Modul praktikum - Minggu 03 - Struktur leksikal

Dosen pengampu: **Henokh Lugo Hariyanto**
Asisten mata kuliah: **Jein Ananda - (10221031)**; **Muhammad Aulia Rahman - (10221055)**

Tujuan:

- Mampu memahami struktur leksikal dalam bahasa pemrograman JavaScript.
- Mampu menggunakan struktur leksikal seperti: *comments*, *literals*, *identifier*, *reserved words*, *unicode*.
- Mampu memahami penting tidaknya menggunakan semicolon di akhir baris.

> Tips belajar bahasa pemrograman adalah mengetik ulang perintah yang kita temukan di buku atau di internet, lalu kita ubah-ubah untuk menguji pemahaman kita sudah tepat atau belum. Faktor bermain-main dan eksplorasi sangat diperlukan untuk memahami setiap perintah bahasa pemrograman yang kita pelajari. Setiap potongan kode di bawah dapat ditulis dalam berkas `.js` lalu dapat di-*running* dengan Node.js.

Struktur leksikal adalah suatu kumpulan aturan dasar yang mengatur tata cara menuliskan program dengan bahasa yang kita pilih, misalnya disini struktur leksikal yang akan kita bahas adalah JavaScript.

Struktur leksikal mengatur sintaks (tata cara penulisan) bahasa JavaScript. terkait penamaan variable, karakter *delimiters* untuk *comments*, dan tata cara memisahkan satu *statement* dengan *statement* yang lain.

Ada 6 hal yang akan kita coba untuk pelajari secara langsung melalui penulisan kode JavaScript

- Sensitivitas penamaan variabel, aturan spasi antar sintaks, dan spasi antar baris
- *Comments*
- Literal
- *Identifiers* dan *Reserved words*
- Unicode
- Penggunaan titik koma (*semicolon*)

Case sentivity, *spaces*, dan *line breaks*

- Huruf besar kecil menentukan pembeda antara nama variables:
  case-sensitive.js

```
let Hello = 1;
let hello = 2;
let heLLo = 3;
let HELLO = 4;
console.log(Hello, hello, heLLo, HELLO)
```

- Spasi antara token (nama variable, tanda operator dan nilai variable) dapat diabaikan
  ignored-space.js

```
let hello=1;
console.log(hello);

hello =        5;
console.log(hello);
```

- *Line breaks* dapat diabaikan dalam penulisan kode JavaScript asalkan dipisahkan dengan titik koma (*semicolon*)
  ignored-linebreaks.js

```
let hello = 1; world = 5; console.log(hello, world);
```

Comments

Pembuatan komentar sangat perlu untuk memberitahu *programmer* seperti apa jalannya program yang dia buat.

- Komentar satu baris dapat menggunakan *double slashes*
- Komentar untuk baris untuk lebih dari satu baris menggunakan pasangan buka tutup *slashes and stars*: `/* */`

comment-in-js.js

```
// Ini merupakan komentar satu baris

/*
 * Ini merupakan komentar lebih dari satu baris. Ekstra satu karakter *
 * diawal bari tidak harus ditulis. Itu ditulis hanya untuk terlihat
 * lebih konsisten dan menarik.
 */
```

Literals

- *Literals* adalah suatu nilai data yang ditulis secara langsung menggunakan karakter seperti contohnya karakter angka "2", akan memberikan bilangan bulat 2 apabila diberikan pada suatu variabel. Berikut beberapa contoh *literals* dan penggunaannya. (literal disini diberikan sebagai nilai sebelah kanan tanda "=").
  literals.js

```
let int_num = 12;             // Bilangan bulat dua belas
let dec_num = 1.2;            // Bilangan desimal 1.2
let str_1 = "hello world!";   // string "hello world"
let str_2 = 'Hi';             // string dengan delimiter tanda petik satu
let bool_1 = true;            // Nilai boolean true
```

```
  let bool_2 = false;           // Nilai boolean false;
  let null_obj = null;          // nilai untuk ketidakadaan tipe data objek
```

Identifiers dan *Reserved Words*

- *Identifier* berarti nama. *identifier* biasanya digunakan untuk menamakan nilai konstanta, *variable*, *properties*, *class*, dan label untuk iterasi dalam perulangan.
  Identifier dalam JavaScript harus diawali oleh suatu huruf (bukan angka), garis bawah (*underscore*) atau tanda dollar (*dollar sign*). Huruf kedua dan selanjutnya boleh angka, garis bawah, atau tanda dollar.
  identifiers-reserved.js

```
  // Berikut ini adalah identifier yang legal
  let i = 0;                        // i adalah identifier
  let my_variable_name = "name";    // tanda garis bawah setelah huruf
pertama
  let v13 = 13.3;                   // angka setelah huruf pertama
  let _dummy = "dummy";             // boleh diawali tanda garis bawah
  let $str = 'new string';          // boleh diawali oleh tanda dollar
  let new$sign = "rupiah";          // tanda dolar boleh setelah huruf
pertama

  console.log(i, my_variable_name, v13, _dummy, $str, new$sign);
```

  Reserved words merupakan kata-kata yang sudah dipakai oleh JavaScript untuk menamakan beberapa prosedur dan tidak bisa digunakan untuk penamaan variable. Berikut adalah daftar *reserved words* yang umum digunakan oleh JavaScript.

```
  as          const       export    get         null      target      void
  async       continue    extends   if          of        this        while
  await       debugger    false     import      return    throw       with
  break       default     finally   in          set       true        yield
  case        delete      for       instanceof  static    try         catch
  do          from        let       super       typeof    class       else
  function    new         switch    var
```

Unicode

Unicode character adalah daftar kode untuk representasi huruf yang lebih luas. Contohnya seperti huruf-huruf jawa, emoji, latin, jepang, mandarin, dst.
Di dalam JavaScript dapat menggunakan karakter Unicode sebagai *identifier*, namun tidak disarankan secara *programming convention* untuk urusan portability, artinya program diusahakan sebisa mungkin dapat dipahami oleh banyak orang dengan berbagai latar belakang bahasa dan abjad yang digunakan. Namun untuk nilai dari *identifiers* boleh menggunakan karakter Unicode; Berikut contoh penggunaan Unicode:
unicode.js

```
const π = 3.14;
const истинный = true;
const emoji = "🦉";           // unicode emoji sebagai value
const javanese_nya = "ꦲꦩ";    // unicode aksara jawa sebagai value
const otherworld = "異世界";    // unicode kanji sebagai value
console.log(π, истинный, emoji, javanese_nya, otherworld);

// pemanggilan karakter Unicode sebagai idenfier bisa menggunakan cara
// sebagai berikut
let café = 1;   // mendefinisikan variable dengan nama identifier memuat unicode
console.log(caf\u00e9, caf\u{E9})  // dua cara untuk memanggil identifier café"
```

Optional semicolons

Pada bagian pertama, kita sudah membahas bahwa kita dapat mengabaikan *line breaks* dengan cara menggunakan titik koma (*semicolon*).
Namun kita sebenarnya dapat juga sebaliknya mengabaikan *semicolon*, namun kita perlu menambahkan *line breaks*. Sebagai contoh:

ignored-semicolon.js

```
let a
a
=
3
console.log(a);
```

Kode di atas setara dengan satu baris kode berikut:

```
let a; a = 3; console.log(a);
```

Tugas

- Silahkan berdiskusi untuk proyek tugas akhir yang ingin kalian dengan anggota satu kelompok. Tugas ini tidak perlu dikumpul cukup membuat progress di Google Sheet (format bebas) tiap minggu apa saja yang dikerjakan untuk mencapai penyelesaian final project.