

# webpack, module, babel

## Webpack, Module, Babel

Web development has evolved significantly over the years, and with it, the tools and technologies used for building web applications have also evolved. One such tool that has gained immense popularity among web developers is Webpack.

### Webpack

Webpack is a module bundler for JavaScript applications. It takes your code and its dependencies and bundles them into a single, optimized file that can be served to the browser. Webpack is highly configurable and can be used for both simple and complex applications. It comes with a built-in development server that makes it easy to develop and test your application.

In the context of web development, a dependency refers to a package or library that is required by a project to function properly. Dependencies can be managed using package managers such as npm and yarn, which allow developers to easily install, update, and remove dependencies as needed.

In web development, bundlers are used to manage and optimize a project's codebase. They help package all the code and its dependencies into a single or multiple files that can be loaded by the browser. By doing so, bundlers allow developers to write modular code and import/export different parts of the code as needed, without worrying about how it will be served to the browser.

Webpack is one of the most popular and widely used bundlers in the web development community. It can be used for both simple and complex applications and comes with a built-in development server that makes it easy to develop and test your application. Webpack is highly configurable, allowing developers to customize it to meet their specific needs.

Bundlers like Webpack are especially useful for larger projects, where managing dependencies and optimizing code can become a challenge. By bundling all the code

and its dependencies into a single file, bundlers can significantly improve a website's performance by reducing the number of requests made to the server. This can result in faster load times and improved user experience.

In summary, a bundler is a tool used in web development to manage and optimize a project's codebase. Bundlers like Webpack allow developers to write modular code and package it with its dependencies into a single or multiple files that can be loaded by the browser. This can significantly improve a website's performance and user experience.

## Module

In JavaScript, a module is a piece of code that encapsulates related functionality. Modules help keep our code organized and maintainable by providing a way to organize code into logical units. Modules can be imported and exported, allowing us to reuse code across different parts of our application.

## Babel

Babel is a JavaScript compiler that allows us to write modern JavaScript code and have it transpiled to a compatible version that can run on older browsers. Babel supports the latest features of JavaScript, such as arrow functions, template literals, and destructuring, and compiles them down to ES5, which is widely supported by browsers.

In conclusion, Webpack, module, and Babel are essential tools for modern web development. They help us write maintainable, scalable, and optimized code that can run on a variety of browsers and devices. While there are other tools available, these three are among the most popular and widely used in the web development community.


## 웹팩, 모듈, 바벨

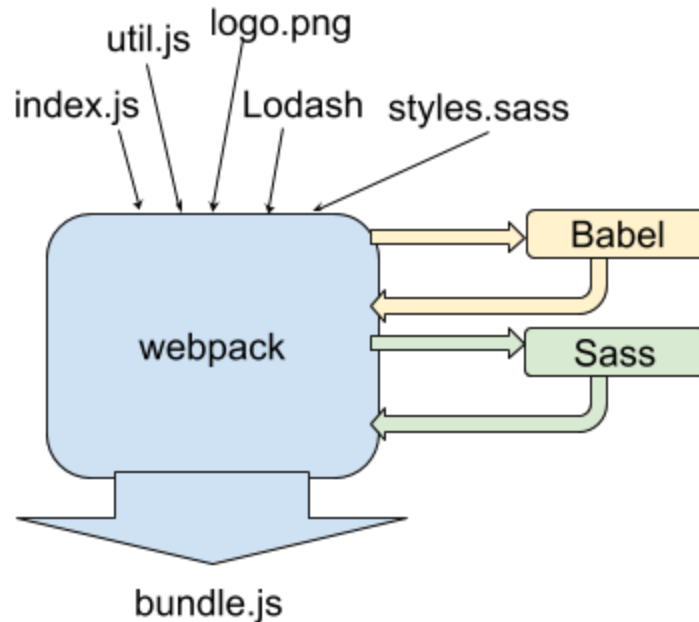
웹 개발은 수년간 크게 발전해 왔고, 이에 따라 웹 애플리케이션을 구축하는 데 사용되는 도구와 기술도 발전해 왔습니다. 웹 개발자들 사이에서 인기를 얻은 도구 중 하나는 웹팩입니다.

### 웹팩

웹팩은 자바스크립트 애플리케이션을 위한 모듈 번들러입니다. 코드와 해당 코드가 의존하는 모듈들을 하나로 묶어 최적화된 파일로 만들어 브라우저에 제공합니다. 웹팩은 매우 유연하며, 작

은 프로젝트부터 복잡한 프로젝트까지 모두 사용할 수 있습니다. 빌드된 결과물을 직접 확인할 수 있는 개발 서버도 내장되어 있어, 애플리케이션을 개발하고 테스트하기 용이합니다.

- **번들러** *bundler* - 프로젝트에 사용되는 파일들을 하나 또는 소수의 파일들로 압축
- 어플리케이션이 로딩 및 실행 속도 향상
- 각종 플러그인과 옵션을 사용하여 코드를 다양한 방법으로 변환/압축 가능
-  [공식 사이트 보기](#)
- 동종/유사 제품: RollUp, Parcel, Gulp, Vite...



#### ▼ 프로젝트에서 사용해보기

### 프로젝트에 사용해보기

#### 0. 소스 저장소 분리하기

- `src` 폴더를 만들고 `.js` 파일들 모두 이동

## 1. 프로젝트에 웹팩 설치

```
npm install webpack webpack-cli --save-dev
```

## 2. 웹팩 설정 파일

webpack.config.js

```
const path = require('path'); module.exports = {  entry: './src/main.js',  output: {    filename: 'main.js',    path: path.resolve(__dirname, 'dist'),  },  // 💡 추가설정들  watch: true, // 파일 수정 후 저장시 자동으로 다시 빌드  experiments: {    topLevelAwait: true // 모듈 await 가능하도록  }  };
```

- ./src/main.js 파일과, 연결된 모든 모듈들을 ./dist/main.js 파일로 통합

## 3. 빌드 명령 추가

package.json

```
"scripts": {  "build": "webpack" },
```

- script 항목에 "build": "webpack" 추가
- { "type": "module" } 부분 제거하기

## 4. 빌드 및 실행

```
npm run build
```

- 위 명령어로 빌드 실행
- 💡 ./dist/main.js 파일 확인 - javascript beautifier 사이트들에서 살펴보기
- HTML 파일에서 script 의 src 를 ./dist/main.js 로 변경
- 페이지에서 확인, 코드 수정 후 변화 확인

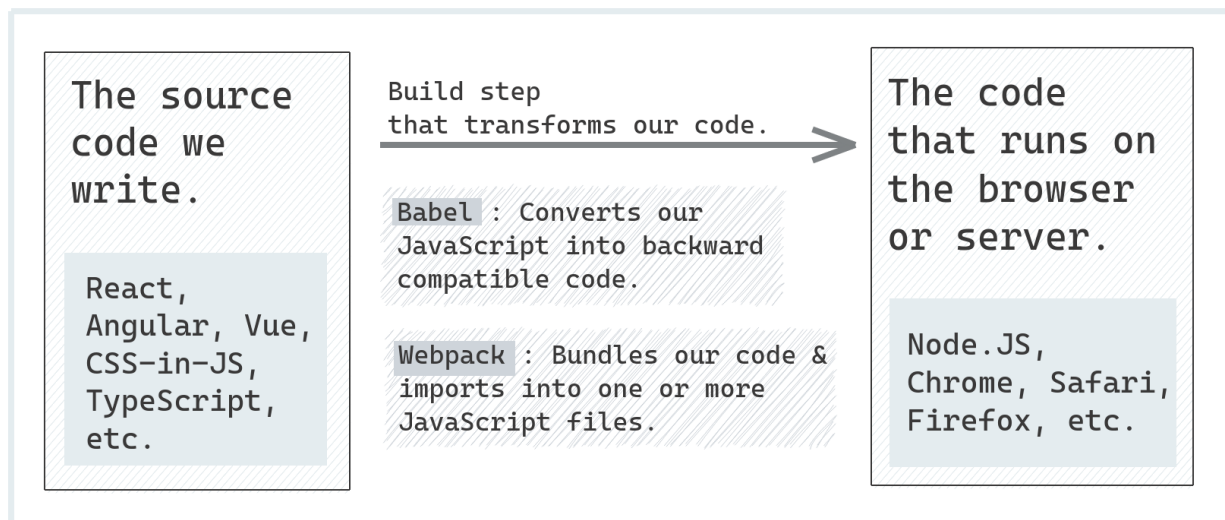
웹 개발에서 의존성(dependency)이란, 프로젝트가 올바르게 작동하기 위해 필요한 패키지나 라이브러리를 의미합니다. 의존성은 npm과 yarn과 같은 패키지 매니저를 사용하여 관리할 수 있으며, 필요할 때 쉽게 설치, 업데이트, 삭제할 수 있습니다.

번들러(bundler)는 프로젝트의 코드를 관리하고 최적화하는 데 사용됩니다. 번들러는 프로젝트의 코드와 해당 코드가 의존하는 모듈들을 하나 혹은 여러 개의 파일로 묶어 브라우저에서 로드할 수 있도록 합니다. 이로 인해, 번들러를 사용하면 모듈화된 코드를 쉽게 작성하고 필요한 부분을 가져다 쓸 수 있습니다.

웹팩은 웹 개발자들 사이에서 가장 인기 있는 번들러 중 하나입니다. 작은 프로젝트부터 복잡한 프로젝트까지 모두에게 적합하며, 빌드된 결과물을 직접 확인할 수 있는 개발 서버도 내장되어 있어, 애플리케이션을 개발하고 테스트하기 용이합니다. 또한, 매우 유연하여 개발자가 필요에 맞게 웹팩을 설정할 수 있습니다.

번들러와 같은 웹 개발 도구는 대규모 프로젝트에서 특히 유용합니다. 의존성을 관리하고 코드를 최적화하는 것이 어려워지기 때문입니다. 모든 코드와 해당 코드가 의존하는 모듈들을 하나의 파일로 묶음으로써, 번들러는 서버로 보내는 요청 수를 줄여 웹 사이트의 성능을 크게 향상시킬 수 있습니다. 이는 더 빠른 로딩 시간과 개선된 사용자 경험으로 이어질 수 있습니다.

요약하자면, 번들러는 웹 개발에서 프로젝트의 코드를 관리하고 최적화하기 위한 도구입니다. 웹팩 같은 번들러를 사용하면 모듈화된 코드를 쉽게 작성하고 필요한 부분을 가져다 쓸 수 있으며, 이는 웹 사이트의 성능과 사용자 경험을 크게 향상시킬 수 있습니다.



## 모듈

자바스크립트에서 모듈은 관련 기능을 캡슐화한 코드 조각입니다. 모듈을 사용하면 코드를 논리적 단위로 구성하여 유지보수하기 쉬워집니다. 모듈은 가져오고 내보내는 것이 가능하므로, 애플리케이션의 다른 부분에서도 재사용할 수 있습니다.

모듈(module)이란 프로그램을 구성하는 구성 요소로, 관련된 데이터와 함수를 하나로 묶은 단위를 의미합니다.

보통 하나의 소스 파일에 모든 함수를 작성하지 않고, 함수의 기능별로 따로 모듈을 구성합니다.

이러한 모듈을 합쳐 하나의 파일로 작성하는 방식으로 프로그램을 만들게 됩니다.

위처럼 프로그램 코드를 기능별로 나눠서 독립된 파일에 저장하여 관리하는 방식을 모듈화 프로그래밍이라고 합니다.

개발하는 애플리케이션의 크기가 커지면 언젠간 파일을 여러 개로 분리해야 하는 시점이 옵니다. 이때 분리된 파일 각각을 '모듈(module)'이라고 부르는데, 모듈은 대개 클래스 하나 혹은 특정한 목적을 가진 복수의 함수로 구성된 라이브러리 하나로 구성됩니다.

자바스크립트가 만들어진 지 얼마 안 되었을 때는 자바스크립트로 만든 스크립트의 크기도 작고 기능도 단순했기 때문에 자바스크립트는 긴 세월 동안 모듈 관련 표준 문법 없이 성장할 수 있었습니다. 새로운 문법을 만들 필요가 없었던 것이죠.

그런데 스크립트의 크기가 점차 커지고 기능도 복잡해지자 자바스크립트 커뮤니티는 특별한 라이브러리를 만들어 필요한 모듈을 언제든지 불러올 수 있게 해준다거나 코드를 모듈 단위로 구성해 주는 방법을 만드는 등 다양한 시도를 하게 됩니다.

그 시도는 다음과 같은 모듈 시스템으로 이어졌습니다.

- AMD – 가장 오래된 모듈 시스템 중 하나로 require.js라는 라이브러리를 통해 처음 개발되었습니다.
- CommonJS – Node.js 서버를 위해 만들어진 모듈 시스템입니다.
- UMD – AMD와 CommonJS와 같은 다양한 모듈 시스템을 함께 사용하기 위해 만들어졌습니다.

이런 모듈 시스템은 오래된 스크립트에서 여전히 발견할 수 있는데, 이제는 역사의 뒤안길로 사라져가고 있습니다.

모듈 시스템은 2015년에 표준으로 등재되었습니다. 이 이후로 관련 문법은 진화를 거듭해 이제는 대부분의 주요 브라우저와 Node.js가 모듈 시스템을 지원하고 있습니다. 이제 본격적으로 모던 자바스크립트에서 쓰이는 모듈에 대해 알아보시다.

## 모듈이란

모듈은 단지 파일 하나에 불과합니다. 스크립트 하나는 모듈 하나입니다.

모듈에 특수한 지시자 `export` 와 `import` 를 적용하면 다른 모듈을 불러와 불러온 모듈에 있는 함수를 호출하는 것과 같은 기능 공유가 가능합니다.

- `export` 지시자를 변수나 함수 앞에 붙이면 외부 모듈에서 해당 변수나 함수에 접근할 수 있습니다(`모듈 내보내기`).
- `import` 지시자를 사용하면 외부 모듈의 기능을 가져올 수 있습니다(`모듈 가져오기`).

`export` 지시자를 사용해 파일 `sayHi.js` 내부의 함수 `sayHi` 를 외부로 내보내 봅시다.

```
// 📁 sayHi.js
export function sayHi(user) {
  alert(`Hello, ${user}!`);
}
```

이제 `import` 지시자를 사용해 `main.js` 에서 함수 `sayHi` 를 사용할 수 있게 해봅시다.

```
// 📁 main.js
import {sayHi} from './sayHi.js';

alert(sayHi); // 함수
sayHi('John'); // Hello, John!
```

위 예시에서 `import` 지시자는 상대 경로(`./sayHi.js`) 기준으로 모듈을 가져오고 `sayHi.js` 에서 내보낸 함수 `sayHi` 를 상응하는 변수에 할당합니다.

이제 브라우저에서 모듈이 어떻게 동작하는지 예시를 이용해 알아보시다.

모듈은 특수한 키워드나 기능과 함께 사용되므로 `<script type="module">` 같은 속성을 설정해 해당 스크립트가 모듈이란 걸 브라우저가 알 수 있게 해줘야 합니다.

아래와 같이 말이죠.

```
// say.js
export function sayHi(user) {
  return `Hello, ${user}!`;
}

// index.html
<!doctype html>
<script type="module">
  import {sayHi} from './say.js';

  document.body.innerHTML = sayHi('John');
</script>

// 결과
Hello, John!

// 브라우저가 자동으로 모듈을 가져오고 평가한 다음, 이를 실행한 것을 확인할 수 있습니다.
```



**모듈은 로컬 파일에서 동작하지 않고, HTTP 또는 HTTPS 프로토콜을 통해서만 동작합니다.**

로컬에서 `file://` 프로토콜을 사용해 웹페이지를 열면 `import`, `export` 지시자가 동작하지 않습니다. 예시를 실행하려면 로컬 웹 서버인 static-server나, 코드 에디터의 '라이브 서버' 익스텐션(Visual Studio Code 에디터의 경우 Live Server Extension)을 사용하세요.

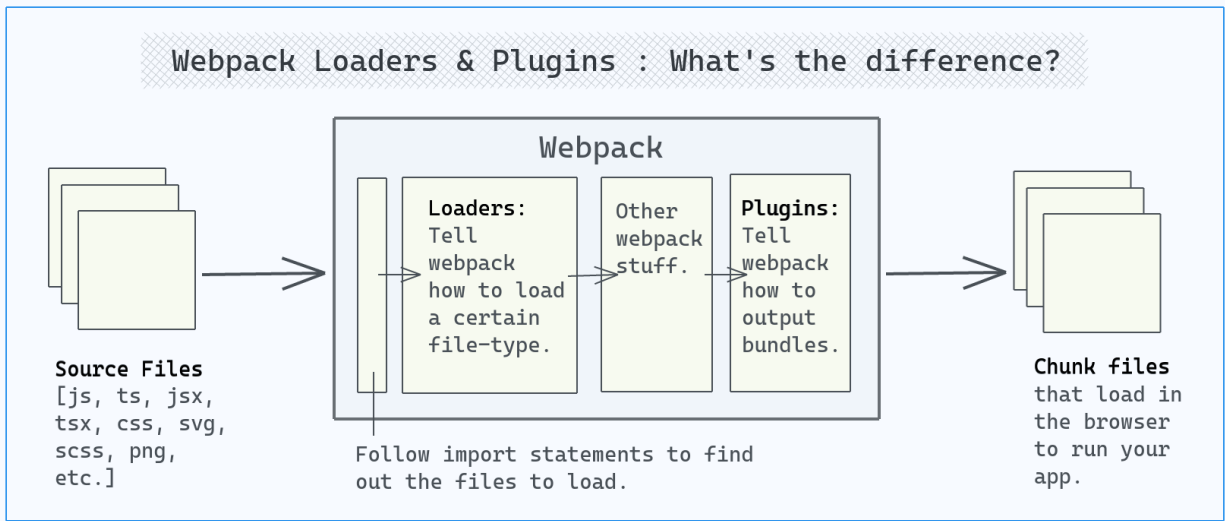
## 바벨

바벨은 최신 자바스크립트 코드를 구형 브라우저에서도 실행 가능한 호환성 있는 코드로 변환해주는 자바스크립트 컴파일러입니다. 바벨은 화살표 함수, 템플릿 리터럴, 구조 분해 할당과 같은 최신 자바스크립트 기능을 지원하며, 이를 ES5와 같은 구형 브라우저에서도 실행 가능한 호환성 있는 코드로 변환합니다.

- 자바스크립트를 보다 오래된 환경에서 동작할 수 있는 버전으로 컴파일



- 🐞 공식 사이트 보기
- 기타 방법: 타입스크립트 컴파일러 사용



## ▼ 프로젝트

### 1. 🔗 사이트에서 체험해보기

**TARGETS** 를 아래로 설정

ie 11

### 아래의 코드들 붙여넣어보기

```
const x = 1; let y = 2;
```

```
const add = (x, y) => x + y;
```

```
const { length } = [1, 2, 3];
```

```
class Bird { wings = 2 } const 새둘이 = new Bird();
```

### 2. 웹팩 프로젝트에 적용해보기

#### 1. 프로젝트에 관련 모듈 설치

```
npm install --save-dev babel-loader @babel/core @babel/preset-env
```

#### 2. 웹팩에 설정 추가

webpack.config.js 에 아래의 프로퍼티들 추가

```
target: ['web', 'es5'], // ☆ ES5 이하로 해야 할 시 필요  module: {      rules: [      {  
test: /\.m?js$/,          exclude: /node_modules/,      use: {      loader:  
'babel-loader',          options: {      presets: [  
['@babel/preset-env', { targets: "ie 11" }]]      ]      }      }      }  
]      }
```

### 3. 빌드하고 결과 확인

npm run build

- 위 명령어로 빌드 실행
- 💡 ./dist/main.js 파일 확인 - javascript beautifier 사이트들에서 살펴보기

결론적으로, 웹팩, 모듈, 바벨은 현대적인 웹 개발에서 필수적인 도구입니다. 이들은 유지보수 가능하고 확장 가능하며 최적화된 코드를 작성할 수 있게 도와주며, 다양한 브라우저와 기기에서 실행 가능하게 만들어줍니다. 이 외에도 다른 도구들이 있지만, 웹 개발자들 사이에서 웹팩, 모듈, 바벨은 가장 인기 있는 도구 중 하나입니다.

출처

<https://www.yalco.kr/@javascript-abyss/15-3/>

#### 웹팩과 바벨

어려운 프로그래밍 개념들을 쉽게 설명해주는 유튜브 채널 '알파한 코딩사전'. 영상에서 다 알려주지 못한 정보들이나 자주 묻는 질문들의 답변들, 예제 코드들을 알코에서 확인하세요!

 <https://www.yalco.kr/@javascript-abyss/15-3/>

<https://ko.javascript.info/modules-intro#:~:text=개발하는 애플리케이션의 크기,구성된 라이브러리 하나로 구성됩니다.>

<https://www.tezify.com/post/babel-and-webpack-fundamentals/>

## Babel & Webpack Fundamentals

How the front-end code we write ends up being executed when visitors use our web-app

<https://www.tezify.com/post/babel-and-webpack-fundamentals/>

