**Tokens** are the smallest individual words, phrases, or characters that **JavaScript** can understand. When **JavaScript** is interpreted, the browser parses the script into these **tokens** while ignoring comments and white space.

Automatic semicolon insertion (ASI) rules:

Rule #1: when reading tokens of a program from left to right, a token that doesn’t match the grammar rule has a semicolon inserted before it if either of the following conditions are met:

1. The error token is separated from the previous by at least one line terminator
2. The error token is a }

Watch out for lines beginning with [ or (

Rule #2: if the program is parsed until the end of the input and it’s not yet a complete program (ie there were no outright errors that would have caused an exception to throw before reaching the end of the tokens) a semicolon is appended

Rule #3: if a line terminator is encountered with a “restricted production” ASI will try to save you by adding a semicolon before the line terminator is encountered

ASI is not implied in:

Head of a For Loop

Statements appearing on the same line

Some statements don’t need to be terminated by a semicolon and won’t trigger ASI

* If you are going to write JS without option semicolons, make sure you understand what ASI is actually doing
* Minification, compression, etc. on otherwise valid JS code could cause unforeseen errors if your code doesn’t use semicolons
* ASI has the potential to make your invalid JS code valid gibberish, making it harder to debug since it (sometimes) doesn’t have an outright syntax error
* Performance & file size aren’t significantly affected by semicolons or lack thereof

Why understand ASI?

Prepare to contribute to open source projects

Some companies omit option semicolons (Github, Twitter-Bootstrap)

Poor syntax = triggers ASI = hard to debug

Using semicolons doesn’t absolve you from “ASI creep”

**Babeljs.io - shows transpilation of ES6 (and ES5) + shows ASI**

**JSLint**

**JSHint**