11. Javascript Engine

ellapresso

자바스크립트 엔진이란?

JS 코드를 실행하는 프로그램 또는 인터프리터

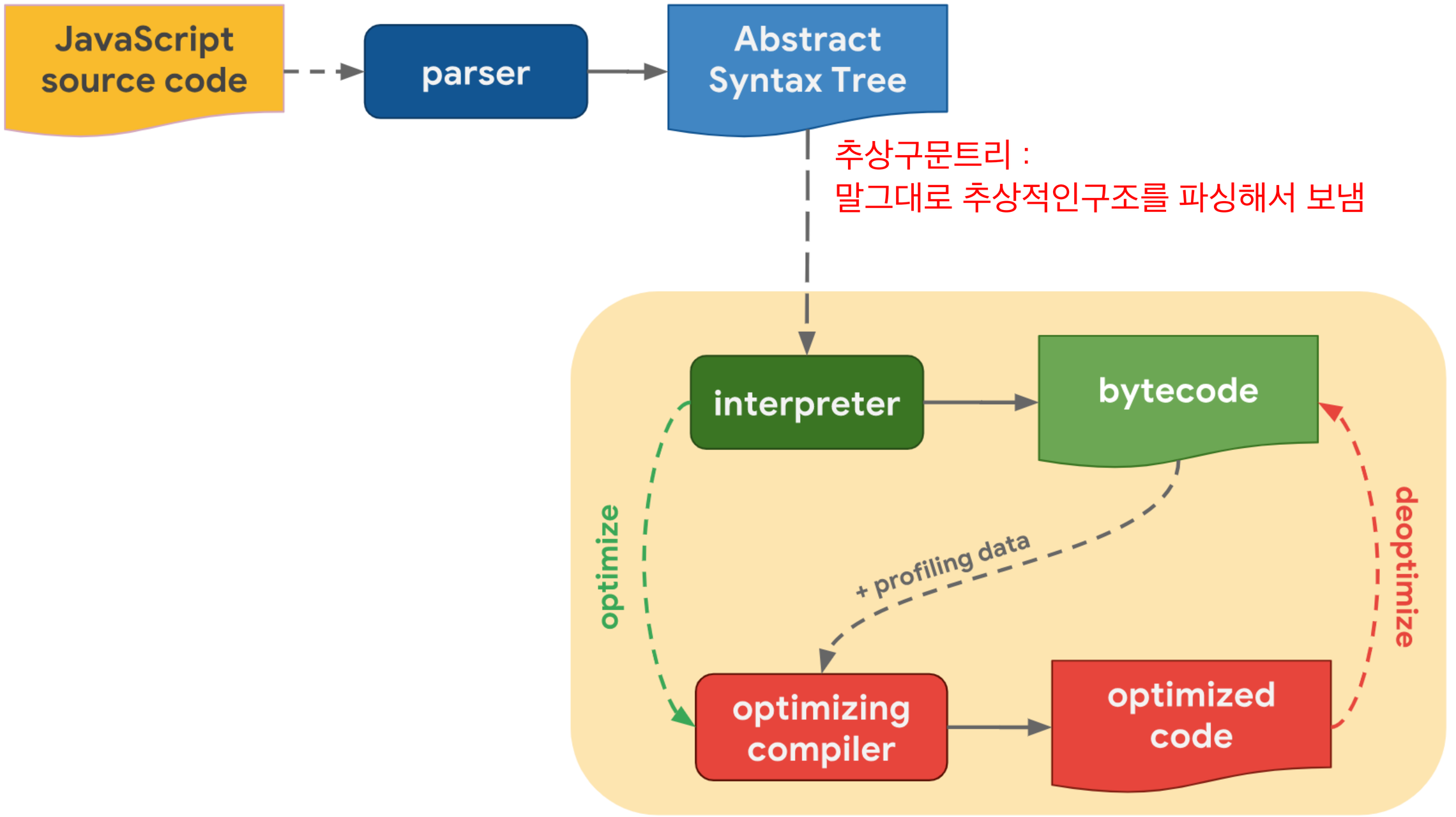
인터프리터 vs 컴파일러

인터프리터 : 고급언어 => 중간언어 => 실행

컴파일러 : 고급언어 => 컴파일(오래걸릴수 있음) => 기계어 (실행이 빠름)

자바스크립트 엔진 종류

- **V8 (:: Crome / Nodejs)**
- Spider Monkey (:: Firefox)
- Chakra (:: Microsoft Edge)
- Webkit
- Javascript Core (:: Safari)
- Rhino
- JerryScript

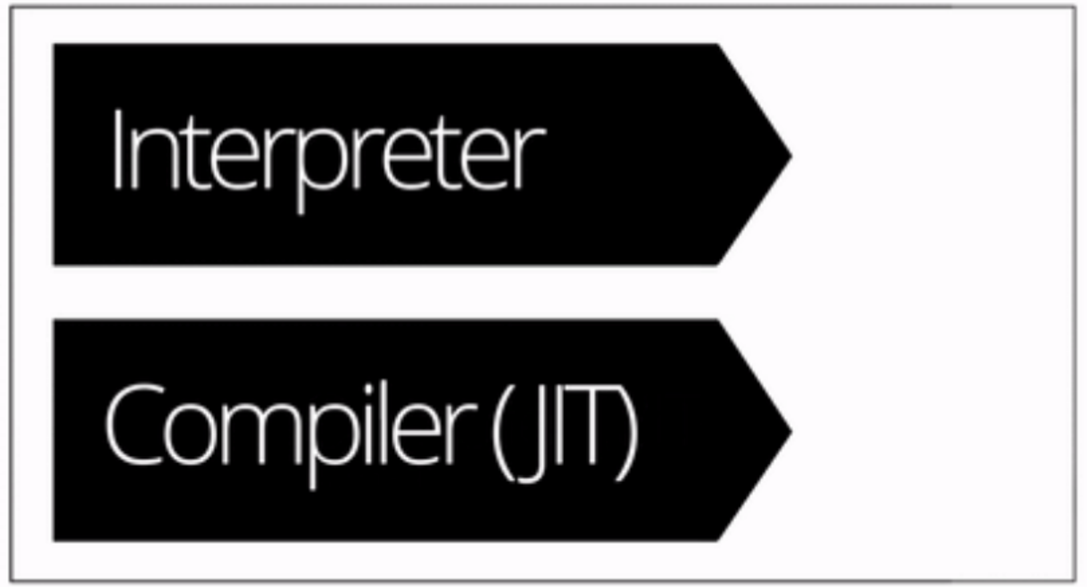

JavaScript source code
parser
Abstract Syntax Tree
추상구문트리 :
말그대로 추상적인구조를 파싱해서 보냄
interpreter
bytecode
optimize
+ profiling data
deoptimize
optimizing compiler
optimized code

Source Code

```
cons getName = (person) => person.lastname;

cons han = {firstname: "Han", lastname: "Solo"};
cons luke = {firstname: "Luke", lastname: "Skywalker"};
cons leia = {firstname: "Leia", lastname: "Organa"};
cons obi = {firstname: "Obi-Wan", lastname: "Kenobi"};

cons persons = [han, luke, leia, obi];

for(var i = 0; i < 1000 * 1000 * 1000; i++) {
  getName(persons[i & 3]);
}
```

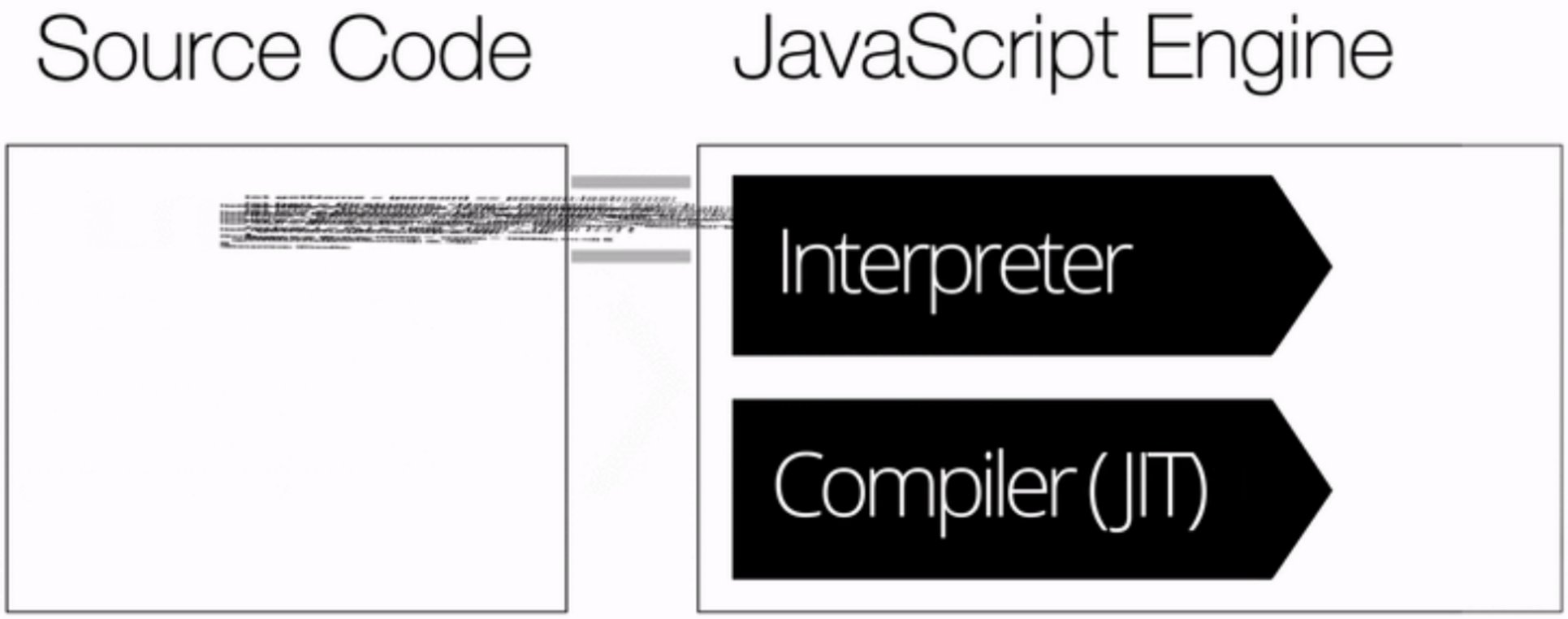
© Rainer Hahnekamp

JavaScript Engine



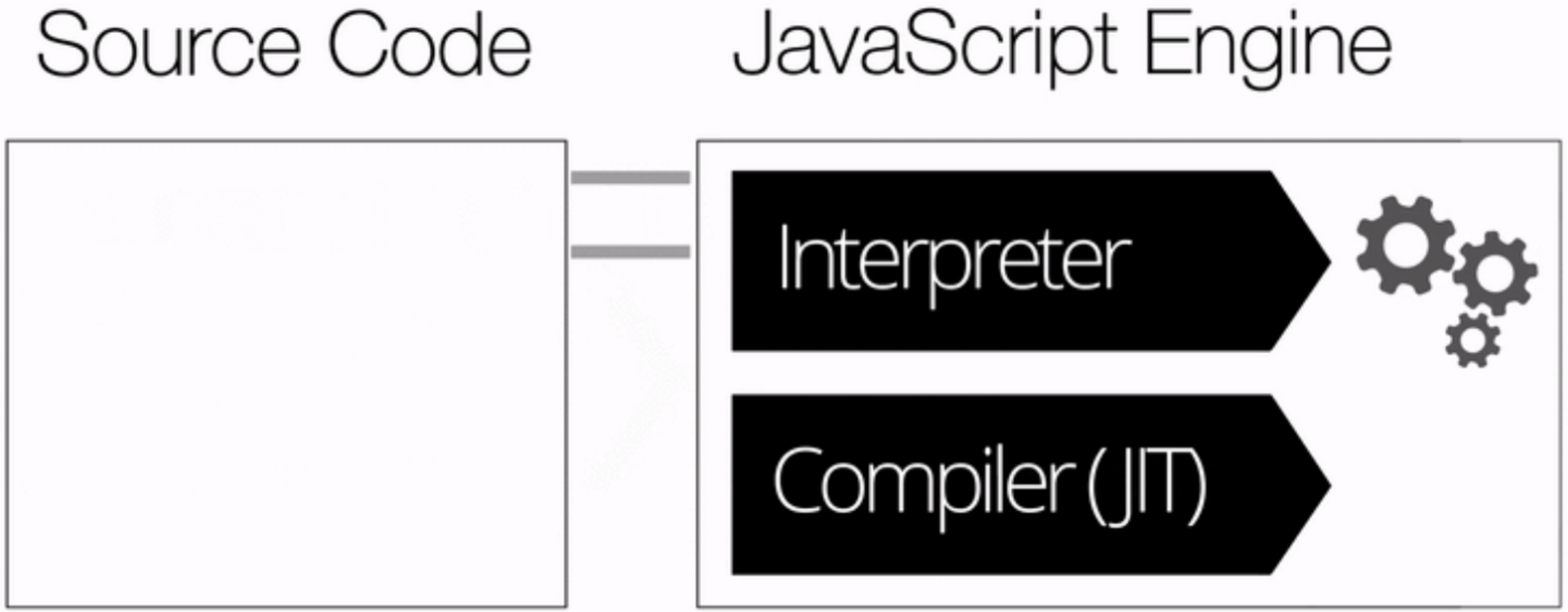
재고 비용을 최소화하기 위해 구성 요소가
필요한 바로 전에 전달되는 제조 시스템을 말함

1. Engine loads Source Code



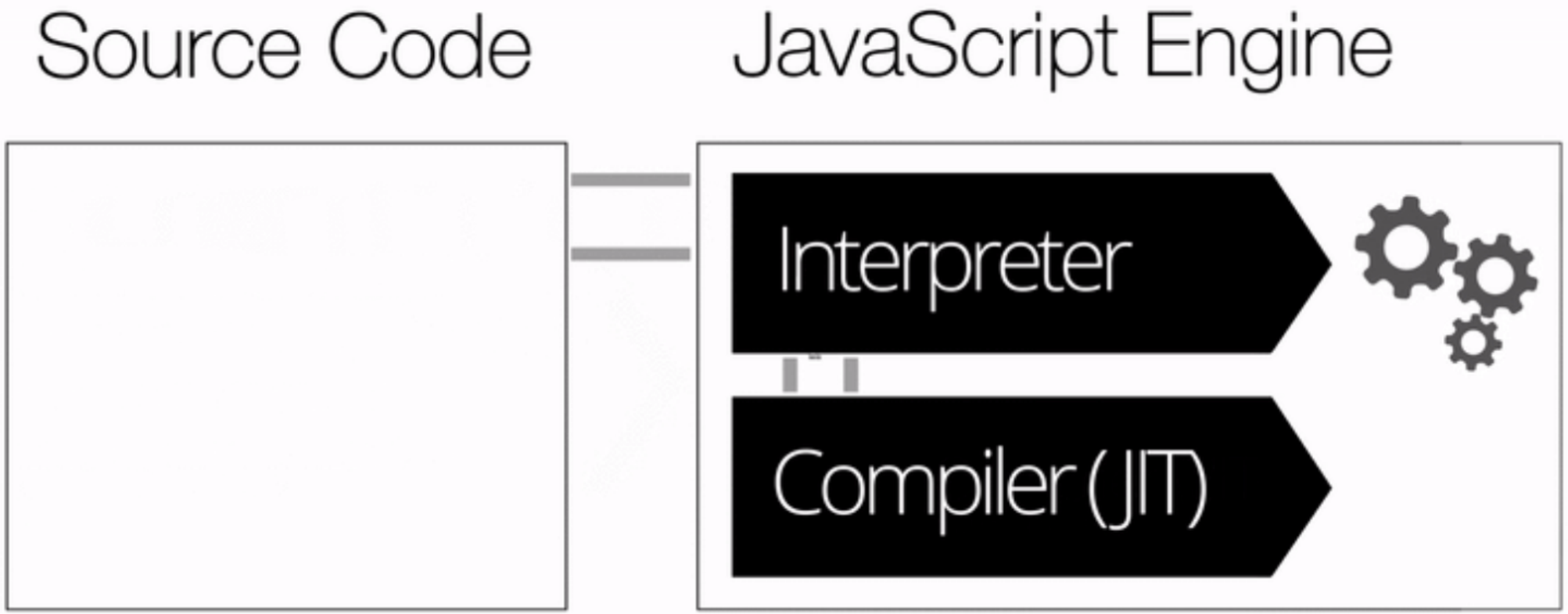
2. Interpreter starts the application

바이트코드(중간언어)를 빠르게 생산(최적화x)

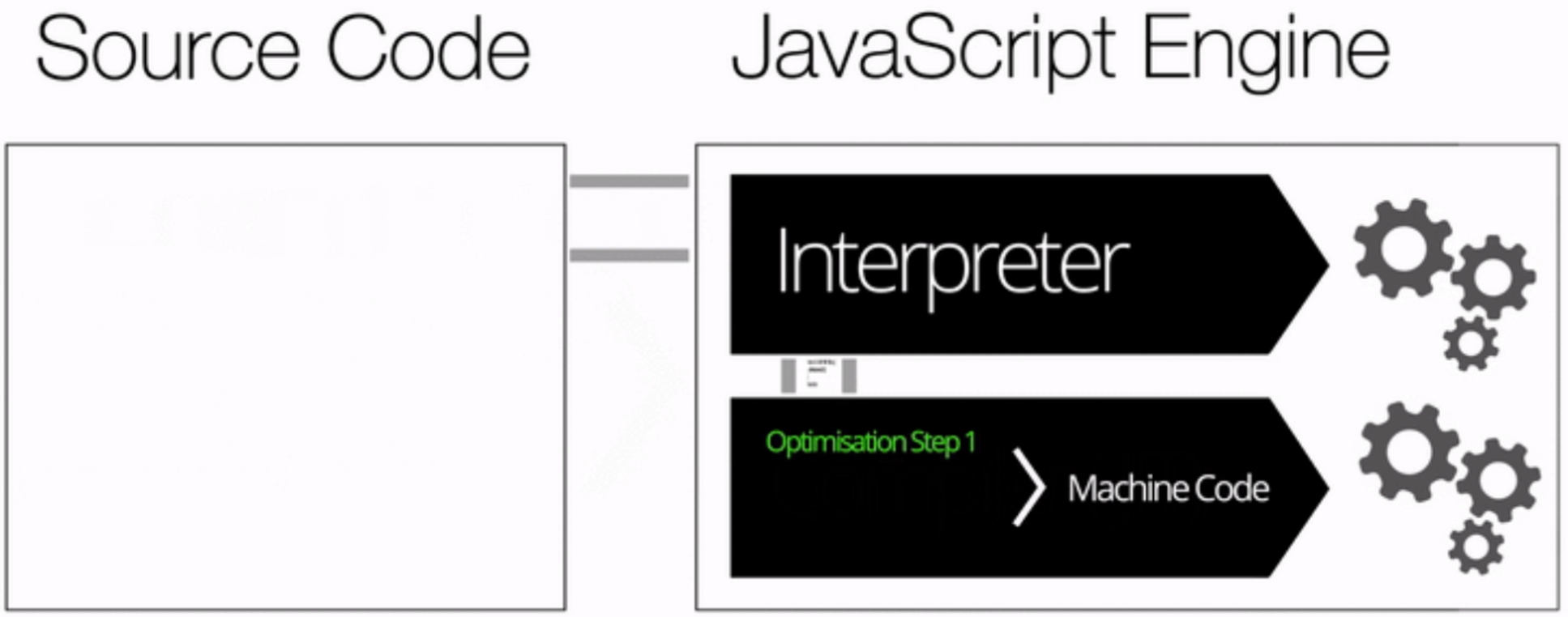


3. Compiler receives code: "Hot Path" first

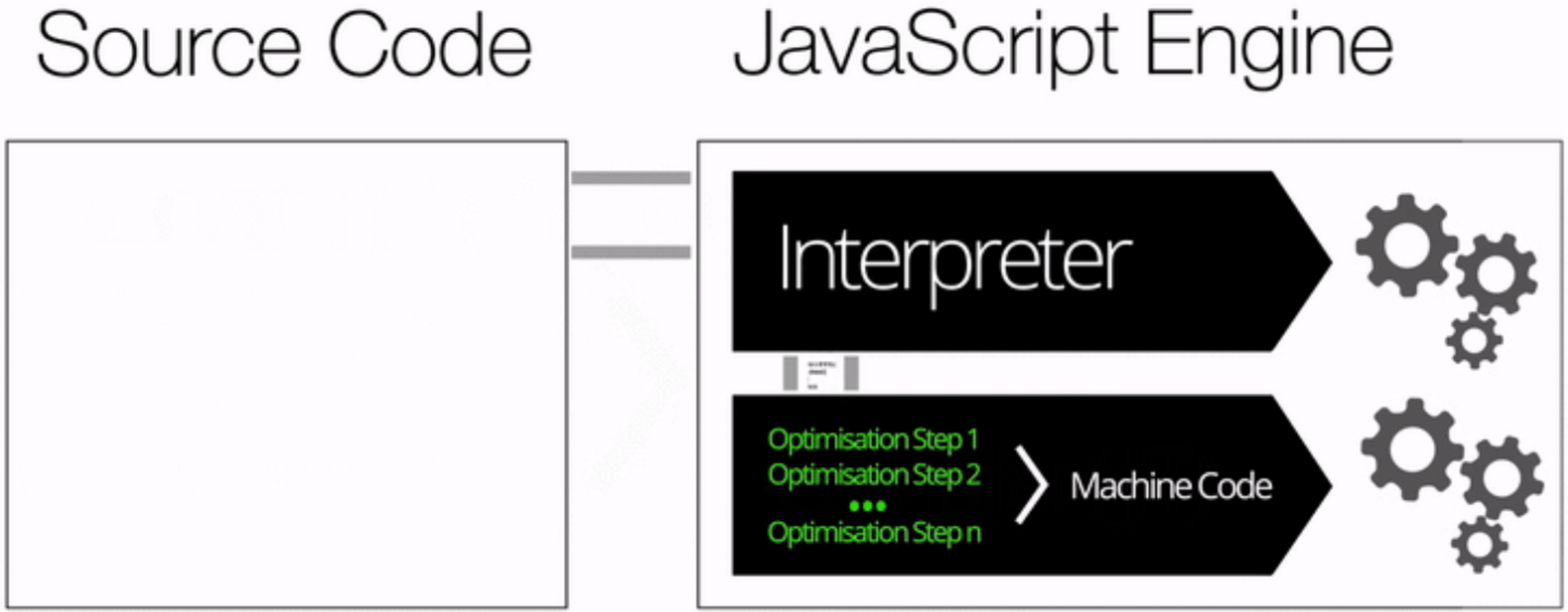
자주 실행되는 부분을 먼저 최적화 컴파일러로!



4. Compiler starts optimisation and compilation



5. Compiler incrementally optimises the code



V8은 조금 다르다?

인터프리터 없이 바로 기계어로 컴파일

두가지의 컴파일러를 사용함.

- 전체 컴파일러
- 최적 컴파일러

V8 - 전체 컴파일러

모든 코드를 기계어로 변환

런타임시 데이터 타입이 변경될수 있으므로 건드리지 않는다.

- 인라인 캐싱으로 런타임시 즉석해서 처리함

- 여러 타입을 처리할 경우 성능 저하

V8 - 최적 컴파일러

전체 컴파일러와 병렬적으로 처리

자주 호출되는 함수를 재 컴파일 함.

- 인라인 캐싱으로 얻은 정보로 최적화함

인라인캐싱 !

자바스크립트 엔진의 최적화 철학이 잘 담겨있다 !

루프 안에서 필드 접근을 하려는 객체가 같은 hidden class를 가리키고 있어야 함.

루프 안에서, 루프 중간에 객체의 필드 구조를 바꾸게 되면 좋지 않지만

실제로 이렇게 구현할 일은 거의 없을 것…

그러므로 배열 내의 객체들을 반복해서 접근할 때,

하나의 배열 안에는 모두 같은 필드 구조를 가지는 (같은 hidden class를 가지는)

객체들만 넣어서 접근하도록 해 주어야 함.

참고 사이트

https://devtimothy.tistory.com/94

https://sjh836.tistory.com/92

https://meetup.toast.com/posts/78