5주차

# 소프트웨어 시스템 설계 및 개발

2024.1학기

# CONTENTS

1. Node.js 기초 - Node.js 교과서(길벗)
2. 예제
3. 실습

# 예제 1

- **text 분해와 결합**

```
const text = 'Hello World!';
console.log(text);
console.log(text.split(' '));
console.log(text.split(' ').join(', '));


const text2 = 'abc';
console.log(text2);
console.log(text2.split(''));
console.log(text2.split('').map(t => {
    if(t === 'a'){
        t = 'e';
    }
    return t;
}));
console.log(text2.split('').map(t => {
    if(t === 'a'){
        t = 'e';
    }
    return t;
}).join(',')
);
```

# 예제 2

- 파일 읽기 : 파일경로를 입력하면 해당 파일을 읽어서 출력

```
const fs = require('fs');
const readline = require('readline');

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout
});

function procFile() {
    rl.question('파일 경로를 입력해주세요 : ', (filePath) => {
        fs.readFile(filePath, 'utf8', (err, data) => {
            if (err) {
              console.error('파일을 읽는 동안 오류가 발생했습니다:', err);
              return;
            }
            console.log('처리된 텍스트:', data);
        });
        rl.close();
    });
}

procFile();
```

# 예제 3

- 파일 쓰기 : 파일 내용을 입력하면 해당 내용을 'a.out' 파일에 쓰기

```
const fs = require('fs');
const readline = require('readline');

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout
});

function procFile() {
    rl.question('파일 내용을 입력해주세요 : ', (data) => {
        fs.writeFile('a.out', data, (err) => {
            if (err) {
              console.error('파일을 쓰는 동안 오류가 발생했습니다:', err);
            } else {
              console.log(`파일이 성공적으로 쓰여졌습니다.`);
            }
          });
        rl.close();
    });
}

procFile();
```

# 예제 4

- **시저 암호(CaesarCipher) : 문자열을 특정 숫자만큼 미는 암호**

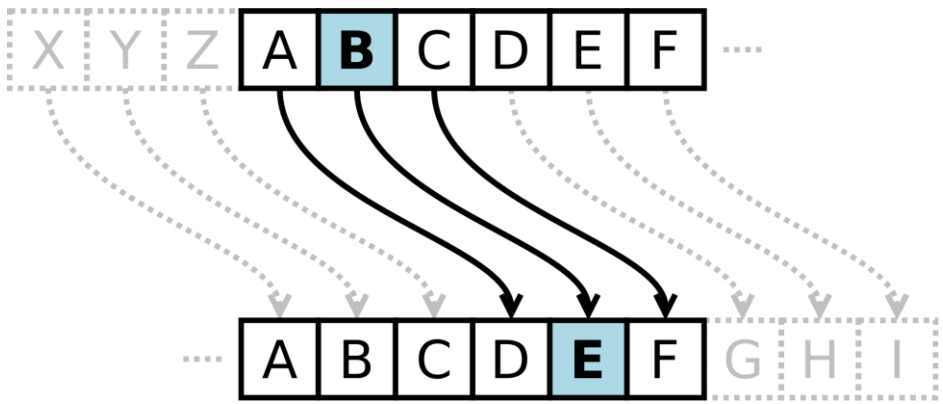
```
const fs = require('fs');
const readline = require('readline');

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout
});

function caesarCipher(text, shift) {
  return text.split('').map(char => {
    if (char.match(/[a-z]/i)) { // 정규표현식으로, i 플래그를 포함하여 a-z까지 문자를 대소문자 구분없이 찾음
      let code = char.charCodeAt();

      if (code >= 65 && code <= 90) {
        return String.fromCharCode(((code - 65 + shift) % 26 + 26) % 26 + 65);
      } else if (code >= 97 && code <= 122) {
        return String.fromCharCode(((code - 97 + shift) % 26 + 26) % 26 + 97);
      }
    }
    return char;
  }).join('');
}
```

# 예제 4




```
function ask() {
  rl.question('파일 경로를 입력해주세요 : ', (filePath) => {
    rl.question('암호화는 +, 복호화는 -를 입력해주세요(+,-) : ', (method) => {
      rl.question('암호화 키를 입력해주세요(정수) : ', (key) => {
        const shift = parseInt(key);
        if (isNaN(shift)) {
            console.log('잘못된 키 값입니다. 숫자를 입력해주세요.');
            rl.close();
            return;
        }
        processFile(filePath, method, shift);
      });
    });
  });
}

function processFile(filePath, method, shift) {
  fs.readFile(filePath, 'utf8', (err, data) => {
    if (err) {
      console.error('파일을 읽는 동안 오류가 발생했습니다:', err);
      return;
    }

    const k = method === '+' ? shift : -shift;
    const proc = method === '+' ? 'enc' : 'dec';

    const processedText = caesarCipher(data, k);
    console.log('처리된 텍스트:', processedText);

    // 결과를 새 파일에 저장
    const outputPath = `${filePath}_${proc}`;
    fs.writeFile(outputPath, processedText, (err) => {
      if (err) {
        console.error('파일을 쓰는 동안 오류가 발생했습니다:', err);
      } else {
        console.log(`파일이 성공적으로 쓰여졌습니다: ${outputPath}`);
      }
    });
  });
  rl.close();
}

// 호출 흐름 : ask -> processFile -> caesarCipher
ask();
```

# 예제 5

- **http모듈을 활용한 간단한 서버 실행**

```
const { createServer } = require('http');

const hostname = '127.0.0.1';
const port = 3000;

const server = createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

# 실습

- node 채팅 서버 (npm 설치 필요)
- 채팅서버 실행 후, index.html을 실행하여 접속

```
// 중요! 소스코드의 경로에 한글이 있으면 안 됨
// npm init -y
// npm install ws

const WebSocket = require('ws');

const wss = new WebSocket.Server({ port: 8080 });

wss.on('connection', function connection(ws) {
  ws.on('message', function incoming(message) {
    console.log('received: %s', message);

    // 받은 메시지를 모든 클라이언트에게 보냄
    wss.clients.forEach(function each(client) {
      if (client.readyState === WebSocket.OPEN) {
        client.send(message);
      }
    });
  });

  ws.send('Welcome to the chat server!');
});

console.log('Chat server is running on ws://localhost:8080');
```

chat.js

```
<!DOCTYPE html>
<html>
<head>
    <title>Chat App</title>
</head>
<body>
    <div id="chat"></div>
    <input type="text" id="messageInput" placeholder="Type a message...">
    <button onclick="sendMessage()">Send</button>

    <script>
        const ws = new WebSocket('ws://localhost:8080');
        const chat = document.getElementById('chat');
        const messageInput = document.getElementById('messageInput');

        ws.onmessage = function(event) {
            // event.data가 Blob 인스턴스일 경우 FileReader를 사용해서 읽어오기
            if (event.data instanceof Blob) {
                const reader = new FileReader();
                reader.onload = function() {
                    const message = document.createElement('p');
                    message.textContent = reader.result;
                    chat.appendChild(message);
                };
                reader.readAsText(event.data);
            } else {
                // 메시지가 문자열인 경우, 직접 표시
                const message = document.createElement('p');
                message.textContent = event.data;
                chat.appendChild(message);
            }
        };

        function sendMessage() {
            const message = messageInput.value;
            ws.send(message);
            messageInput.value = '';
        }
    </script>
</body>
</html>
```

index.html

# 조별 과제

- 실습의 소스코드를 활용
- 조원별로 조금이라도 소스코드에 기여하여 조금이라도 채팅 실습이 발전하면 됨
  - ex: 채팅창 외곽선 박스, 채팅에 닉네임 출력 등

- 과제 정보
  - 조원 구성 (4인 이상 자유)
  - **제출 방법** : 메일로 제출하면서 pjk5401 깃헙 ID를 레포에 초대(**중요! 레포 확인 불가시 채점 안됨**)
  - **제출 내용** : 조별 프로젝트 레포지터리(비공개), 조원의 깃헙 ID
  - **제출 마감** : 4/22(1분반), 4/24(2분반) 수업시간 전까지
  - **점수**
    - 과제 제출 완료 및 기여함 : 15점
    - 과제 제출 및 미기여 : 5점
    - 과제 미제출 : 0점