Babel, Webpack

Babel과 Webpack

- Babel
  - ES6+ 문법의 자바스크립트 코드를 구버전의 웹브라우저에서 돌아가도록 변환해 준다.
- Webpack
  - 프로젝트 내의 여러 자바 스크립트와 스타일시트, 이미지 등을 의존성을 참조하여 하나의 파일로 만들어준다
  - html에서는 만들어진 최종 자바스크립트만 불러 쓰면 된다.

Babel 실행

- @babel/cli 로 터미널에서 직접 실행
- babel.config.js 로 각정 설정해서 실행하기
  - @babel/preset-env, @babel/preset-react 등의 미리 만들어둔 설정들도 있다.
- webpack의 babel-loader로 실행하기
  - webpack 과 함께 설치해서 loader 를 babel-loader로 사용하도록 설정

프로젝트 초기화

- 프로젝트 폴더 생성 후 해당 프로젝트 폴더의 터미널에서
- npm init -y
- npm install -D @babel/core @bable/cli
- npm install -D @babel/preset-env

Babel config

- 프로젝트의 루트 폴더에 babel.config.js 파일을 추가

```
module.exports = {
    presets: [
      '@babel/preset-env'
    ]
  }
```

package.json

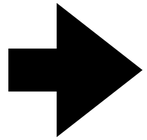
- package.json 수정
- babel 원본폴더 -w -d 출력폴더
  - -w :watch 자동 감지하여 변환
  - -d 출력폴더 :변환 결과물을 저장할 폴더

```
{
  "name": "babel-webpack",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  ▷ Debug
  "scripts": {
    "build": "babel src/js -w -d dist/js"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "devDependencies": {
    "@babel/cli": "^7.14.5",
    "@babel/core": "^7.14.6",
    "@babel/preset-env": "^7.14.7"
  }
}
```

실습

src/js/test.js

```
1   let age=10
2   const prefix="Hello"
3
4   let message=''
5   if(age > 19)
6       message=`${prefix}, Mr/Ms.`
7   else
8       message=`${prefix}, student.`
9   console.log(message)
```



dist/test.js

```
1   "use strict";
2
3   var age = 10;
4   var prefix = "Hello";
5   var message = '';
6   if (age > 19) message = "".concat(prefix, ", Mr/Ms.");
7   else message = "".concat(prefix, ", student.");
8   console.log(message);
```

문제점

- 자바스크립트들 간의 import 나 require는 Babel로 정상 동작함
- 그러나 이렇게 생성된 자바스크립트를 html에서 사용할 수는 없음
  - 웹브라우저는 import를 처리 못함
  - webpack을 이용해 통합된 파일로 제공할 수 있음

Webpack 개념: Entry

- 파일이 다른 파일에 의존할 때마다 weboack은 이를 ‘의존성’으로 처리. 이미지, 웹폰트 등도 의존성으로 제공됨.
- 의존성 그래프를 생성하기 위해 사용하는 모듈(시작점)을 Entry point라고 함.
- webpack은 Entry point가 의존하는 다른 모듈과 라이브러리를 찾아냄.
- 기본 값은 ./src/index.js 이며 설정에서 수정가능함.

Output

- 생성된 번들을 보낼 위치와 파일이름을 설정
- 기본 값은 ./dist/main.js

Loader

- 자바스크립트, JSON 파일 및 다른 형식의 파일을 처리할 수 있도록 모듈로 변환하거나 의존성 그래프에 추가해줌
- test와 use 속성을 가짐
  - test: 변환할 파일을 식별하는 속성
  - use: 변환을 수행할 로더를 가리킴

Webpack 설치

- npm install -D webpack webpack-cli
- npm install -D babel-loader

Webpack 설치

- package.json을 열어 build 명령어 실행 시 webpack이 실행되도록 한다.

```
{
  "name": "babel-webpack",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  ▷ Debug
  "scripts": {
    "build": "webpack -w"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "devDependencies": {
    "@babel/cli": "^7.14.5",
    "@babel/core": "^7.14.6",
    "@babel/preset-env": "^7.14.7",
    "babel-loader": "^8.2.2",
    "webpack": "^5.44.0",
    "webpack-cli": "^4.7.2"
  }
}
```

js files

- src/js 폴더에 다음 파일들 추가

src/js/maht_lib.js

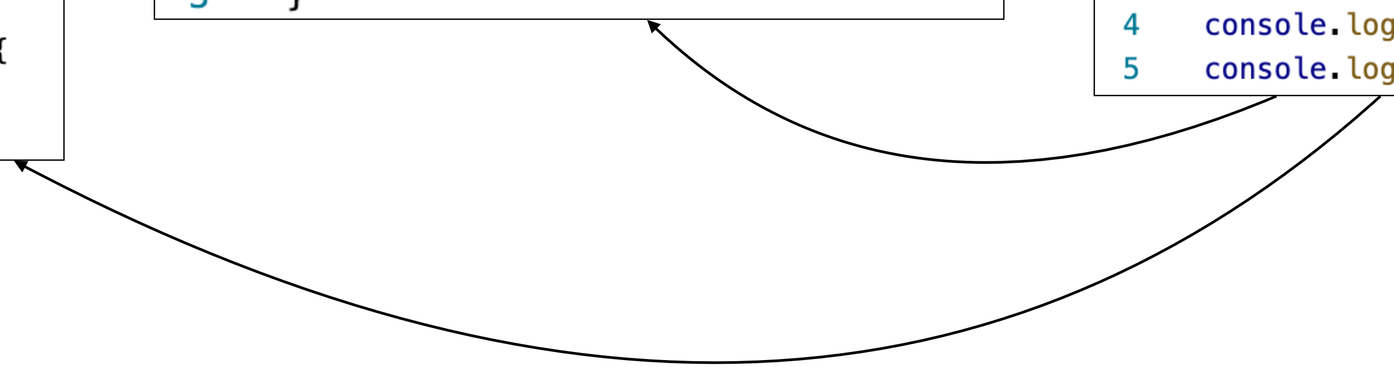
```
1   export function pow(a, b){
2       return a**b
3   }
4
5   export function mod(a, b){
6       return a%b
7   }
```

src/js/str_lib.js

```
1   exports.hello= (name='user')=>{
2       return `Hello, ${name}`
3   }
```

src/js/main.js

```
1   import { pow, mod } from "./math_lib";
2   const str_lib = require('./str_lib')
3
4   console.log(pow(2, 3))
5   console.log(str_lib.hello('ABC'))
```



index.html

- 프로젝트 루트에 index.html 작성

index.html

```
1    <!DOCTYPE html>
2    <head>
3
4    </head>
5    <body>
6        <script src="./dist/js/bundle.js"></script>
7    </body>
8    </html>
```

webpack 실행

- npm run build

```
suyeoncho@Suyeonui-MacBook-Pro babel-webpack % npm run build

> babel-webpack@1.0.0 build
> webpack -w

asset bundle.js 4.78 KiB [compared for emit] (name: main)
runtime modules 670 bytes 3 modules
cacheable modules 379 bytes
  ./src/js/main.js 125 bytes [built] [code generated]
  ./src/js/math_lib.js 100 bytes [built] [code generated]
  ./src/js/str_lib.js 154 bytes [built] [code generated]
webpack 5.44.0 compiled successfully in 466 ms
```