**MODULE PEMBELAJARAN JAVASCRIPT LANJUTAN**

**ES6**

1. **Object**

JavaScript adalah bahasa berbasis objek dan dalam JavaScript hampir semuanya adalah objek atau bertindak seperti objek. Jadi, untuk bekerja dengan JavaScript secara efektif dan efisien, kita perlu memahami cara kerja objek serta cara membuat objek kalian sendiri dan menggunakannya.

Objek JavaScript hanyalah kumpulan nilai yang diberi nama. Nilai-nilai ini biasanya disebut sebagai properti dari objek. Jika kalian ingat dari bab array JavaScript, array adalah kumpulan nilai, di mana setiap nilai memiliki indeks (kunci numerik) yang dimulai dari nol dan bertambah satu demi satu untuk setiap nilai. Objek mirip dengan array, tetapi perbedaannya adalah kalian menentukan sendiri kunci, seperti nama, usia, jenis kelamin, dan sebagainya. Di bagian berikut kita akan belajar tentang objek secara detail.

**Membuat Object**

Objek dapat dibuat dengan kurung kurawal **{}** dengan daftar properti opsional. Properti adalah pasangan “key: value”, di mana kunci (atau ***nama properti***) selalu berupa string, dan nilai (atau ***nilai properti***) dapat berupa tipe data apa pun, seperti string, angka, Boolean, atau tipe data kompleks seperti array, fungsi, dan objek lainnya. Selain itu, properti dengan fungsi sebagai nilainya sering disebut metode untuk membedakannya dari properti lainnya. Objek JavaScript yang umum akan terlihat seperti ini:

**var** person = {

    name: "Peter",

    age: 28,

    gender: "Male",

    displayName: **function**() {

        alert(**this**.name);

    }

};

Contoh di atas membuat objek yang disebut **person** yang memiliki tiga properti yaitu **name, age**, dan **gender** dan satu metode **displayName()**. Metode **displayName()** menampilkan nilai **this.name**, yang diputuskan menjadi **person.name**. Ini adalah cara termudah dan disukai untuk membuat objek baru dalam JavaScript, yang dikenal sebagai sintaks object literals.

Nama-nama properti umumnya tidak perlu dikutip kecuali kata-kata yang dicadangkan, atau jika mengandung spasi atau karakter khusus (apa pun selain huruf, angka, dan **\_** dan **$** karakter), atau jika mereka mulai dengan angka, seperti yang ditunjukkan dalam contoh berikut:

**var** person = {

    "first name": "Peter",

    "current age": 28,

    gender: "Male"

};

##### Mengakses Properti Obyek

Untuk mengakses atau mendapatkan nilai properti, kalian dapat menggunakan notasi titik **(.)**, Atau braket persegi **([])**, seperti yang ditunjukkan dalam contoh berikut:

**var** book = {

    "name": "Harry Potter and the Goblet of Fire",

    "author": "J. K. Rowling",

    "year": 2000

};

// Dot notation

document.write(book.author);  // Prints: J. K. Rowling

// Bracket notation

document.write(book["year"]); // Prints: 2000

Notasi titik (dot) lebih mudah dibaca dan ditulis, tetapi hal itu tidak selalu dapat digunakan. Jika nama properti tidak valid (mis. Jika berisi spasi atau karakter khusus), maka kalian tidak dapat menggunakan notasi titik; kalian harus menggunakan notasi braket, seperti yang ditunjukkan pada contoh berikut:

**var** book = {

    name: "Harry Potter and the Goblet of Fire",

    author: "J. K. Rowling",

    "publication date": "8 July 2000"

};

// Bracket notation

document.write(book["publication date"]); // Prints: 8 July 2000

Notasi braket persegi (bracket notation) menawarkan lebih banyak fleksibilitas daripada notasi titik. Ini juga memungkinkan kalian untuk menentukan nama properti sebagai variabel alih-alih hanya string literal, seperti yang ditunjukkan pada contoh di bawah ini:

|  |
| --- |
| **var** person = {      name: "Peter",      age: 28,      gender: "Male"  };    **var** key = prompt("Enter any property name to get its value");  alert(person[key]); // Outputs: Peter (if enter "name") |

**Looping Melalui Properti Object**

Kalian dapat mengulangi (iterate) melalui pasangan key-value suatu objek menggunakan ***for…in*** loop. Loop ini secara khusus dioptimalkan untuk mengulangi properti objek. Berikut contohnya:

**var** person = {

    name: "Peter",

    age: 28,

    gender: "Male"

};

// Iterating over object properties

**for**(**var** i **in** person) {

    document.write(person[i] + "<br>"); // Prints: name, age and gender

}

##### Mengatur Properti Obyek

Demikian pula, kalian dapat mengatur properti baru atau memperbarui yang ada menggunakan notasi titik **(.)** Atau braket **([])**, seperti yang ditunjukkan dalam contoh berikut:

**var** person = {

    name: "Peter",

    age: 28,

    gender: "Male"

};

// Setting a new property

person.country = "United States";

document.write(person.country); // Prints: United States

person["email"] = "peterparker@mail.com";

document.write(person.email); // Prints: peterparker@mail.com

// Updating existing property

person.age = 30;

document.write(person.age); // Prints: 30

person["name"] = "Peter Parker";

document.write(person.name); // Prints: Peter Parker

##### Menghapus Properti Object

Operator **delete** dapat digunakan untuk sepenuhnya menghilangkan properti dari objek. Menghapus adalah satu-satunya cara untuk benar-benar menghilangkan properti dari objek. Mengatur properti menjadi **undefined** atau **null** hanya akan mengubah nilai properti. Itu tidak menghapus properti dari objek.

**var** person = {

    name: "Peter",

    age: 28,

    gender: "Male",

    displayName: **function**() {

        alert(**this**.name);

    }

};

// Deleting property

**delete** person.age;

alert(person.age); // Outputs: undefined

##### Memanggil Metode Object

Kalian dapat mengakses metode objek dengan cara yang sama seperti kalian mengakses properti — menggunakan notasi titik (dot) atau menggunakan notasi kurung siku (bracket). Berikut contohnya:

**var** person = {

    name: "Peter",

    age: 28,

    gender: "Male",

    displayName: **function**() {

        alert(**this**.name);

    }

};

person.displayName(); // Outputs: Peter

person["displayName"](); // Outputs: Peter

##### Memanipulasi Nilai vs. Referensi

Objek JavaScript adalah tipe referensi yang berarti ketika kalian membuat salinannya, kalian benar-benar hanya menyalin referensi ke objek itu. Sedangkan nilai primitif seperti string dan angka ditugaskan atau disalin sebagai nilai keseluruhan. Untuk lebih memahami semua ini, mari kita lihat contoh berikut:

**var** message = "Hello World!";

**var** greet = message; // Assign message variable to a new variable

greet = "Hi, there!";

document.write(message);  // Prints: Hello World!

document.write(greet);  // Prints: Hi, there!

Pada contoh di atas, kami telah membuat salinan variabel **message** dan mengubah nilai salinan itu (mis. Variabel **greet**). Dua variabel tetap berbeda dan terpisah. Tetapi, jika kita melakukan hal yang sama dengan suatu objek, kita akan mendapatkan hasil yang berbeda, seperti yang kalian lihat pada contoh berikut:

|  |
| --- |
| **var** person = {      name: "Peter",      age: 28,      gender: "Male"  };    **var** user = person; // Assign person variable to a new variable  user.name = "Harry";    document.write(person.name);  // Prints: Harry  document.write(user.name);  // Prints: Harry |

Kalian dapat melihat dengan jelas, setiap perubahan yang dilakukan pada variabel ***user*** juga mengubah variabel ***person***; itu terjadi karena kedua variabel referensi objek yang sama. Jadi, hanya dengan menyalin objek tidak akan benar-benar mengkloningnya tetapi hanya menyalin referensi ke objek itu saja.