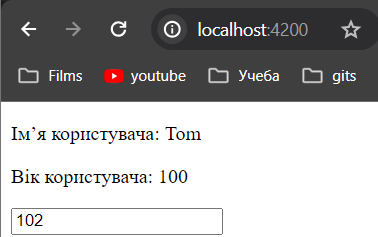
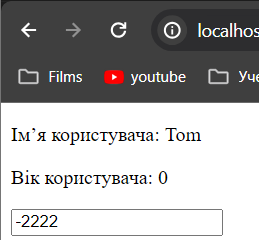
Іноді нам необхідно якось опрацьовувати передані значення. Наприклад, валідувати їх, або якимось чином змінювати. У такому разі можна скористатися прив’язкою до сетера.

Наведемо приклад такої прив’язки у child.component:



Тоді app.component матиме такий вигляд:



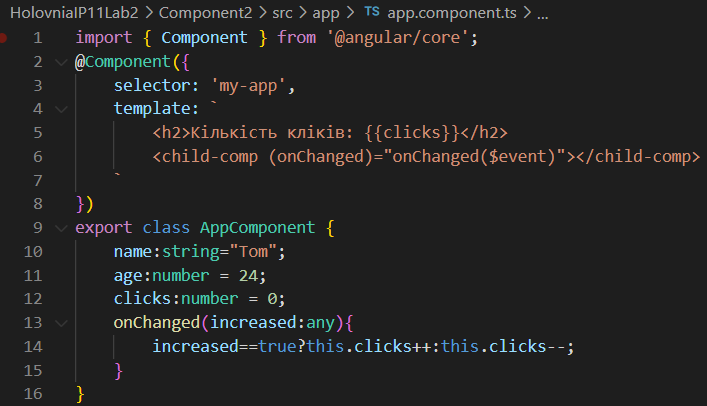
  

Рисунки 1.7-1.9. Приклади прив’язки до сетера.

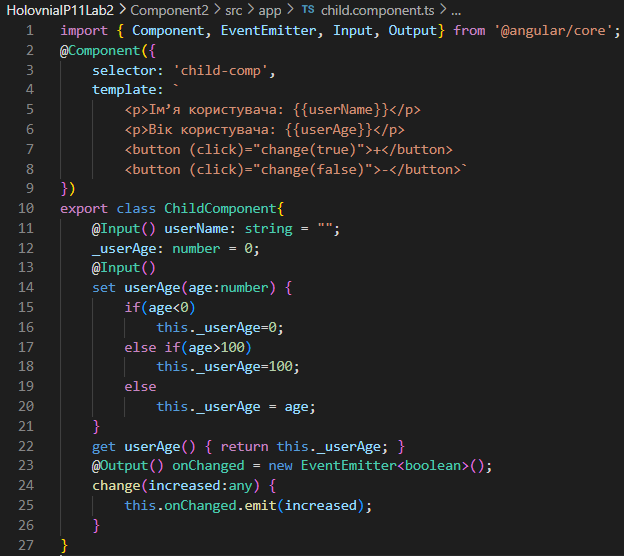
**Вправа 8: Прив'язка до подій дочірнього компонента.**

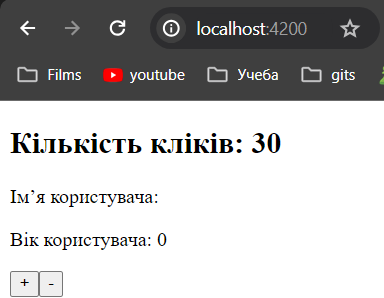
Іноді нам необхідно у основному компоненті реагувати на зміні у дочірньому. Цього можна досягти таким чином:

app.component:



child.component:

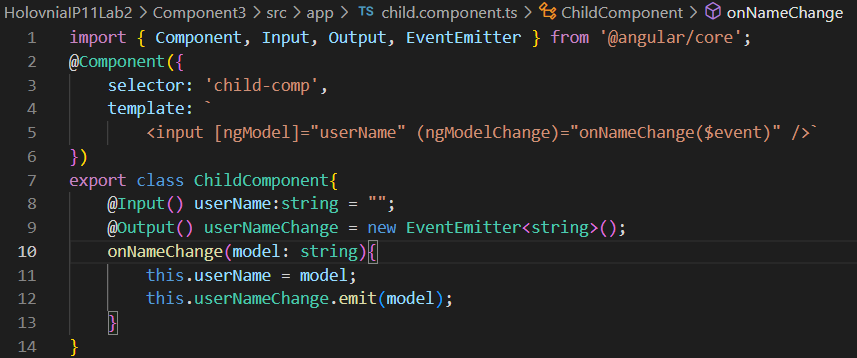
У такому разі при натисканні кнопок дочірнього компоненту відбуватиметься зміна значення моделі основного компоненту:

  
Рисунок 1.10. Приклади прив’язки до подій дочірнього компонента

**Вправа 9: Двостороння прив'язка.**

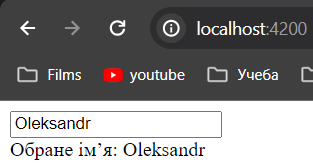
Ми також можемо використовувати двосторонню прив'язку між властивостями головного і дочірнього компонента:

child.component:



app.component:

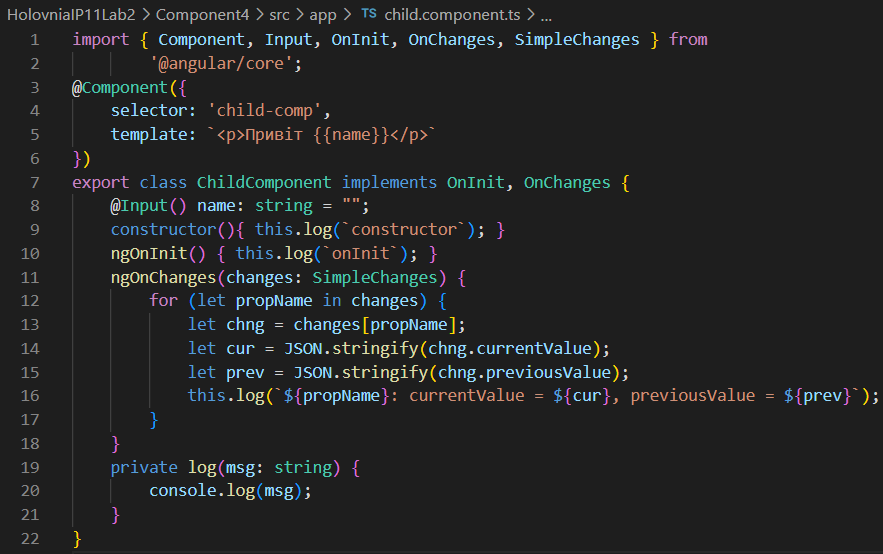


  
Рисунок 1.11. Приклад двосторонньої прив’язки.

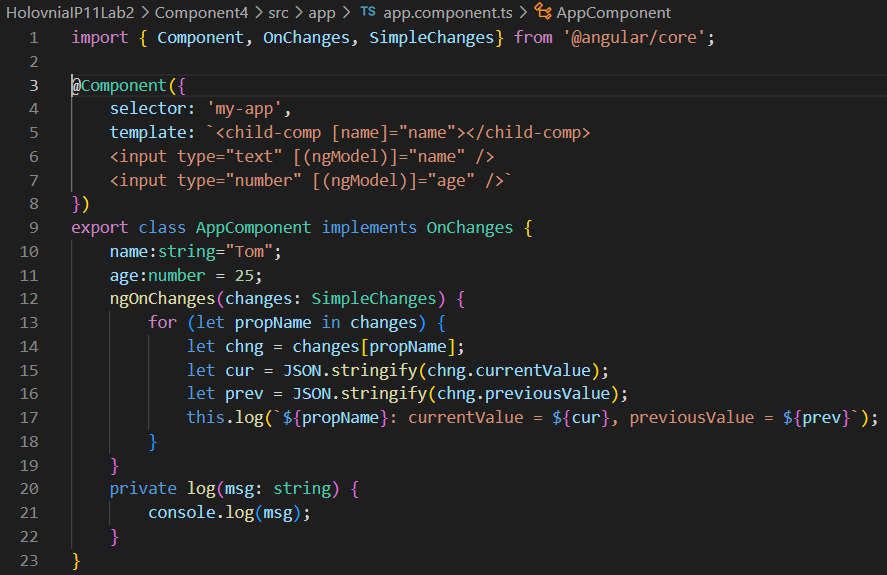
**Вправа 10: Життєвий цикл компоненту.**

Існує також можливість відслідковувати різні етапи життєвого циклу компоненту за допомогою методів ngOnInit, ngOnChanges і так далі.

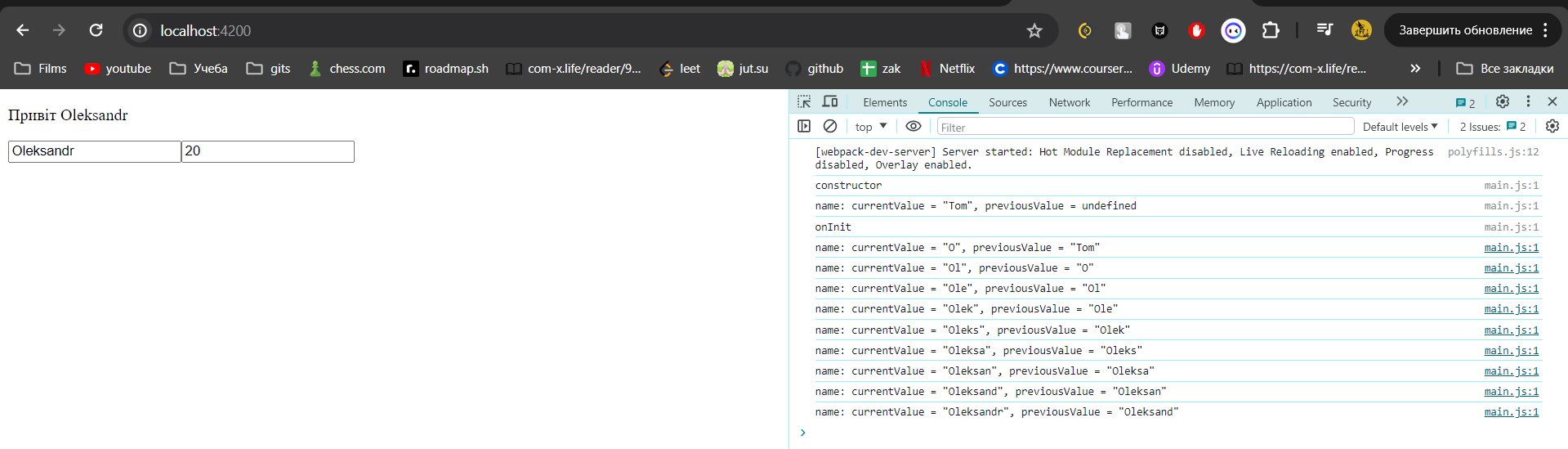
Нехай дочірній компонент виглядає так:



Тоді основний компонент:



Після запуску додатку та зміни значень полів у консолі зможемо відслідкувати порядок запуску тих чи інших методів. Також помітимо, що при зміні значення моделі, яка не використовується дочірнім компонентом, його метод OnChange не викликатиметься.

  
Рисунок 1.12. Життєвий цикл компоненту.

**Вправа 11: Реалізація всіх методів.**

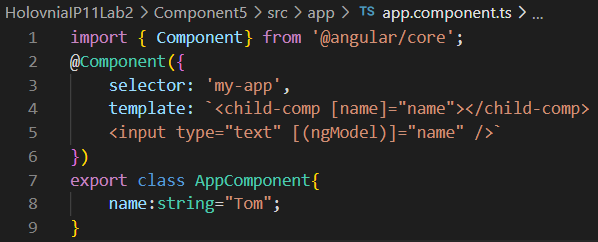
Тепер у новому додатку спробуємо реалізувати усі методи.

child.component:

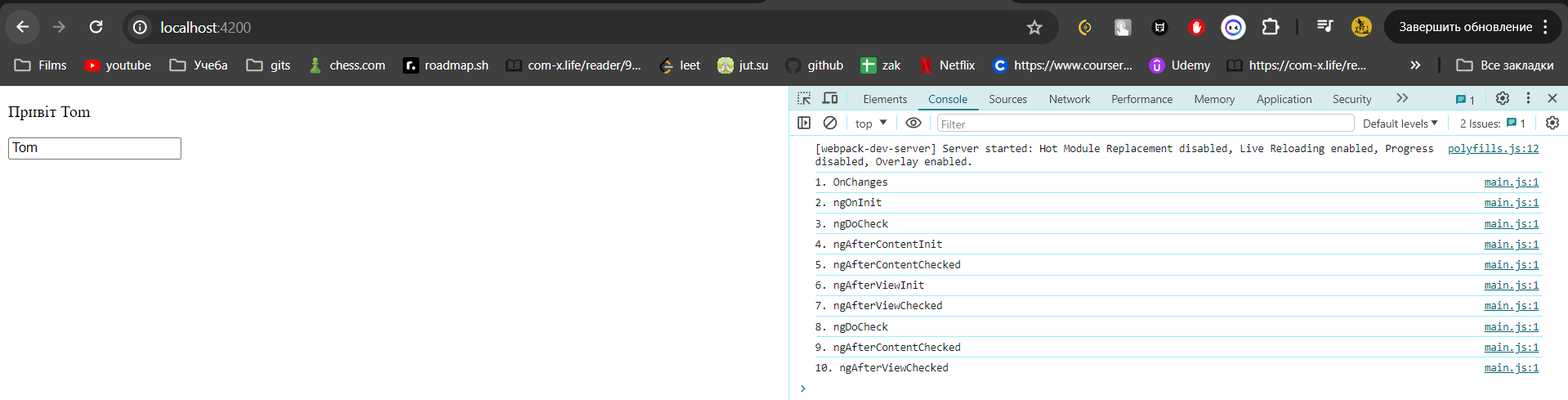




app.component:



Як бачимо з коду, тепер при будь-якій події відбуватиметься логування:

  
Рисунок 1.13. Результати логування усіх методів.

Посилання на додатки:

<https://holovniaip11lab2c1.web.app/>

https://holovniaip11lab2c5.web.app/

**Висновок:**

Під час виконання комп’ютерного практикуму я навчився змінювати стилі компоненту засобами Angular, підключати зовнішні файли, створювати допоміжні компоненти і як передавати у них дані через ng-content тег чи декоратор @Input, також отримувати дані з допомогою двосторонньої прив’язки через декоратор @Output та EventEmitter. Також ознайомився з життєвим циклом компоненту та навчився реагувати на його події. Мною було створено 5 додатків, два з яких було розгорнуто на платформі firebase.

**Список використаних джерел:**

1. Component Lifecycle: https://angular.io/guide/lifecycle-hooks