

# 03

## 함수 활용 실습





# 학습내용

- 01 챗봇 구현 소개
- 02 챗봇 구현 실습





## 학습목표

- 함수를 통해 입력 → 처리 → 출력의 구조를 표현할 수 있다.
- 챗봇의 동작 방식에서 일부 기능을 함수를 활용해 실습 코드를 작성할 수 있다.



# 01

## 챗봇 구현 소개



# 1) 메시지 분석기 함수(문자열 전처리 기초)



01

문자열의 양쪽 공백을 제거하는  
customTrim() 함수 구현

02

대문자를 소문자로 변환하는  
customToLowerCase() 함수 구현

03

이를 활용하여  
preprocessMessage() 구현

문자열



customTrim()  
문자열의 공백 제거



customToLowerCase()  
대소문자 변환



preprocessMessage()  
문자열 전처리 제어



전처리된 문자열 출력

# 1) 메시지 분석기 함수(문자열 전처리 기초)



- 문자열의 양쪽 공백을 제거하는 customTrim() 함수 구현

```
// 1. 문자열 앞뒤 공백 제거 함수
```

```
function customTrim(str) {
```

```
  let start = 0;
```

```
  let end = str.length - 1;
```

```
  while (start <= end && str[start] === ' '){
```

```
    start++;
```

```
}
```

```
  while (end >= start && str[end] === ' ')
```

```
{
```

```
    end--;
```

```
}
```

```
// 왼쪽 코드로 이어서...
```

```
let result = '';
```

```
for (let i = start; i <= end; i++) {
```

```
  result += str[i];
```

```
}
```

```
return result;
```

```
} // customTrim(str)
```

# 1) 메시지 분석기 함수(문자열 전처리 기초)



## customToLowerCase() 함수 구현

대문자를 소문자로 변환하는 기능

// 2. 문자열을 소문자로 바꾸는 함수(A-Z → a-z)

```
function customToLowerCase(str) {
```

```
  let result = "";
```

```
  for (let i = 0; i < str.length; i++) {
```

```
    const code = str.charCodeAt(i);
```

```
    result += (code >= 65 && code <= 90)
```

```
      ? String.fromCharCode(code + 32)
```

```
      : str[i]; //대문자면 소문자로 바꿔 추가하고, 아니면 그대로 추가함
```

```
  }
```

```
  return result;
```

```
}
```

// 사용 예시

```
console.log(customToLowerCase("HELLO World!")); // hello world!
```

# 1) 메시지 분석기 함수(문자열 전처리 기초)



preprocessMessage() 함수 구현

사용자 메시지를 정제하기 위한 제어 함수

// 3. 사용자 메시지를 정제하는 메인 함수

```
function preprocessMessage(message) {
```

```
  const trimmed = customTrim(message);
```

```
  const lowered = customToLowerCase(trimmed);
```

```
  return lowered;
```

```
}
```

// 테스트

```
console.log(preprocessMessage(" Hello ChatGPT! ")); // "hello chatgpt!"
```

```
console.log(preprocessMessage(" OPENAI ")); // "openai"
```



## 2) 입력 메시지에 따른 조건 분기 응답 생성



{ 차후 LLM API 응답 처리로 대체 가능 }

```
function getResponse(message) {  
  const cleaned = preprocessMessage(message);  
  // 입력 메시지에 대한 응답 내용 처리 로직  
  if (cleaned.includes("hello")) {  
    return "Hi there! How can I help you?";  
  } else if (cleaned.includes("bye")) {  
    return "Goodbye!";  
  } else {  
    return "I'm not sure what you mean.";  
  }  
}
```

```
// 왼쪽 코드로 이어서...  
// 테스트  
console.log(getResponse(" Hello there! "));  
console.log(getResponse("bye now"));  
console.log(getResponse("what is this?"));
```

### 3) 사용자-챗봇 대화 시뮬레이터(구조화된 함수 연결)



{ 함수를 구성 요소 단위로 나누고 연결해 작은 챗봇 흐름 구성 }

이후에는 LLM 호출, 대화 히스토리 등으로 확장 가능

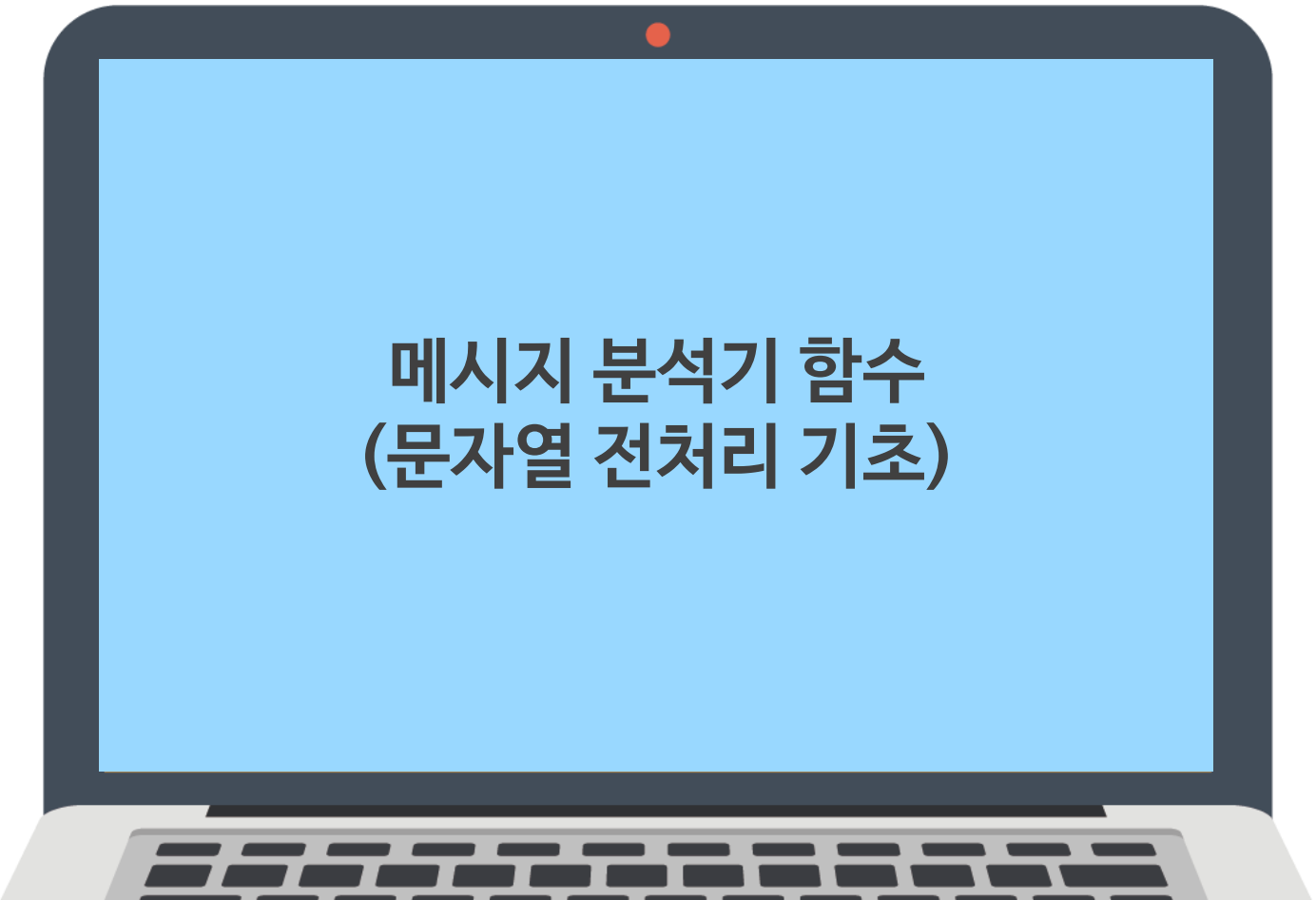
```
function simulateChat(userInput) {  
  const cleanedMessage = preprocessMessage(userInput);  
  const response = getResponse(cleanedMessage);  
  return `[User] ${userInput}\n[Bot] ${response}`; //백틱  
}  
  
// 테스트  
console.log(simulateChat("Hello, bot!"));  
console.log(simulateChat("Can you help me?"));  
console.log(simulateChat("bye"));
```



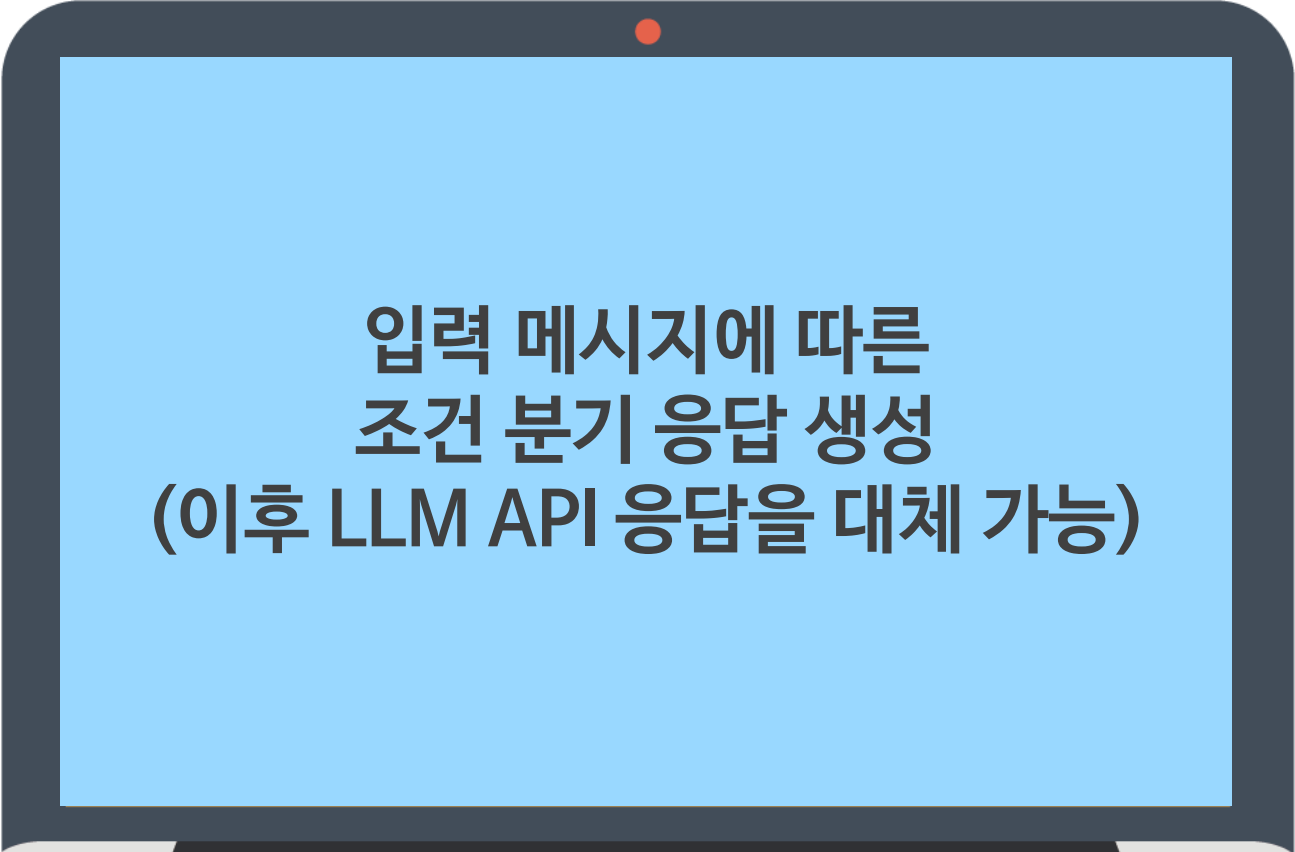
# 02

## 챗봇 구현 실습

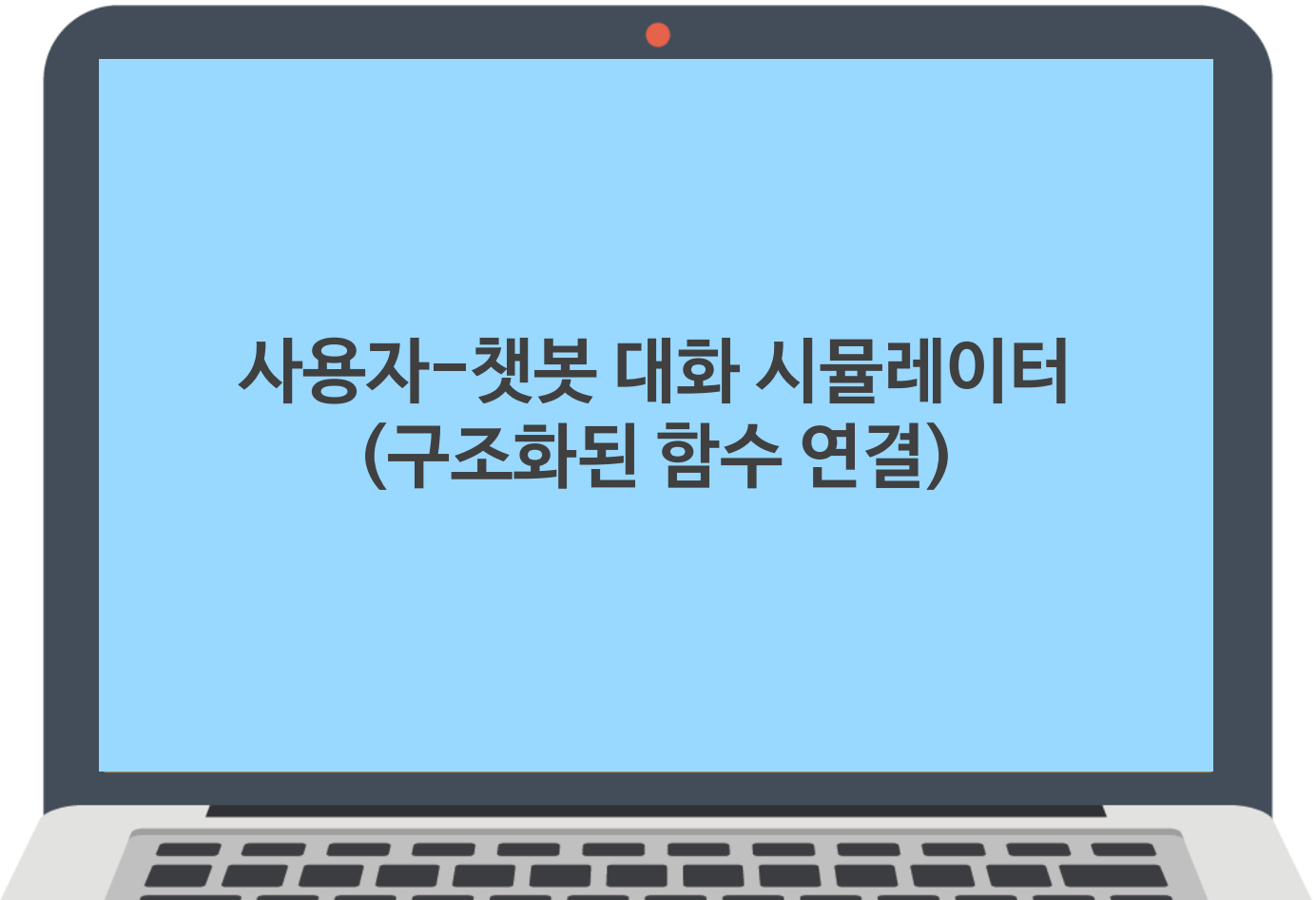




메시지 분석기 함수  
(문자열 전처리 기초)



입력 메시지에 따른  
조건 분기 응답 생성  
(이후 LLM API 응답을 대체 가능)



사용자-챗봇 대화 시뮬레이터  
(구조화된 함수 연결)



## 학습 평가

Q1

Q2

Q3

Q4

Q5

Q6

Q1

JavaScript에서 변수의 유효 범위를 무엇이라고 하는가?

- 1 변수 체인
- 2 컨텍스트
- 3 스코프
- 4 실행 컨텍스트



# 학습 평가

Q1

Q2

Q3

Q4

Q5

Q6

## Q1

JavaScript에서 변수의 유효 범위를 무엇이라고 하는가?

1 변수 체인

2 컨텍스트

☒ 3 스코프

4 실행 컨텍스트

정답

3

해설

스코프는 변수가 접근 가능한 코드 범위(유효 영역)를 말합니다.





## 학습 평가

Q1

Q2

Q3

Q4

Q5

Q6

Q2

다음 중 let 또는 const에 의해 형성되는 스코프는 무엇인가?

- 1 함수 스코프
- 2 전역 스코프
- 3 블록 스코프
- 4 매개변수 스코프

# 학습 평가

Q1

Q2

Q3

Q4

Q5

Q6

Q2

다음 중 let 또는 const에 의해 형성되는 스코프는 무엇인가?

- 1 함수 스코프
- 2 전역 스코프
- ☒ 3 블록 스코프
- 4 매개변수 스코프

정답

3

해설

let, const는 중괄호 {} 안에서만 유효한 블록 스코프를 형성합니다.



## 학습 평가

Q1

Q2

Q3

Q4

Q5

Q6

### Q3

다음 코드의 출력 결과는 무엇인가?

```
let a = 1;  
function test() {  
  let a = 2;  
  console.log(a);  
}  
test();
```

1

1

2

2

3

undefined

4

오류





## 학습 평가

Q1

Q2

Q3

Q4

Q5

Q6

Q3

다음 코드의 출력 결과는 무엇인가?

```
let a = 1;  
function test() {  
  let a = 2;  
  console.log(a);  
}  
test();
```

1

1



2

3

undefined

4

오류

정답

2

해설

스코프 체인에 의해 내부 let a = 2가 우선되므로 출력은 2가 됩니다.



## 학습 평가

Q1

Q2

Q3

Q4

Q5

Q6

### Q4

다음 코드의 출력 결과는 무엇인가?

```
(function() {  
  console.log("Hello");  
})();
```

1 Hello

2 undefined

3 오류

4 아무 것도 출력되지 않음

# 학습 평가

Q1

Q2

Q3

Q4

Q5

Q6

## Q4

다음 코드의 출력 결과는 무엇인가?

```
(function() {  
  console.log("Hello");  
})();
```



Hello

2

undefined

3

오류

4

아무 것도 출력되지 않음

**정답****1****해설**

즉시 실행 함수(IIFE)로 인해 “Hello”가 즉시 출력됩니다.

# 학습 평가

Q1

Q2

Q3

Q4

Q5

Q6

## Q5

다음 코드에서 callback 함수의 역할은 무엇인가?

```
function greet(name, callback) {  
  console.log("Hello " + name);  
  callback();  
}
```

- 1 greeting 종료 후 아무 역할 없음
- 2 greet 함수 호출 방지
- 3 후속 동작 실행
- 4 콘솔에 name 출력

# 학습 평가

Q1

Q2

Q3

Q4

Q5

Q6

## Q5

다음 코드에서 callback 함수의 역할은 무엇인가?

```
function greet(name, callback) {  
  console.log("Hello " + name);  
  callback();  
}
```

1

greeting 종료 후 아무 역할 없음

2

greet 함수 호출 방지



후속 동작 실행

4

콘솔에 name 출력

**정답****3****해설**

callback 변수는 Hello 출력 후 실행될 다음 동작을 담은 함수를 전달받아 실행하는 역할을 합니다.





## 학습 평가

Q1

Q2

Q3

Q4

Q5

Q6

### Q6

다음 중 고차 함수에 해당하는 것은 무엇인가?

- 1 값을 반환하는 함수
- 2 함수를 인자로 받거나 반환하는 함수
- 3 변수만 사용하는 함수
- 4 for문을 포함하는 함수



## 학습 평가

Q1

Q2

Q3

Q4

Q5

Q6

### Q6

다음 중 고차 함수에 해당하는 것은 무엇인가?

- 1 값을 반환하는 함수
- ☒ 2 함수를 인자로 받거나 반환하는 함수
- 3 변수만 사용하는 함수
- 4 for문을 포함하는 함수

정답

2

해설

고차 함수는 함수를 매개변수 또는 반환값으로 다룹니다.

# 학습정리

1/9

## 스cope(Scope)

- 스코프란

핵심 개념 요약	비고
변수의 유효 범위 (접근 가능한 코드 영역)	함수/블록 내부에서 선언된 변수는 외부에서 접근 불가

## 스cope(Scope)

- 스코프의 종류

핵심 개념 요약	비고
<ul style="list-style-type: none"><li>- 전역 스코프</li><li>- 함수 스코프(var)</li><li>- 블록 스코프(let, const)</li></ul>	<ul style="list-style-type: none"><li>- var : 함수 단위</li><li>- let/const : 블록 단위</li></ul>

## 스코프(Scope)

- 함수 레벨 스코프

핵심 개념 요약	비고
var로 선언된 변수는 함수 전체에서 유효	블록 무시 (if, for 안에서도 사용 가능)

# 학습정리

4/9

## 스cope (Scope)

- 렉시컬 스코프

핵심 개념 요약	비고
함수 정의 시점 기준으로 접근 가능 여부 결정	실행 위치가 아닌 선언 위치 기준

# 학습정리

5/9

## 스cope(Scope)

- 스코프 체인

핵심 개념 요약	예시
내부 함수에서 외부 변수 접근 가능(계층 구조)	함수 중첩 시 외부 변수 참조

## 고급 함수의 활용

- 즉시 실행 함수(IIFE)

핵심 개념 요약	비고
선언과 동시에 실행되는 함수	(function() { ... }) (); 구조, 전역 변수 오염 방지



## 고급 함수의 활용

- 중첩 함수

핵심 개념 요약	비고
함수 내부에 다른 함수 정의 가능	내부 함수는 외부 함수의 변수 접근 가능 (캡슐화)
외부 함수 종료 후에도 그 변수에 접근할 수 있는 함수	클로저

# 학습정리

8/9

## 고급 함수의 활용

- 고차 함수

핵심 개념 요약	예시
함수를 인자로 받거나 함수를 반환하는 함수	map, filter, forEach, reduce 등

## 고급 함수의 활용

- 콜백 함수

핵심 개념 요약	비고
다른 함수에 인자로 전달되는 함수	이벤트 처리, 비동기 작업 시 주로 사용 (setTimeout, addEventListener)

06주. 유지보수 비용을 줄이자! 객체지향 이야기

# 01

## 객체지향 프로그래밍의 개념

