CHAPTER 03 조건문

자바스크립트 프로그래밍 입문 (2판)

Contents

학습목표

- if 조건문의 기본 사용 방법을 익힙니다.
- if 조건문과 논리 연산자를 함께 사용하는 방법을 익힙니다.
- switch 조건문을 이해합니다.

내용

1. if 조건문

- 기본 형태

```
if (불_표현식) {

}
```

  - 불 표현식이 true이면 문장을 실행, false이면 문장을 무시함

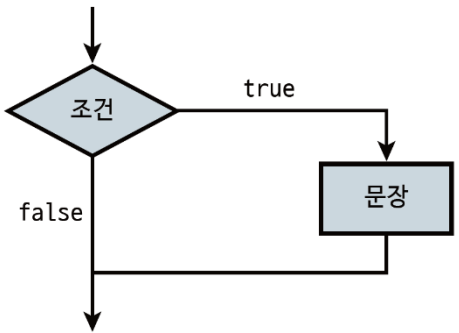

그림 3-1 if 조건문

1. if 조건문

- [예제 3-1] 홀수 짝수 구분(1)

코드 3-1 홀수와 짝수 구분 (1) conditionBasicA.js

```
let input = 32;

if (input % 2 == 0) {
    console.log("짝수입니다!");
}

if (input % 2 == 1) {
    console.log("홀수입니다!");
}
```

실행 결과

```
짝수입니다!
```

1. if 조건문

- [예제 3-2] 오전 오후 구분(1)
  - ❶ 현재 시간 구하기

코드 3-2 현재 시간 구하기

```
let date = new Date();

console.log(date.getFullYear());
console.log(date.getMonth());
console.log(date.getDay());
console.log(date.getHours());
console.log(date.getMinutes());
console.log(date.getSeconds());
```

1. if 조건문

- ❷ 오전과 오후 구분하기

코드 3-3 오전과 오후 구분 (1) conditionBasicB.js

```
let date = new Date();

if (date.getHours() < 12) {
    console.log("오전입니다.");
}

if (12 <= date.getHours()) {
    console.log("오후입니다.");
}
```

실행 결과

```
오후입니다.
```

2. if else 조건문

- 기본 형태

```
if (불_표현식) {
    // 불_표현식이 참일 때 실행할 문장
} else {
    // 불_표현식이 거짓일 때 실행할 문장
}
```

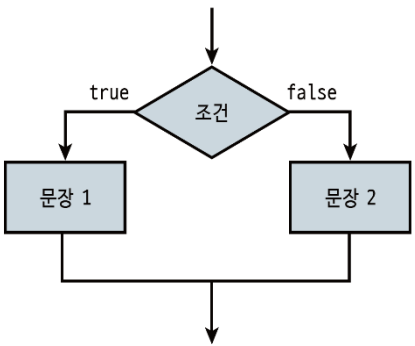

그림 3-2 if else 조건문

2. if else 조건문

- [예제 3-3] 홀수 짝수 구분(2)

코드 3-4 홀수와 짝수 구분 (2) conditionBasicC.js

```
let input = 32;

if (input % 2 == 0) {
    console.log("짝수입니다!");
} else {
    console.log("홀수입니다!");
}
```

실행 결과

```
짝수입니다!
```

2. if else 조건문

- [예제 3-4] 오전 오후 구분(2)

코드 3-5 오전과 오후 구분 (2) conditionBasicD.js

```
let date = new Date();

if (date.getHours() < 12) {
    console.log("오전입니다.");
} else {
    console.log("오후입니다.");
}
```

실행 결과

```
오후입니다.
```

3. 중첩 조건문

- 기본 형태

```
if (불_표현식) {
    if (불_표현식) {
        문장;
    } else {
        문장;
    }
} else {
    if (불_표현식) {
        문장;
    } else {
        문장;
    }
}
```

3. 중첩 조건문

- [예제 3-5] 중첩 조건문
  - DateTime.Now.Hour < 11 조건을 비교
  - false이면 DateTime.Now.Hour < 15 조건을 한 번 더 비교

코드 3-6 중첩 조건문 nestedCondition.js

```
let date = new Date();
let hours = date.getHours();

if (hours < 11) {
    console.log("아침 먹을 시간입니다.");
} else {
    if (hours < 15) {
        console.log("점심 먹을 시간입니다.");
    }
    else {
        console.log("저녁 먹을 시간입니다.");
    }
}
```

4. if else if 조건문

if else if 조건문

- 중복되지 않는 세 가지 이상의 조건을 구분할 때 사용

기본 형태

```
if (불_표현식) {

} else if (불_표현식) {

} else if (불_표현식) {

} else {

}
```

4. if else if 조건문

- [예제 3-6] if else if 조건문

코드 3-7 if else if 조건문 conditionBasicE.js

```
let date = new Date();
let hours = date.getHours();

if (hours < 11) {
    console.log("아침 먹을 시간입니다.");
} else if (hours < 15) {
    console.log("점심 먹을 시간입니다.");
} else {
    console.log("저녁 먹을 시간입니다.");
}
```

실행 결과

```
점심 먹을 시간입니다.
```

5. switch 조건문

- 기본 형태

```
switch (비교할_값) {
    case 값:
        문장
        break;
    case 값:
        문장
        break;
    default:
        문장
        break;
}
```

5. switch 조건문

- [예제 3-7] switch 조건문
  - 홀수와 짝수 구분

코드 3-10 switch 조건문 conditionBasicF.js

```
let input = 32;

switch (input % 2) {
    case 0:
        console.log("짝수입니다.");
        break;
    case 1:
        console.log("홀수입니다.");
        break;
}
```

5. switch 조건문

- switch 조건문

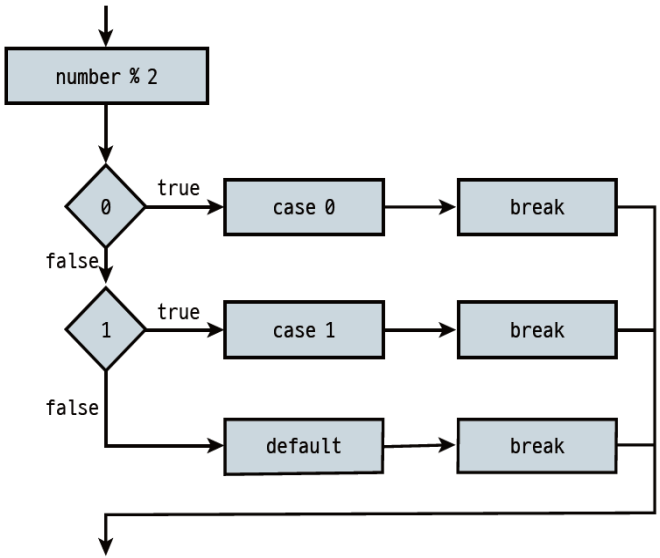

그림 3-3 switch 조건문

5. switch 조건문

- [예제 3-8] break 키워드를 사용하지 않는 switch 조건문

코드 3-11 break 키워드를 사용하지 않는 switch 조건문 switchWithoutBreak.js

```
let date = new Date();

switch (date.getMonth() + 1) {
    case 12:
    case 1:
    case 2:
        console.log("겨울입니다.");
        break;
    case 3:
    case 4:
    case 5:
        console.log("봄입니다.");
        break;
```

5. switch 조건문

```
    case 6:
    case 7:
    case 8:
        console.log("여름입니다.");
        break;
    case 9:
    case 10:
    case 11:
        console.log("가을입니다.");
        break;
    default:
        console.log("대체 어떤 행성에 살고 계신가요?");
        break;
}
```

실행 결과

```
가을입니다.
```

6. 조건 연산자

- 기본 형태

```
불_표현식 ? 참일_때_실행하고_밖으로_낼_값 : 거짓일_때_실행하고_밖으로_낼_값
```

- [예제 3-9] 조건 연산자를 활용한 변수 초기화
  - 변수가 undefined일 때 새로운 값으로 초기화

코드 3-12 조건 연산자를 활용한 변수 초기화 switchWithoutBreak.js

```
// 변수를 선언합니다.
let test;

// 삼항 연산자로 해당 변수가 undefined인지 확인하고 초기화합니다.
test = typeof(test) != 'undefined' ? test : "초기화_1";
console.log(test)

// 삼항 연산자로 해당 변수가 undefined인지 확인하고 초기화합니다.
test = typeof(test) != 'undefined' ? test : "초기화_2";
console.log(test)
```

undefined인지 확인합니다.

실행 결과

```
초기화_1
초기화_1
```

6. 조건 연산자

- 조건문으로 변경한 코드
- 조건 연산자를 활용하는 것이 더 간단하므로, 일반적으로 조건 연산자 사용

코드 3-13

```
let test;

if (typeof(test) != 'undefined') {
  test = "초기화_1"
}
console.log(test)

if (typeof(test) != 'undefined') {
  test = "초기화_2"
}
console.log(test)
```

7. 조금 더 나아가기

- prompt() : 입력 받는 함수. 웹 브라우저에서 사용 가능
  - Node.js에서 작동하는 자바스크립트는 입력을 받을 수 없음
  - Node.js의 사상 → '사용자의 응답을 기다리는 시간',
    '파일을 읽어 들이는 시간',
    '통신에 사용되는 시간'
    을 '기다리는 느린 코드를 절대 만들지 못하게 하겠다'

7. 조금 더 나아가기

- 입력을 받는 방법 소개

코드 3-14 기본적인 입력

```
// 모듈을 추출합니다.
const repl = require('repl');

// 입력을 시작합니다.
repl.start({
    prompt: "입력_때_앞에_출력할_문자열",
    eval: (command, context, filename, callback) => {
        // 입력(command)을 받았을 때 처리를 수행합니다.

        // 처리 완료
        callback();
    }
});
```

8. 조금 더 나아가기

- 입력을 받은 대상이 숫자인지 아닌지 확인하는 예제

코드 3-15 입력 방법 input.js

```
// 모듈을 추출합니다.
const repl = require('repl');

// 입력을 시작합니다.
repl.start({
    prompt: "숫자 입력> ",
    eval: (command, context, filename, callback) => {
        // 입력(command)을 받았을 때 처리를 수행합니다.
        let number = Number(command);

        // 입력이 숫자인지 확인합니다.
        if(isNaN(number)) {
            console.log("숫자가 아닙니다.");
        } else {
            console.log("숫자입니다.");
        }

        // 처리 완료
        callback();
    }
});
```

실행 결과

```
숫자 입력> 10
숫자입니다.
숫자 입력> 20
숫자입니다.
숫자 입력> 30
숫자입니다.
숫자 입력> 안녕
숫자가 아닙니다.
숫자 입력> 하세요
숫자가 아닙니다.
숫자 입력>
```


THANK
YOU!