# Apa itu Pemrograman?

Pemrograman itu seperti memberi tahu komputer apa yang harus dilakukan. Kita menulis “perintah” dalam bentuk kode, dan komputer akan mengikuti perintah itu. Kodenya ditulis dengan cara khusus agar komputer bisa mengerti.

Di pelajaran ini, kita akan belajar bahasa pemrograman yang bernama JavaScript. Dengan JavaScript, kita bisa membuat berbagai macam aplikasi—mulai dari yang sederhana hingga yang rumit!

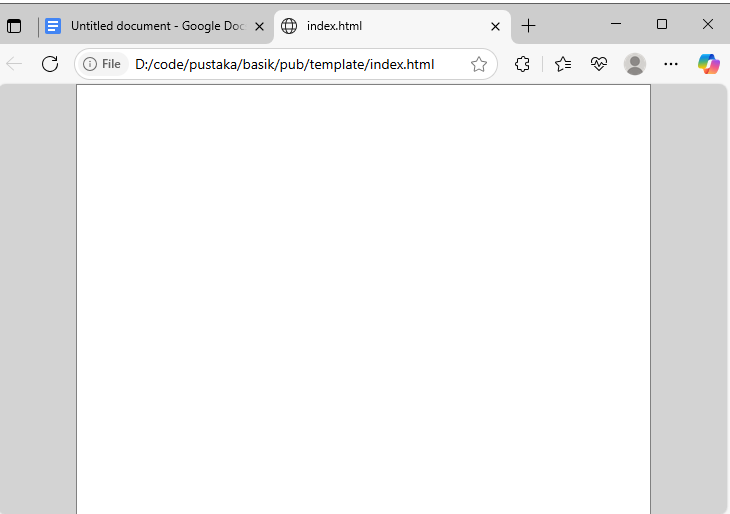
## Latihan 1: Menulis Perintah Sederhana

Mari kita mulai latihan pertama!

Cari folder yang namanya “template”, lalu salin (copy) folder itu. Setelah disalin, beri nama folder barunya: latihan 1.

Buka file index.html Di dalam folder “latihan 1”, ada file yang namanya index.html. Klik dua kali file itu. Komputer akan otomatis membuka file-nya di browser (misalnya Chrome, Edge, atau Firefox), tergantung pengaturan di komputer kamu.

Kalau sudah terbuka, maka akan melihat tampilan awal dari halaman web yang akan kita ubah dan beri perintah.



**Mari kita mulai!**

Sekarang kamu akan melihat layar kosong dengan sebuah kotak di tengah. Kotak ini adalah tempat kita bisa menulis, menggambar, dan bermain dengan kode. Karena belum ada perintah, kotaknya masih kosong.

**Langkah pertama: buka file bernama** index.js

Disarankan untuk membuka file ini dengan aplikasi khusus untuk menulis kode, seperti Notepad++, Visual Studio Code (VSCode), atau aplikasi sejenis. Kalau kamu pakai Notepad biasa juga bisa, tapi tampilannya mungkin kurang nyaman.

Di sini, saya akan menggunakan Notepad++.

Saat ini, file index.js masih kosong.

Kita akan menulis perintah pertama kita di sana.

Tulislah perintah seperti berikut:

tulis("Bismillah, Saya sedang belajar pemrograman");

**Perhatikan baik-baik:**

Saat menulis kode, kamu harus teliti dengan tanda-tanda seperti:

* Tanda kutip dua "
* Tanda kurung ( )
* Titik koma ; di akhir baris

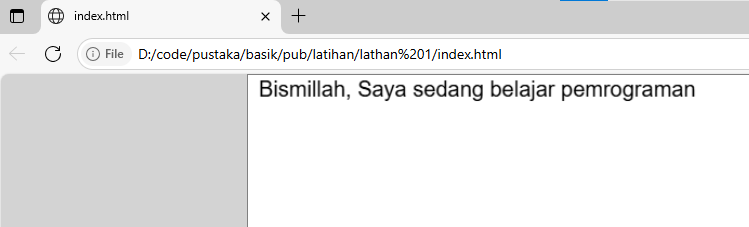
Semua itu penting supaya kode bisa berjalan dengan benar.

Setelah selesai menulis, simpan file index.js.

Lalu buka browser (misalnya Chrome atau Edge) dan tekan tombol F5 atau klik tombol refresh.

Tampilan di layar akan berubah!

Kira-kira tampilannya akan seperti ini:



Kalau kamu belum melihat gambar seperti di atas...

Coba cek lagi cara kamu menulis perintahnya. Mungkin ada yang terlupa, seperti:

* Salah ketik huruf atau tanda
* Lupa menutup kurung atau tanda kutip
* Tidak pakai titik koma; di akhir

Ingat, komputer sangat teliti! Sedikit saja salah, hasilnya bisa berbeda atau tidak muncul sama sekali.

Perintah pertama yang kita tulis tadi adalah untuk “menulis” sesuatu ke layar. Sesuai dengan namanya: perintah untuk menulis.

Sekarang, mari kita lanjut ke perintah kedua. Tulis di bawah perintah pertama, seperti ini:

tulis("Bismillah, Saya sedang belajar pemrograman");

tulis("Nama saya Budi");

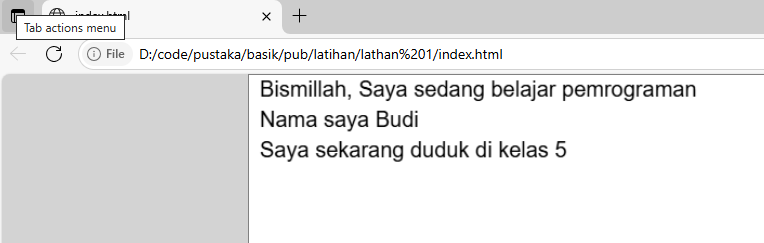
tulis("Saya sekarang duduk di kelas 5");

Jangan lupa!

Setelah menulis perintah kedua:

* Simpan file index.js
* Refresh browser dengan menekan tombol F5 atau klik tombol 🔄

Kalau semua perintah ditulis dengan benar, maka tampilannya akan seperti ini:



Sampai di sini, kalau tampilan yang kamu lihat mirip seperti contoh di atas...

Selamat! Artinya kamu sudah berhasil:

* Memberikan perintah kepada komputer
* Komputer mengerti perintahmu
* Dan komputer berhasil menjalankannya!

Kamu baru saja “berbicara” dengan komputer lewat kode.

Kalau belum muncul seperti yang diharapkan...

Jangan khawatir! Coba ulangi langkah-langkahnya:

Periksa apakah ada huruf yang salah atau kurang

Pastikan semua tanda seperti kurung dan kutip sudah benar

Jangan lupa simpan file index.js sebelum refresh

Dan yang penting: jangan copy-paste! Menulis sendiri akan membuat kamu lebih paham dan terbiasa.

Berikutnya, mari kita kenali bentuk perintah dalam JavaScript!

Secara umum, perintah di JavaScript punya bentuk seperti ini:

perintah("parameter1");

Penjelasan bentuk perintah:

Nama perintah biasanya satu atau dua kata, contohnya:

* tulis()
* update()
* prompt()

Kalau dua kata, penulisannya disambung dan huruf pertama kata kedua pakai huruf kapital, contohnya:

* muatImage()
* buatKanvas()

Tidak boleh ada spasi di nama perintah! Contoh yang salah: buat kanvas() ❌

Kurung buka dan tutup ( ) adalah tempat untuk menaruh parameter. Parameter itu seperti “info tambahan” yang kita berikan ke perintah.  
  
Contoh:

tulis("Bismillah, Saya sedang belajar");

Di sini, "Bismillah, Saya sedang belajar" adalah parameter yang memberi tahu apa yang harus ditulis.

Perintah bisa punya 0, 1, atau lebih banyak parameter. Kalau lebih dari satu, pisahkan dengan tanda koma ,

Contoh:

buatKanvas(300, 200);

Di akhir perintah biasanya ada titik koma; Ini seperti tanda “selesai”. Tapi tidak apa-apa, kalau kamu lupa, JavaScript kadang masih bisa jalan. Tetap disarankan untuk dibiasakan ya!

## Komentar

Komentar itu seperti catatan kecil yang kita tulis di dalam kode. Komentar tidak akan dijalankan oleh komputer, jadi fungsinya hanya untuk memberi penjelasan atau pengingat.

Cara menulis komentar:

Gunakan tanda // di awal baris, lalu tulis komentarmu.  
  
Contoh:

Tambahkan komentar seperti contoh di atas ke file index.js

//ini adalah komenter

//komputer tidak akan menjalankannya

//ini hanya untuk catatan saja

tulis("Bismillah, Saya sedang belajar pemrograman");

tulis("Nama saya Budi");

tulis("Saya sekarang duduk di kelas 5");

Simpan file dan jangan lupa Refresh browser-nya

Hasilnya? Tidak ada perubahan tampilan! Karena komentar memang tidak memengaruhi hasil program.

Kesimpulan Modul Ini, kita sudah belajar:

* Cara menulis perintah di JavaScript
* Cara memberi parameter ke perintah
* Cara membuat komentar sebagai catatan di dalam kode
* Dan bagaimana komputer menjalankan perintah yang kita tulis

# Variable - Si Kotak Ajaib Penyimpan Data

Sebelum kita mulai, mari kita siapkan tempat latihannya dulu:

copy folder template menjadi folder baru, dan beri nama folder ini dengan nama "Latihan 2".

## Apa itu Variable?

Bayangkan kamu punya banyak kotak kecil.

Setiap kotak punya label nama, dan kamu bisa memasukkan benda ke dalamnya.

Nah, dalam dunia pemrograman, variable adalah kotak itu.

Kita bisa menyimpan data di dalamnya, lalu memanggilnya kapan saja.

Contoh:

let namaSiswa = "Budi";

tulis(namaSiswa);

Penjelasan:

* let adalah perintah untuk membuat kotak (variable).
* namaSiswa adalah nama kotaknya.
* "Budi" adalah isi kotaknya.
* tulis(namaSiswa) akan menampilkan isi kotak, yaitu "Budi".

Perhatikan disini kita tidak menulis namaSiswa dengan tanda kutip karena namaSiswa adalah nama variable, jika kita menulis dengan tanda kutip maka hasilnya nanti jadi "namaSiswa" dan bukan "Budi".

## Aturan Penamaan Variable

* Tidak boleh pakai spasi.
* Gunakan huruf kecil di awal, dan huruf besar untuk kata kedua (contoh: namaSiswa).
* Nama variable harus jelas dan mudah dimengerti.

## Operator "=" dan "+"

Javascript memiliki beberapa operator antara lain = dan +:

* Tanda = digunakan untuk mengisi kotak.
* Tanda + digunakan untuk menggabungkan teks atau menjumlahkan angka.  
  Bila digunakan dengan teks dan angka, maka hasilnya akan digabungkan. Namun bila digunakan dengan angka dan angka maka hasilnya akan dijumlahkan.

Contoh:

let namaSiswa = "Budi";

tulis("Nama saya: " + namaSiswa);

Hasilnya:

Nama saya: Budi (screenshot)

Variable Bisa Diisi Belakangan

let namaSiswa;

namaSiswa = "Budi";

tulis("Nama saya: " + namaSiswa);

Variable juga bisa diisi dengan Angka

let roti = 9;

let coklat = 5;

let total = roti + coklat  
tulis(total)

Angka ditulis tanpa tanda kutip. Hasilnya adalah seperti ini:

## Tipe Data di JavaScript

* **String**: Teks, seperti "Budi"
* **Number**: Angka, seperti 90
* **Object**: Kita akan bahas nanti

## Tabel Contoh Penggunaan Variable

|  |  |
| --- | --- |
| **Perintah** | **Penjelasan** |
| let nama; | Membuat kotak bernama nama, belum diisi |
| let nama;  nama = "Umar"; | Membuat kotak nama, lalu diisi dengan teks "Umar" |
| let nama = "Umar"; | Membuat dan mengisi kotak nama sekaligus |
| let roti = 9;  let coklat = 5;  let total = roti + coklat | Menghitung total makanan dan menyimpannya di kotak total |
| let donat = 10;  donat = donat + 10; | Membuat variable donat dan mengisinya dengan angka 10  Menambahkan isi dari kotak donat dengan 10.  Sekarang isinya jadi 20 Ini adalah salah satu pola yang sering digunakan dalam pemrograman |

Prompt

Latihan 2:  
Tulislah latihan sebelumnya dengan menggunakan variable.

# Fungsi tulis tambahan

# Prompt

Tentu, Fajar! Berikut versi yang lebih rapi dan mudah dipahami untuk anak SD atau SMP, dengan gaya yang ringan dan penuh contoh nyata:

---

## 🧮 Operator dalam JavaScript

Dalam JavaScript, \*operator\* adalah simbol yang digunakan untuk melakukan berbagai macam perhitungan, seperti penjumlahan, pengurangan, dan lainnya.

### 📌 Contoh Operator Pengisian (=)

```javascript

let namaSiswa = "Budi";

```

Tanda `=` digunakan untuk \*mengisi\* nilai ke dalam sebuah variabel. Di sini, kita mengisi nama siswa dengan "Budi".

---

### 📌 Contoh Operator Matematika

```javascript

let jumlahNanas = 5;

let jumlahApel = 6;

let total = jumlahNanas + jumlahApel;

```

Tanda `+` digunakan untuk \*menjumlahkan\* dua angka. Di sini, kita menjumlahkan jumlah nanas dan apel.

---

### 📌 Gabungan Teks dan Angka

```javascript

tulis("Nama siswa: " + namaSiswa);

tulis("Jumlah Nanas dibeli: " + jumlahNanas);

tulis("Jumlah Apel dibeli: " + jumlahApel);

tulis("Total buah dibeli: " + total);

```

Tanda `+` juga bisa digunakan untuk \*menggabungkan teks\*. Kalau kita gabungkan teks dan angka, hasilnya akan menjadi satu kalimat panjang.

Contoh:

- `"Jumlah Apel dibeli: " + 6` → hasilnya: `"Jumlah Apel dibeli: 6"`

---

### 📚 Daftar Operator Matematika

| Simbol | Arti |

|--------|-------------|

| `+` | Penjumlahan |

| `-` | Pengurangan |

| `\*` | Perkalian |

| `/` | Pembagian |

---

## ✂️ Operator Singkatan

Kadang kita bisa menulis perhitungan dengan cara yang lebih singkat.

### Contoh:

```javascript

let angka = 5;

angka = angka + 10;

```

Bisa disingkat menjadi:

```javascript

let angka = 5;

angka += 10;

```

Tanda `+=` artinya: tambahkan angka 10 ke variabel `angka`.

---

### Tambah 1 dengan Cara Singkat

```javascript

let angka = 5;

angka = angka + 1;

```

Bisa disingkat menjadi:

```javascript

let angka = 5;

angka++;

```

Tanda `++` artinya: tambahkan 1 ke variabel `angka`.

---

### 🎯 Kesimpulan

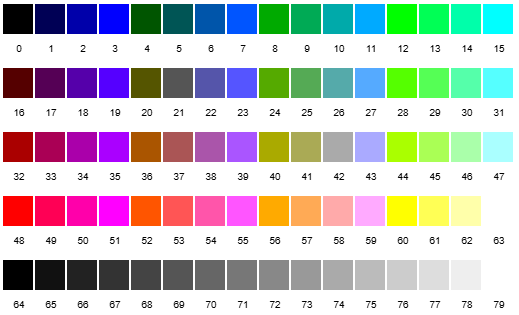
- Gunakan `+`, `-`, `\*`, dan `/` untuk menghitung angka.

- Gunakan `+` juga untuk menggabungkan teks.

- Gunakan `+=` atau `++` untuk menulis lebih singkat.

Kalau kamu ingin, kita bisa buat latihan interaktif atau kuis kecil untuk menguji pemahaman. Mau coba? 😊

# Warna



# 🌀 Belajar Perulangan dengan Perintah `for` di JavaScript

Kadang-kadang, kita ingin melakukan sesuatu berulang kali. Misalnya, menulis kalimat sebanyak 10 kali atau menghitung dari 1 sampai 100. Kalau kita menulis satu per satu, pasti capek dan lama, kan?

Nah, di JavaScript, ada cara pintar untuk mengulang perintah secara otomatis. Namanya **perulangan** atau **looping**. Salah satu jenis perulangan adalah for.

Dengan for, kita bisa membuat kode yang lebih pendek dan rapi!

📘 **Contoh Perulangan**

for (let i = 1; i <= 10; i++) {

tulis("Bismillah, saya sedang belajar pemrograman");

}

Kalimat di atas akan ditulis sebanyak 10 kali! Yuk kita pahami bagian-bagian dari perintah `for` ini:

* let i = 1 → Kita membuat sebuah \*\*variabel\*\* bernama `i`, dan kita isi dengan angka 1.
* i <= 10 → Selama nilai `i` masih kurang dari atau sama dengan 10, perulangan akan terus berjalan.
* i++ → Setelah satu kali perulangan selesai, nilai `i` akan bertambah 1. Ini sama seperti i = i + 1.
* tulis(...) → Ini adalah perintah yang akan dijalankan setiap kali perulangan terjadi.

**🧠 Menambahkan Lebih Banyak Perintah**

Kita bisa menambahkan lebih dari satu perintah di dalam perulangan. Semua perintah itu harus ditulis di antara tanda { dan }.

for (let i = 1; i <= 10; i++) {

tulis("Bismillah, saya sedang belajar pemrograman");

tulis("Saya senang belajar pemrograman");

}

Setiap kali perulangan berjalan, kedua perintah tulis() akan dijalankan.

🔢 **Menampilkan Nilai i**

Kita juga bisa menampilkan nilai dari variabel i agar tahu sudah berapa kali perulangan berjalan.

for (let i = 1; i <= 10; i++) {

tulis("Nilai i saat ini: " + i);

tulis("Bismillah, saya sedang belajar pemrograman");

tulis("Saya senang belajar pemrograman");

}

✨ **Indentasi (Menjorok ke Dalam**)

Kalau kamu perhatikan, semua perintah di dalam {} ditulis agak menjorok ke kanan. Ini disebut **indentasi**.

Indentasi membuat kode lebih rapi dan mudah dibaca. Walaupun tidak wajib, sebaiknya kamu biasakan menulis seperti ini.

Kalau kamu pakai editor seperti **VS Code** atau **Notepad++**, biasanya indentasi akan dibuat otomatis.

# Mengenal percabangan IF

Secara umum, perintah-perintah dalam bahasa pemrograman disusun dari atas ke bawah. Namun ada kondisi dimana perintah itu akan bercabang sesuai dengan kondisi tertentu.

Di Javascript kita memiliki perintah IF untuk menangani percabangan

Dengan menggunakan IF kita bisa mengubah alurnya sesuai dengan kondisi yang kita berikan.

Syntax IF adalah sebagai berikut:

if (kondisi) {

[baris-baris perintah yang dijalankan]

}

kondisi biasanya berupa perbandingan menggunakan operator perbandingan

let jam = 8

if (jam < 12) {

tulis("Hari masih pagi");

}

else {

tulis("Hari sudah siang");

}

Pada contoh diatas, karena jam bernilai 8, maka kondisi (jam < 12) akan bernilai true/benar dan perintah yang ada di antara dua kurung keriting {} akan dijalankan.

If biasa digabungkan dengan else, contoh:

let jam = 8

if (jam < 12) {

TulisBrs("Hari masih pagi");

}

else {

TulisBrs("Hari sudah sore");

}

Pada contoh diatas, apabila nilai dari jam lebih dari 12, maka yang dijalankan adalah perintah TulisBrs("Hari sudah sore"); dan perintah sebelumnya akan dilewati.

Untuk pengkondisian yang lebih rumit, kita gunakan else if, contoh:

let cuaca = "mendung";

if (cuaca == "hujan") {

console.log("Bawa payung.");

} else if (cuaca == "mendung") {

console.log("Siap-siap payung.");

} else {

console.log("Jalan santai.");

}

# Mengenal Function

Function adalah kumpulan perintah yang dijadikan satu agar bisa dipakai berulang-ulang.

Untuk bisa memanfaatkan Function, pertama Anda harus mendeklarasikannya terlebih dahulu. Kemudian Anda bisa memakainya dengan cara memanggil fungsi tersebut.

Function dideklarasikan dengan cara berikut:

function namaFunction() {

[perintah yang akan dijalankan]

[perintah yang akan dijalankan]

[perintah yang akan dijalankan]

}

berikut adalah contoh function untuk menulis resep

function resepGorengTelur() {

TulisBrs("Siapkan peralatan memasak");

TulisBrs("Panaskan wajan");

TulisBrs("Masak telur di atas wajan");

TulisBrs("Goreng sampai matang");

TulisBrs("Telur siap disajikan");

}

setelah itu kita bisa menggunakan function ini dengan memanggil namanya:  
resepGorengTelur();

Function bisa dikembangkan untuk menerima parameter.

Contoh

function TulisXY(x, y, teks) {

PosisiTeks(x,y);

Tulis(teks);

}

kita bisa menggunakan fungsi ini dengan cara memanggillnya sbb:

tulisXY(100,100, "Hallo");

tulisXY(200,200, "Saya sedang belajar");

tulisXY(100, 300, "Saya senang belajar koding");

Kita bisa memanggil fungsi berkali-kali. Dengan demikian kita bisa menghemat baris perintah.

Latihan:

buat lagi latihan yang dulu tapi memakai fungsi

# Menggambar Bentuk

# Event

Event adalah sebuah kejadian/aksi yang terjadi ketika sebuah aplikasi berjalan. Contoh Event adalah ketika User menggerakkan mouse, ketika user menekan tombol mouse, ketika sebuah gambar selesai di load, dsb.

BASIC menyediakan cara penanganan event dengan cara yang sederhana. Kita cukup membuat fungsi dengan nama yang sudah ditentukan untuk menangani event tertentu.

salah satu contoh event adalah event Udate. Event update adalah event yang terjadi saat applikasi di update.

Applikasi diupdate rata-rata 30-60 kali perdetik. Kita bisa menangani event update dengan cara membuat function dengan nama Update()

contoh:

function Update() {

Cls();

PosisiTeks(MouseX(), MouseY());

Tulis("Halo saya sedang belajar pemrograman JavaScript");

}

Bila dijalankan, maka ketika Anda akan melihat tulisan yang mengikuti posisi mouse.

Cls() adalah perintah untuk membersihkan layar.

MouseX(), dan MouseY() masing-masing akan mengembalikan posisi x dan y dari mouse.

contoh 2:

function TombolDitekan() {

PosisiTeks(MouseX(), MouseY());

Tulis("Halo saya sedang belajar pemrograman JavaScript");

}

Saat kita mengklik di layar, maka akan muncul tulisan di tempat kita mengklik.

Berikut adalah daftar event dan fungsi yang berkaitan.

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

### 

# Image

Image adalah sebuah object yang memudahkan kita untuk memanipulasi gambar.

Sebuah image bisa dibuat melalui fungsi MuatImage(). Fungsi ini akan mengembalikan object image yang perlu kita simpan dalam sebuah variable

latihan Image 1

let bg = muatImage("bg\_bintang.jpg");

function update() {

gambarImage(bg);

}



Pada contoh di atas kita memuat image dari file "bg\_bintang.jpg" kemudian kita menyimpannya dalam variable bg.

file "bg\_bintang.jpg" ada di dalam folder asset. Anda bisa membuka folder asset dan akan menemukan banyak file gambar di sana.

fungsi muatImage() memiliki satu parameter bertipe string yang berisi file apa yang akan kita muat.

Perintah gambarImage() adalah perintah untuk menggambar object Image di layar. Kita menaruh perintah ini didalam fungsi update agar Imagenya di gambar terus-menerus.

Berikutnya kita akan memuat beberapa gambar lagi.

latihan Image 1:

let bg = muatImage("bg\_bintang.jpg");

let matahari = muatImage("matahari.png");

let roket = muatImage("roket.png");

function update() {

gambarImage(bg);

gambarImage(matahari);

gambarImage(roket);

}

Kita memuat dua gambar tambahan, yaitu "matahari.png" dan "roket.png". Dua file ini tersedia di folder asset.

Jangan lupa untuk menggambar Image ke layar agar bisa dilihat.



sebuah Image adalah sebuah object. Object adalah variable yang kompleks yang memiliki property dan method. property bisa disebut juga sifat, seperti panjang, lebar, posisi, dll. method adalah sesuatu yang bisa dilakukan oleh object seperti jalan, lari, lompat, dll.

perintah muatImage() akan menghasilkan sebuah Object Image. object Image memiliki property seperti panjang, lebar, rotasi, posisi, transparansi, dll.

Kita akan menggunakan property-property ini untuk memanipulasi object Image. Perhatikan contoh berikut:

latihan Image 1:

let bg = muatImage("bg\_bintang.jpg");

let matahari = muatImage("matahari.png");

let roket = muatImage("roket.png");

matahari.x = 142;

matahari.y = 142;

function update() {

gambarImage(bg);

gambarImage(matahari);

gambarImage(roket);

}

Pada contoh di atas kita menambahkan dua baris perintah:

matahari.x = 142;

matahari.y = 142;

Perintah di atas adalah contoh penggunaan property. Untuk mengakses property dari sebuah object kita menambahkan titik diikuti dengan nama property.

x dan y adalah property untuk menentukan posisi x dan y dari sebuah object Image. Kita mengisi kedua property ini dengan nilai 142.

Hasilnya akan tampak seperti ini: perhatikan bahwa sekarang posisi matahari sudah bergeser ke kanan dan ke bawah.



Selanjutnya kita akan mengubah ukuran matahari dengan mengubah properti panjang dan lebar.

let bg = muatImage("bg\_bintang.jpg");

let matahari = muatImage("matahari.png");

let roket = muatImage("roket.png");

matahari.x = 142;

matahari.y = 142;

matahari.panjang = 128;

matahari.lebar = 128;

function update() {

gambarImage(bg);

gambarImage(matahari);

gambarImage(roket);

}

Hasilnya adalah seperti berikut:



Selanjutnya kita akan membuat tampilannya lebih interaktif. Kita akan membuat supaya roket bisa bergerak mengikuti mouse. Untuk itu kita akan menggunakan perintah mouseX() dan mouseY().

let bg = muatImage("bg\_bintang.jpg");

let matahari = muatImage("matahari.png");

let roket = muatImage("roket.png");

matahari.x = 142;

matahari.y = 142;

matahari.panjang = 128;

matahari.lebar = 128;

function update() {

roket.x = mouseX();

roket.y = mouseY();

gambarImage(bg);

gambarImage(matahari);

gambarImage(roket);

}

Jika diperhatikan posisi roket sekarang berada di sebelah kanan bawah kursor. Ini adalah posisi default dari pusat Image. Pusat Image dipakai untuk menentukan pusat rotasi dan menggambar. Kita akan memindahkannya ke tengah.

let bg = muatImage("bg\_bintang.jpg");

let matahari = muatImage("matahari.png");

let roket = muatImage("roket.png");

matahari.x = 142;

matahari.y = 142;

matahari.panjang = 128;

matahari.lebar = 128;

**roket.pusatX = 32;**

**roket.pusatY = 46;**

function update() {

roket.x = mouseX();

roket.y = mouseY();

gambarImage(bg);

gambarImage(matahari);

gambarImage(roket);

}

Perhatikan code di atas. Kita telah memindahkan pusat untuk image roket ke tengah. Sekarang roket akan tampak di tengah-tengah kursor mouse. Panjang dan lebar image roket adalah 64 dan 92, sehingga tenganya berada pada posisi 32 dan 46.

Berikutnya kita akan membuat matahari berputar. Kita akan menggunakan property rotasi. Pada saat image dimuat, rotasi akan memiliki nilai 0.

let bg = muatImage("bg\_bintang.jpg");

let matahari = muatImage("matahari.png");

let roket = muatImage("roket.png");

matahari.x = 142;

matahari.y = 142;

matahari.panjang = 128;

matahari.lebar = 128;

roket.pusatX = 32;

roket.pusatY = 46;

function update() {

roket.x = mouseX();

roket.y = mouseY();

**matahari.rotasi++;**

gambarImage(bg);

gambarImage(matahari);

gambarImage(roket);

}

Kita menggunakan operator ++ untuk menambah rotasi matahari dengan angka satu. Kita menaruh perubahan rotasi di fungsi update() sehingga rotasinya akan terus menerus ditambah.

Kalau kita perhatikan, matahari sekarang berotasi, tapi terlihat aneh. Ini karena matahari berotasi namun pusat rotasinya tidak di tengah. Kita perlu mengubahnya. Caranya sama seperti pada roket.

let bg = muatImage("bg\_bintang.jpg");

let matahari = muatImage("matahari.png");

let roket = muatImage("roket.png");

matahari.x = 142;

matahari.y = 142;

matahari.panjang = 128;

matahari.lebar = 128;

**matahari.pusatX = 64;**

**matahari.pusatY = 64;**

roket.pusatX = 32;

roket.pusatY = 46;

function update() {

roket.x = mouseX();

roket.y = mouseY();

matahari.rotasi++;

gambarImage(bg);

gambarImage(matahari);

gambarImage(roket);

}

# Pena