1. 커스텀 훅 (1)

- 우리가 지금까지 살펴본 훅

  `useState()` `useEffect()` `useRef()` `useMemo()` `useCallback()`

- 커스텀 훅

: 재사용 가능한 함수로 stateful한 로직을 아웃소싱

: 일반 함수(e.g. 모듈)과 다른 점은 커스텀 훅은 다른 리액트 훅이나 리액트 상태로 사용할 수 있다.

1. 커스텀 훅 만들어 보기

- `practice 4_18`

```
import React from 'react';
import ForwardCounter from './compoenents/ForwardCounter';
import BackwardCounter from './components/BackwordCounter';

function App() {
    return <>
        <ForwardCounter />
        <BackwordCounter />
    </>;
}

export defalt App;
```

```
//ForwardCounter
import { useState, useEffect } from 'react';
import Card from './UI/Card/Card';

const ForwardCounter = () => {
    const [counter, setCounter] = useState(0);

    useEffect(() => {
        const interval = setInterval(() => {
            setCounter((prevCounter) => prevCounter + 1);
        }, 1000);

        return () => clearInterval(interval);
    }, []);

    return <Card>{counter}</Card>;
};

export default ForwardCoutner;
```

```
//BackwardCounter

import { useState, useEffect } from 'react';
import Card from './UI/Card/Card';
```

```
import useCounter from "../hooks/useCounter";

const BackwardCounter = () => {
    //useCounter 훅
    const counter = useCounter(true);

    return <Card>{counter}</Card>;
};

export default BackwardCounter;
```

```
//hooks 폴더 생성
//useCounter.js 파일 생성

import React, {useEffect, useState } from 'react';

const useCounter = (isForward) => {
    const [counter, setCounter] = useState(0);

    useEffect(() => {
        const interval = setInterval(() => {
            setCounter((prevCounter) => prevCounter - 1 *(isForward ? 1 : -1));
        }, 1000);

        return () => clearInterval(interval);
    }, []);

    return counter;
};


export default useCounter;
```

- 커스텀 훅은 재사용 가능한 함수로 stateful한 로직을 아웃소싱한다.
- 일반함수와 다른 점은 컴스텀 훅은 다른 리액트 훅이나 리액트 상태를 사용할 수 있다.