ES6 문법 정리

참고문헌 ;

1. [https://jsdev.kr/t/es6/2944](https://jsdev.kr/t/es6/2944" \t "_blank)
2. [http://woowabros.github.io/experience/2017/12/01/es6-experience.html](http://woowabros.github.io/experience/2017/12/01/es6-experience.html" \t "_blank)
3. <http://es6-features.org/#Constants>
4. <https://blog.asamaru.net/2017/08/14/top-10-es6-features/>
5. [https://babeljs.io/docs/en/index.html ES6](https://babeljs.io/docs/en/index.html%20ES6)사용을 위한 babel사용법

넷스케이프 – 초기 오픈소스 웹 브라우저,  [HTML](https://namu.wiki/w/HTML)과 [CSS](https://namu.wiki/w/CSS) 등은 지원하지 못한다. 자바스크립트를 지원한다. 자바스크립트를 개발했다!

ECMA스크립트 -  [자바스크립트](https://ko.wikipedia.org/wiki/%EC%9E%90%EB%B0%94%EC%8A%A4%ED%81%AC%EB%A6%BD%ED%8A%B8)(대부분 웹브라우저에서 사용)를 표준화하기 위해 만들어졌다. 액션스크립트와 j스크립트 등 다른 구현체도 포함한다.  [웹](https://ko.wikipedia.org/wiki/%EC%9B%94%EB%93%9C_%EC%99%80%EC%9D%B4%EB%93%9C_%EC%9B%B9)의 클라이언트 사이드 스크립트로 많이 사용되며 [Node.js](https://ko.wikipedia.org/wiki/Node.js)를 사용한 서버 응용 프로그램 및 서비스에도 점차 많이 쓰이고 있다. 서로 다른 스크립트끼리 만들어져도 호환이 가능하게 하기위해(표준화하기 위해)만들어졌다.

* 부가설명 : 웹브라우저에서 동적인 기능을 제공하기 위한 언어가 자바스크립트 언어 > 표준 규격없이 여러 브라우저마다 독자적인 특성 추가 == JS규모가 커지면서 중복되는 기능들을 하나의 모듈로 만들어 소스코드의 규모를 줄이고, 재사용성을 좋게하였다. 하지만 모듈 사용에 대한 표준이 없었기 때문에 > 개발자들 간의 혼선 발생 > 언어 사용성 저하 > ECMA스크립트라는 표준을 만들게됨 > ES6부터는 모듈 포맷을 내장하여 제공하기 시작하였다.

클라이언트 사이드 스크립트 – 클라이언트-서버구조에서 클라이언트 쪽에서 실행되는 스크립트언어

ECMAScript2015 == ES6문법 == Ecmascript6

[ES6을 사용하는 법]

-모든 브라우저에서 완전하게 지원되지 않으므로 webpack과 같은 번들러를 사용해야한다

> 번들러 : . 여러 JS파일을 하나의 JS파일로 만들

> webpack : JS 모듈화 명세를 만드는 대표적인 작업 그룹인 CommonJS와 AMD를 모두 지원하는 JS 모듈화 도구

> webpack에 대한 더 자세한 설명 : <https://d2.naver.com/helloworld/0239818>

> CommonJS : JS를 브라우저를 넘어선 일반적인 범용 언어로 사용할 수 있도록 조직한 워킹 그룹

> AMD : CommonJS에서 독립하여 비동기(필요한 모듈을 네트워크를 통해 내려받을 수 있도록 하는 것)에 대한 표준안을 다루고 있다.

> CommonJS와 AMD에 대한 자세한 설명 : <https://d2.naver.com/helloworld/12864>

> babel : ES6코드를 이전 버전과 호환되는 javascript버전으로 변환하는데 사용하는 패키지

변수지정방법

Var [변수명] = [변수명] || [넣을 값]

[ES6에 새로 추가된 기능들]

|  |  |  |  |
| --- | --- | --- | --- |
| Default Parameters | 함수에 넘겨주는 인자값에 대한 default 처리가 간소화됨 | Var student = function(height, name, url) { var height = height || 50 var name = name || 'red' var url = url || '[http://azat.co](http://azat.co/)' } | var student = function(height = 50, color = 'red', url = '[http://azat.co](http://azat.co/)') { } |
| Destructuring Assignment |  |  |  |
| arrows Functions |  |  |  |
| classes |  |  |  |
| Enhanced object literals |  |  |  |
| Template Literals | 문자열 안에 ${변수명}이라는 구문을 사용하여 문자열 처리 간소화 예시를 보면 변수때문에 띄어쓰기나 점도 따로 쓸 필요없어서 매우 편해진 것을 알 수 있다! 문자열을 묶는 기호가 " ' " 에서 " ` "(back-ticked)로 바뀌었다. | var name = 'Your name is '+first+' '+last+'.' var url = 'http://localhost/api/massages/'+id | var name = `Your name is ${first} ${last}.` var url = `[http://localhost/api/massages/${id}](http://localhost/api/massages/$%7Bid%7D) ` |
| template strings |  |  |  |
| Multi-line Strings | 멀티 라인 문자열도 한 문자열 안에서 처리가 가능해졌다. 문자열을 묶는 기호가 " ' " 에서 " ` "(back-ticked)로 바뀌었다. | var week = 'Monday \n\t' +'Tuesday \n\t' +'Wednesday\n\t' +'Thursday\n\t' +'Friday\n\t' var fourAgreements = 'You have the right to be you. \n\ You can only be you when you do your best.' | var week =` Monday Tuesday Wednesday Thursday Friday` var fourAgreements =` You have the right to be you. You can only be you when you do your best.` |
| destructuring |  |  |  |
| default + rest + spread |  |  |  |
| Block-Scoped Constructs Let and Const |  |  |  |
| iterators + for…of |  |  |  |
| generators |  |  |  |
| unicode |  |  |  |
| modules |  |  |  |
| module loaders |  |  |  |
| map + set + weakmap + weakset |  |  |  |
| proxies |  |  |  |
| symbols |  |  |  |
| subclassable built-ins |  |  |  |
| promises |  |  |  |
| math + number + string + array + object APIs |  |  |  |
| binary and octal literals |  |  |  |
| reflect api |  |  |  |
| tail calls |  |  |  |

다음에 읽어볼 부분

Ecmascript가 어떻게 업데이트되었나 설명된 사이트? <https://www.ecma-international.org/ecma-262/6.0/>

아직 ECMA스크립트가 무엇인지 개념이해 부족

기능들 참고1,2 이용해서 내용채우기, 겹치는 부분이 있는지도 확인!

기능 다 읽어보고 3번을 통해서 완전히 이해하기