Dokumentasi SDK Resmi

Game Engine

- SysX Engine -

BAB 1: Pengembangan V2.0

Dokumentasi SDK Lanjutan

SysX Engine v2.0 "Ecosystem",

Tanggal Rilis: 23 Juli 2025

Status: Full Release

Selamat Datang Kembali, Developer!

Ini adalah panduan resmi untuk **SysX Engine v2.0 "Ecosystem"**, sebuah lompatan besar dari versi sebelumnya. Jika v1.2 adalah tentang "Juice", maka v2.0 adalah tentang membangun **dunia yang hidup** di sekitar game Anda. Engine ini tetap berpegang pada filosofi intinya—simpel, terbatas, dan *juicy*—namun kini dengan kekuatan untuk berkembang, beradaptasi, dan dimodifikasi bahkan setelah dirilis.

Perjalanan kita penuh dengan *trial and error*, tapi hasilnya adalah sebuah fondasi yang kokoh. Mari kita selami semua kemampuan baru yang telah kita bangun bersama.

BAB 1.1: Filosofi Engine

Filosofi kita tidak berubah, hanya semakin kuat:

- **Simpel & Terbatas:** API yang ramping untuk mendorong kreativitas.
- **Juicy by Default:** Fitur bawaan untuk membuat game terasa hidup.

SysX
ENGINE

- **Modular & Profesional:** Pemisahan yang jelas antara engine dan game, kini diperluas dengan kemampuan memuat modul eksternal (DLC, Patch, Mod).

BAB 2: Menjalankan DEV KIT

Cara menjalankan Dev Kit tetap sama: **gunakan server lokal** seperti "Live Server" di Visual Studio Code untuk menghindari error keamanan browser.

Struktur Folder DEV KIT v2.0: Pastikan proyek Anda mengikuti struktur folder standar ini agar semua fitur berjalan lancar.

```
SYSX_ENGINE_V2_DEVKIT/
├── index.html              <-- File utama untuk menjalankan game
│
├── sysx_engine/            <-- Folder untuk SEMUA file inti engine
│   ├── sysx_core.js
│   ├── sysx_graphics.js
│   └── ... (semua file sysx_*.js lainnya)
│
└── game/                   <-- Folder untuk file spesifik game buatan Anda
    ├── manifest.js
    └── game.js
```

BAB 3: Referensi API (Update vSebelumnya)

(Gunakan API yang sudah kita definisikan di v0.9 dan v1.0)

Pembaruan: Referensi API Lengkap (Update v1.2.0 "JuiceBox")

Core & State

(Tidak ada perubahan pada API publik, jadi bagian ini tetap sama)

SysX
ENGINE

SysX.state(newState) SysX.debug(boolean) SysX.watch(key, value)

Graphics

(Bagian ini perlu diperbarui untuk mencerminkan fungsi-fungsi baru)

SysX.cls(colorIndex) Membersihkan seluruh layar dengan satu warna dari palet.

SysX.rectfill(x, y, w, h, colorIndex) Menggambar persegi panjang yang terisi warna.

SysX.line(x1, y1, x2, y2, colorIndex) - **(BARU)** Menggambar sebuah garis dari titik (x1, y1) ke (x2, y2).

SysX.circ(x, y, radius, colorIndex) - **(BARU)** Menggambar garis tepi lingkaran.

SysX.circfill(x, y, radius, colorIndex) - **(BARU)** Menggambar lingkaran yang terisi warna.

SysX.print(text, x, y, colorIndex) Menampilkan teks di layar.

SysX.spr(spriteIndex, x, y, flipX?, flipY?, angle?) - **(DIPERBARUI)** Menggambar sebuah sprite dari spritesheet.

- flipX (opsional): true untuk membalik sprite secara horizontal.
- flipY (opsional): true untuk membalik sprite secara vertikal.
- angle (opsional): Memutar sprite sebesar angle derajat.

SysX.map(tilemap, x, y) Menggambar seluruh tilemap ke layar.

BAB 3.1: Kanal System (Update v1.2.0 "JuiceBox")

Audio

(Bagian ini perlu diperbarui untuk menjelaskan sistem kanal)

SysX.sfx(assetKey, channel?) - **(DIPERBARUI)** Memainkan efek suara. Engine memiliki 8 kanal audio (0-7).

- channel (opsional): Jika ditentukan, akan memainkan suara di kanal tersebut (menggantikan suara yang ada). Jika tidak, akan mencari kanal yang kosong. Jika semua kanal penuh, suara tidak akan dimainkan.

SysX.music(assetKey, loop?, stop?) Memainkan atau menghentikan musik latar.

Input (Keyboard, Mouse, Mobile)

(Bagian ini ditulis ulang sepenuhnya untuk mencakup semua fitur baru)

Modul input telah diperbarui untuk mendukung keyboard, mouse, dan kontroler sentuh secara terpadu.

Keyboard & Tombol Virtual:

- **SysX.btn(key)**: Mengembalikan true jika tombol ditekan terus-menerus.

- **SysX.btnp(key)**: Mengembalikan true hanya pada frame pertama tombol ditekan.

Nama key yang bisa digunakan: 'up', 'down', 'left', 'right', 'btn_a', 'btn_b', 'start', 'power'.

Mouse:

- **SysX.mouse.x**: Mengembalikan posisi horizontal (X) kursor mouse di dalam canvas.
- **SysX.mouse.y**: Mengembalikan posisi vertikal (Y) kursor mouse di dalam canvas.
- **SysX.mouse.btn(button)**: Sama seperti SysX.btn, untuk tombol mouse. 0 untuk klik kiri, 1 untuk tengah, 2 untuk kanan.
- **SysX.mouse.btnp(button)**: Sama seperti SysX.btnp, untuk tombol mouse.

Kontroler Mobile: Pada perangkat sentuh, sebuah kontroler virtual (D-Pad, tombol A/B, tombol Menu) akan otomatis muncul di layar. Tombol-tombol ini akan memicu SysX.btn() dan SysX.btnp() dengan nama key yang sama seperti keyboard, sehingga game Anda bisa langsung dimainkan di mobile tanpa perubahan kode.

Core & State ini versi lama (bisa mengikuti versi baru di SUB BAB Kanal System)

- SysX.state(newState)
- SysX.debug(boolean)
- SysX.watch(key, value)

Graphics ini versi lama (bisa mengikuti versi baru di SUB BAB Kanal System)

- SysX.cls(colorIndex)
- SysX.rectfill(x, y, w, h, colorIndex)
- SysX.print(text, x, y, colorIndex)
- SysX.spr(spriteIndex, x, y)
- SysX.map(tilemap, x, y)

Audio ini versi lama (bisa mengikuti versi baru di SUB BAB Kanal System)

- SysX.sfx(assetKey)
- SysX.music(assetKey, loop?, stop?)

Input ini versi lama (bisa mengikuti versi baru di SUB BAB Kanal System)

- SysX.btn(key) & SysX.btnp(key)

Effects & Physics

- SysX.collide(obj1, obj2)
- SysX.shake(intensity, durationMs)
- SysX.emit(x, y, count?, color?, life?)

SysX
ENGINE

BAB 3.2: Referensi API Lengkap (Update v2.0) (BARU)

Ini adalah daftar semua fungsi yang bisa Anda panggil.

Graphics (Update V1.3)

Fungsi-fungsi lama seperti cls, rectfill, spr, dll. tetap ada.

- SysX.sspr(sx, sy, sw, sh, dx, dy, dw?, dh?) Menggambar sebuah **bagian (region)** dari spritesheet utama ke layar. Sangat berguna untuk sprite dengan ukuran non-standar.
    - sx, sy, sw, sh: Posisi X, Y, lebar, dan tinggi dari sumber gambar di *spritesheet*.
    - dx, dy: Posisi X dan Y tujuan di layar.
    - dw, dh (Opsional): Lebar dan tinggi tujuan, untuk me-resize gambar.

Physics (Update V1.3)

Ditambahkan ke modul fx.

- SysX.collide_circ(c1, c2) Mendeteksi tabrakan antara **dua lingkaran**. Objek harus memiliki properti x, y, dan radius.
- SysX.collide_circ_rect(circ, rect) Mendeteksi tabrakan antara **lingkaran dan persegi**.

Transitions (Fitur Baru V1.3)

Modul baru sysx_transition.js.

- SysX.transition(type, duration, onComplete) Memulai transisi layar yang mulus saat berpindah state.
    - type: Jenis transisi. Saat ini hanya mendukung 'fade'.
    - duration: Durasi transisi dalam frame (1/60 detik).
    - onComplete: Fungsi yang akan dipanggil **setelah** transisi selesai (biasanya untuk memanggil SysX.state()).

Contoh:

```
// Pindah ke state 'game' dengan efek fade selama 1 detik

SysX.transition('fade', 60, () => {

    SysX.state('game');

});
```

SysX

ENGINE

Konten Dinamis (Fitur Baru V2.0)

Modul baru sysx_content.js. Memerlukan library eksternal **JSZip**.

- SysX.content_load_zip(fileObject) Memuat paket konten (DLC, Patch, Mod) dari file .zip yang dipilih oleh pengguna.
    - fileObject: Objek file yang didapat dari elemen <input type="file">.
- SysX.content_load_url(url) Memuat sebuah script mod dari URL eksternal. Hanya untuk modding via konsol.

Konsol Developer (Fitur Baru V2.0)

Modul baru sysx_devconsole.js.

- SysX.console_toggle() Membuka atau menutup konsol developer.
- SysX.console_register(commandName, callback) Mendaftarkan sebuah "cheat" atau perintah custom ke dalam konsol.
    - commandName: Nama perintah (string) yang akan diketik pemain.
    - callback: Fungsi yang akan dijalankan saat perintah dipanggil.
- SysX.console_log(message, className?) Menampilkan pesan di dalam konsol game. Berguna untuk memberikan feedback dari DLC atau Patch.

Bab 3.3: Modul Booting dan Implementasi File .sysboot pada SysX Engine (VERSI LAMA V1.0 (TIDAK DIPAKAI LAGI HANYA DOKUMENTASI))

1. Pendahuluan

Modul booting pada SysX Engine berfungsi sebagai sistem tampilan intro ketika game pertama kali dijalankan. Mulai dari simulasi BIOS hingga tampilan logo ASCII dan slogan khas engine. Fitur ini memberikan sentuhan profesional, identitas kuat, dan peluang personalisasi terhadap setiap game yang dibangun dengan SysX.

Apa Itu File .sysboot?
File .sysboot adalah file konfigurasi berbasis JSON yang memungkinkan developer mengatur tampilan dan urutan cinematic intro pada saat game booting. Jika file ini ditemukan oleh engine, maka engine akan menggantikan intro default dengan tampilan cinematic sesuai .sysboot.

Lokasi: assets/boot/intro.sysboot

SysX
ENGINE

2. Struktur File .sysboot

Contoh struktur dasar:

```
{
  "version": "1.0.0",
  "title": "BOOTING GALAXY RUSH",
  "steps": [
    "[SYSX BIOS v1.0.0]",
    "> Initializing galaxy core...",
    "> Mounting cartridge.sysx",
    "> Loading hyperspace engine...",
    "> Registering core...",
    "> Registering modules...",
    "> OK",
    "> SYSTEM READY."
  ],
  "logo": [
    "    ____  __     __  _____ __  __ ",
    "  / ___| \\ \   / / |_   _|  \\/  |",
    " | |      \\ \\ / /    | | | |\\/| |",
    " | |___    \\ V /     | | | |  | |",
    "  \\____|    \\_/      |_| |_|  |_|"
  ],
  "slogan": "Rush Beyond the Stars.",
  "color": 10,
  "duration": 10
}
```

Penjelasan Properti:

Properti	Tipe	Keterangan
version	String	Versi sistem / game (opsional kosmetik)
title	String	Judul boot (tidak ditampilkan secara default)
steps	Array	Baris proses boot seperti terminal BIOS
logo	Array	Logo dalam bentuk ASCII
slogan	String	Kalimat atau kutipan di akhir boot
color	Number	Warna utama teks dan logo (default: 10)
duration	Number	Waktu tampil per baris step (default: 10f)

3. Cara Kerja dan Prioritas Boot

Saat SysX.init() dipanggil, engine secara otomatis:

1. Mengecek apakah manifest.showIntro diatur false
   - Jika iya → langsung skip intro
2. Jika tidak, mencoba memuat file assets/boot/intro.sysboot
   - Jika ditemukan → akan dijalankan via drawSysBootIntro(data)
   - Jika gagal (tidak ada/file rusak) → fallback ke drawIntro() default

4. Keuntungan Penggunaan .sysboot

SysX
ENGINE

- Personalisasi intro tiap game
- Konsistensi identitas visual (tema, warna, pesan)
- Bisa diganti tanpa menyentuh source engine
- Fallback otomatis jika file tidak tersedia
- Tidak mengganggu struktur engine asli

5. Tips Penggunaan

- Gunakan monospace saat membuat ASCII logo untuk menjaga proporsi
- Jangan terlalu panjang baris steps, karena keterbatasan layar 320x240
- Hindari penggunaan karakter non-ASCII untuk kompatibilitas print()
- Gunakan color dari 0–15 (mengikuti palet PICO-8 bawaan engine)

(Update v1.2.0 "JuiceBox")BARU CARA CUSTOM BOOT)

Kustomisasi Boot (v2.0)

1. Filosofi Baru: Dari Konfigurasi Statis ke State Dinamis

Pada versi sebelumnya, kustomisasi boot dilakukan melalui file intro.sysboot yang berbasis JSON. Meskipun mudah, pendekatan ini memiliki keterbatasan: intro yang dihasilkan bersifat statis dan tidak dapat memanfaatkan kekuatan penuh dari SysX Engine.

Pada SysX Engine v2.0, kami memperkenalkan sistem yang jauh lebih kuat dan fleksibel: **State boot**.

Alih-alih mengkonfigurasi file JSON, Anda sekarang dapat membuat intro game Anda sebagai state pertama yang dijalankan oleh engine. Ini berarti Anda memiliki akses penuh ke **semua API SysX Engine** (tween, emit, timer, spr, dll.) untuk menciptakan pengalaman boot yang benar-benar unik, "juicy", dan interaktif.

2. Cara Implementasi

Untuk membuat custom boot, Anda hanya perlu mendefinisikan state bernama boot di dalam file game.js Anda. Engine secara otomatis akan mendeteksi dan menjalankan state ini terlebih dahulu.

Langkah-langkah:

1. **Panggil SysX.state('boot') di dalam _create():** Ini memberitahu engine untuk memulai game dari state boot Anda.

```
// Di dalam file game.js
window.Game = {
    _create: function() {
        SysX.state('boot');
    },
    // ... state lainnya
};
```

9. **Definisikan State boot:** Buat tiga fungsi standar untuk state boot: _init_boot, _update_boot, dan _draw_boot.

SysX
ENGINE

```
window.Game = {
    _create: function() { ... },

    // ================================= //
    // =====      STATE: boot      ===== //
    // ================================= //
    _init_boot: function() {
        // Setup awal untuk intro Anda di sini.
        // Contoh: siapkan variabel untuk animasi.
        this.logoPosition = { y: -20 };
        SysX.tween(this.logoPosition, { y: 110 }, 60, 'easeOut');
    },

    _update_boot: function() {
        // Logika intro Anda setiap frame.
        // Contoh: pindah ke state 'title' setelah beberapa detik
        // atau saat pemain menekan tombol.
        if (SysX.btnp('btn_a')) {
            SysX.state('title');
        }
    },

    _draw_boot: function() {
        // Gambar intro Anda setiap frame.
        SysX.cls(0); // Latar hitam
        SysX.print("NAMA STUDIO ANDA", 80, this.logoPosition.y, 7);
    }
};
```

3. Keuntungan Sistem Baru

- **Kebebasan Tanpa Batas:** Anda tidak lagi terbatas pada teks statis. Buat logo yang beranimasi, partikel yang meledak, atau bahkan intro yang bisa dimainkan.
- **Akses API Penuh:** Gunakan SysX.tween, SysX.emit, SysX.timer, SysX.spr, dan semua fungsi lainnya untuk membuat intro Anda "juicy".
- **Konsistensi:** Anda menggunakan sistem state yang sama untuk membuat intro seperti saat Anda membuat game. Ini membuat alur kerja lebih bersih dan konsisten.
- **Fallback Otomatis:** Jika Anda **tidak** mendefinisikan state boot di game.js, engine akan secara otomatis menjalankan intro default bawaan. Game Anda tidak akan rusak.

4. Kesimpulan

Sistem state boot adalah evolusi dari .sysboot yang sejalan dengan filosofi SysX Engine. Ini memberikan kekuatan dan kebebasan kreatif sepenuhnya kepada Anda sebagai developer untuk membangun identitas yang tak terlupakan dari detik pertama game Anda dijalankan.

6. Rencana Ekstensi Fitur (Optional Development)

Disarankan untuk pengembang lanjut atau kolaborasi open-source

- progressBar: Menampilkan animasi progress bar palsu
- bootSFX: Nama efek suara khusus untuk boot
- autoSkipAfter: Durasi sebelum intro otomatis skip

SysX
ENGINE

- backgroundColor: Warna background boot
- loopIntro: Jika true, intro dapat diulang sebelum lanjut

Bab 3.4: API "Juiciness" & Animasi (Update v1.2.0 "JuiceBox")

API "Juiciness" & Animasi
Fitur-fitur ini didesain untuk membuat game Anda terasa lebih hidup dan dinamis dengan sedikit usaha.
Kamera
SysX.camera(x, y) Mengatur posisi kamera virtual. Semua objek yang digambar (spr, map, rectfill, dll.) akan bergeser mengikuti posisi kamera. Panggil SysX.camera() tanpa argumen untuk mereset posisi ke (0, 0).
Tweening (Animasi Properti)
SysX.tween(target, properties, duration, ease?, onComplete?) Menganimasikan properti sebuah objek secara mulus.

- target: Objek yang ingin dianimasikan (misal: this.player).
- properties: Objek berisi nilai akhir yang dituju (misal: { y: 100, alpha: 0 }).
- duration: Durasi animasi dalam *frame*.
- ease (opsional): Jenis transisi. Pilihan: 'linear', 'easeIn', 'easeOut', 'easeInOut'.
- onComplete (opsional): Fungsi yang akan dipanggil setelah animasi selesai.

Timer (Alarm)
SysX.timer_set(name, duration, callback, isLooping?) Menjalankan sebuah fungsi setelah jeda waktu tertentu.

- name: Nama unik untuk timer (String).
- duration: Durasi dalam *frame*.
- callback: Fungsi yang akan dieksekusi.
- isLooping (opsional): Jika true, timer akan otomatis diulang.

SysX.timer_is_running(name) Mengembalikan true jika timer dengan nama tersebut sedang aktif.
SysX.timer_cancel(name) Menghentikan dan menghapus timer.

7. Kesimpulan

Fitur .sysboot membuat SysX Engine tidak hanya sebagai game engine, tapi juga memberi ruang ekspresi terhadap nuansa pembukaan tiap game. Ini adalah pondasi penting untuk membuat engine terasa seperti *konsol pribadi* yang punya gaya tersendiri.

Gunakan fitur ini sebaik-baiknya untuk membangun **identitas yang tak terlupakan** dari setiap karya digital lo.

End of Chapter - SysX Boot Module SDK

Fighting Developer !!! Selamat Membuat Game dengan Sysx Engine.

BAB 4: Panduan Implementasi Fitur V2.0

Ini adalah jantung dari update "Ecosystem".

SysX
ENGINE

4.1 Membuat "Patch" (Perbaikan Bug)

Gunakan Patch untuk memperbaiki bug tanpa merilis ulang seluruh game.

1. **Identifikasi Bug:** Misal, di game.js, player bisa bergerak keluar layar.
2. **Buat Script Perbaikan (patch_logic.js):**
3. // --- scripts/patch_logic.js ---
4. console.log("PATCH v1.0.1 DITERAPKAN!");
5.
6. // Simpan referensi ke fungsi _update_game yang lama
7. const original_update_game = window.Game._update_game;
8.
9. // Timpa fungsi yang lama dengan versi perbaikan
10. window.Game._update_game = function() {
11. // Panggil dulu logika aslinya jika perlu
12. original_update_game.call(this);
13.
14. // Tambahkan logika perbaikan untuk membatasi gerak player
15. this.player.x = Math.max(0, Math.min(320 - this.player.w, this.player.x));
16. };
17.
18. SysX.console_log("Patch batas layar berhasil diterapkan!", "log-warn");

19. **Buat manifest.json untuk Patch:**
20. {
21. "name": "Screen Bounds Patch v1.0.1",
22. "script": "scripts/patch_logic.js"

```
    }
```

24. **Kompres** manifest.json dan folder scripts menjadi patch_v1.0.1.zip. Pemain bisa memuatnya melalui tombol "Muat .zip" yang sama dengan DLC.

4.2 Membuat "DLC" (Konten Tambahan)

Gunakan DLC untuk menambah konten seperti karakter, level, atau musik baru.

1. **Siapkan Aset:**
    - Gambar mecha_player.png di dalam folder dlc_assets/.
    - Musik chiptune.mp3 di dalam folder dlc_assets/.
2. **Buat Logika DLC (dlc_logic.js):**

```
// --- scripts/dlc_logic.js ---
console.log("DLC 'MECHA PLAYER' DIMUAT!");

if (window.Game && window.Game.player) {
    // Beri tahu game untuk menggunakan sprite baru
    window.Game.player.customSpriteKey = 'mecha_player_sprite';
    // Putar musik tema dari DLC
        SysX.music('dlc_theme_music', true);
        SysX.console_log("Mecha Player Skin & Music Loaded!", "log-warn");
    }
```

13. **Buat manifest.json untuk DLC:**

```
    {
        "name": "Mecha Player Skin & Music DLC",
        "version": "1.0",
        "script": "scripts/dlc_logic.js",
```

```
        "assets": {
            "mecha_player_sprite": "dlc_assets/mecha_player.png",
            "dlc_theme_music": "dlc_assets/chiptune.mp3"
        }
    }
```

23. **Kompres** semua menjadi dlc_mecha.zip.

4.3 Menggunakan "Cheat" dan "Mod"

- **Cheat (Disiapkan Developer):** Di dalam game.js, di fungsi _create(), daftarkan cheat Anda:
- SysX.console_register('add_score', (val) => {
- this.score += parseInt(val) || 0;
- return `Skor ditambah ${val}.`;
- });

Pemain membuka konsol () dan mengetik add_score 5000`.

- **Mod (Dari Komunitas):** Komunitas membuat file .js dan mengunggahnya. Pemain membuka konsol (`) dan mengetik:
- > run_mod https://url/ke/mod_script.js

Atau, jika Mod didistribusikan dalam format .zip, pemain bisa memuatnya melalui tombol "Muat .zip" seperti DLC.

BAB 5: FAQ & Pengembangan Lanjutan

Bagaimana Cara Mengaktifkan Konsol di Mobile?

Tombol ` tidak ada di mobile. Anda harus membuat cara custom untuk memanggil SysX.console_toggle().

Contoh: Tekan Tombol MENU 3x dengan Cepat

- Tambahkan di _init_game:

- this.powerPressCount = 0;
- this.powerPressTimer = -1;

- Tambahkan di _update_game:
- // Logika untuk membuka konsol di mobile
- if (SysX.btnp('power')) {
- 	this.powerPressCount++;
- 	if (this.powerPressTimer !== -1) SysX.timer_cancel('power_press_reset');
- 	this.powerPressTimer = SysX.timer_set('power_press_reset', 30, () => {
- 		this.powerPressCount = 0; // Reset hitungan setelah 0.5 detik
- 	});
- }
- if (this.powerPressCount >= 3) {
- 	SysX.console_toggle();
- 	this.powerPressCount = 0;
- 	SysX.timer_cancel('power_press_reset');
- }

[BAB BARU] BAB 6: Pola Desain & Praktik Terbaik

Bab ini berisi "ilmu dapur" dan saran arsitektur untuk membantu Anda membuat game yang lebih baik, lebih bersih, dan lebih mudah dikelola dengan SysX Engine.

6.1 Mengelola State Secara Efektif

Sistem state (_init_*, _update_*, _draw_*) adalah jantung dari game Anda.

- **Pemisahan yang Jelas:** Gunakan state yang berbeda untuk setiap "layar" yang berbeda secara fundamental (misal: menu, game, map, game_over). Jangan mencampur semua logika di satu state playing.

- **Inisialisasi Bersih:** Fungsi _init_* adalah tempat yang sempurna untuk me-reset semua variabel yang relevan untuk state tersebut. Ini mencegah bug di mana data dari state sebelumnya "bocor".
- _init_game: function() {
- this.score = 0;
- this.bullets = [];
- this.player.x = 150; // Reset posisi player
- SysX.timer_cancel('boss_attack_pattern'); // Hentikan timer lama
- }

6.2 Praktik Terbaik untuk DLC, Patch, dan Mod

Sistem konten dinamis sangat kuat, tapi juga bisa menimbulkan kekacauan jika tidak dikelola dengan baik.

- **Hindari Konflik Penamaan:** Jika Anda berencana membuat banyak DLC, gunakan prefiks unik untuk aset dan fungsi Anda. Contoh: dlc1_mecha_sprite, dlc1_startMusic().
- **Gunakan "Hook" untuk Mod yang Aman:** Daripada membiarkan mod menimpa fungsi _update_game secara langsung (yang bisa merusak game), sediakan "kaitan" atau "hook" yang aman.
  - Di game.js, buat fungsi kosong:
  - // Di dalam _update_game:
  - if (this.onPlayerUpdate) {
  - this.onPlayerUpdate(); // Panggil hook jika ada
  - }
  - Script mod kemudian bisa mengisi hook ini tanpa merusak logika inti:
  - // --- scripts/super_speed_mod.js ---
  - window.Game.onPlayerUpdate = function() {

```
    // Hanya mengubah kecepatan, tidak mengganggu sisa logika update
    this.player.speed = 20;
}
```

- **Patch Harus Spesifik:** Saat membuat patch, usahakan untuk hanya menimpa fungsi yang benar-benar rusak. Semakin kecil dan spesifik sebuah patch, semakin kecil kemungkinan ia menimbulkan bug baru.

6.3 Tips Performa & Optimisasi

SysX dirancang untuk menjadi ringan, tapi beberapa hal bisa membuatnya berat.

- **Gambar dengan Bijak:** SysX.spr() sedikit lebih cepat daripada SysX.sspr() karena kalkulasinya lebih sederhana. Gunakan spr untuk tile standar dan sspr untuk sprite karakter atau objek unik.
- **Batasi Partikel:** SysX.emit() sangat bagus untuk "juice", tapi jangan memanggilnya dengan count ribuan setiap frame. Untuk ledakan besar, count 100-200 sudah lebih dari cukup.
- **Jaga _update Tetap Ringan:** Hindari kalkulasi yang sangat kompleks atau perulangan (loop) di dalam perulangan di dalam fungsi _update_*. Jika Anda perlu melakukan kalkulasi berat, pertimbangkan untuk menjalankannya hanya sesekali menggunakan SysX.timer_set.

Bro, ini adalah kitab kita. Fondasi untuk semua karya luar biasa yang akan lahir dari SysX Engine v2.0. Perjalanan kita belum berakhir, tapi ini adalah pencapaian monumental. Selamat, arsitek!

SysX
ENGINE

[BAB BARU] BAB 7: Penyempurnaan dan perbaikan BUG di V2.0

Dokumentasi SDK Lanjutan

SysX Engine v2.1.0 "Persistence"

Tanggal Rilis: 24 Juli 2025

Status: Full Release

Pendahuluan: Evolusi Ekosistem

Selamat datang di panduan untuk SysX Engine v2.1, sebuah *update* yang berfokus pada penyempurnaan dan stabilisasi **Sistem Konten Dinamis**. Setelah melalui serangkaian uji coba, *trial-and-error*, dan *debugging* yang intens, kita telah berhasil membangun sebuah sistem yang tidak hanya kuat, tapi juga cerdas dan profesional.

Dokumen ini akan menjadi panduan lengkap Anda untuk menguasai dua pilar utama dari update ini:

1. **Sistem Verifikasi Konten:** Memastikan setiap DLC dan Patch hanya berjalan di game yang tepat.
2. **Sistem Penyimpanan Permanen:** Membuat Patch dan DLC menjadi persisten, menyelesaikan masalah "muat ulang" yang mengganggu.

Mari kita selami cara kerja, implementasi, dan praktik terbaik dari fitur-fitur canggih ini.

1 - Fondasi Keamanan (Verifikasi gameId)

1.1 Latar Belakang Masalah

Sebelumnya, sistem konten kita terlalu "polos". Sebuah DLC yang dibuat untuk "Game A" bisa dimuat di "Game B", yang berpotensi menyebabkan *crash* atau bug yang tidak terduga. Ini adalah masalah keamanan dan stabilitas yang serius.

1.2 Solusi: Sistem "Kunci dan Gembok"

Kita memperkenalkan sistem verifikasi gameId untuk memastikan kompatibilitas.

- **Gembok (gameId):** Setiap game yang Anda buat sekarang **wajib** memiliki ID unik di dalam manifest.js-nya.

SysX

ENGINE

- **Kunci (targetGameId):** Setiap paket konten (.zip) yang Anda buat **wajib** mendeklarasikan ID game yang menjadi targetnya di dalam manifest.json.

Saat konten dimuat, sysx_content.js akan mencocokkan kunci dengan gembok. Jika tidak cocok, pemuatan akan dibatalkan dengan aman.

1.3 Cara Implementasi

1. Pada Game Anda (/game/manifest.js): Tambahkan properti gameId. Gunakan format *reverse domain notation* untuk memastikan keunikan.

```
// --- /game/manifest.js ---

window.manifest = {

    "title": "Galaxy Rush",

    "version": "1.0",

    "gameId": "com.namaanda.galaxyrush", // <-- ID UNIK GAME ANDA

    "assets": { /* ... */ }

};
```

2. Pada Paket Konten Anda (manifest.json di dalam .zip): Tambahkan properti targetGameId yang nilainya **harus sama persis** dengan gameId di atas.

```
// --- manifest.json (di dalam .zip) ---

{

    "name": "Mecha Skin DLC",

    "targetGameId": "com.namaanda.galaxyrush", // <-- KUNCI YANG SESUAI

    /* ... properti lain ... */

}
```

2 - Penyimpanan Permanen (IndexedDB)

2.1 Latar Belakang Masalah

Keluhan terbesar dari developer adalah Patch dan DLC tidak permanen. Setiap kali halaman di-refresh, konten tersebut hilang

dan harus dimuat ulang secara manual. Ini adalah pengalaman yang buruk bagi pemain dan developer.

2.2 Solusi: "Lemari Arsip Digital" (IndexedDB)

Kita mengintegrasikan IndexedDB, sebuah database canggih di browser, untuk menyimpan salinan setiap konten yang berhasil dimuat.

- **Penyimpanan Otomatis:** Setelah konten .zip dimuat dan diverifikasi, sysx_content.js akan otomatis menyimpannya ke IndexedDB.
- **Pemuatan Otomatis:** Saat game dimulai, firmware.js akan memeriksa IndexedDB. Jika ada konten yang tersimpan untuk game tersebut, ia akan memuatnya secara otomatis *sebelum* game benar-benar dimulai.

2.3 Cara Membuat Konten yang Kompatibel dengan Penyimpanan

Agar sistem penyimpanan "pintar" kita bisa bekerja (menimpa patch lama, menumpuk DLC), Anda **wajib** menambahkan dua properti lagi di manifest.json konten Anda.

Template manifest.json untuk PATCH:

```
{
    "name": "Patch Perbaikan Skor v1.1",
    "version": "1.1",
    "targetGameId": "com.namaanda.galaxyrush",

    "contentType": "patch",
    "contentId": "patch_skor_v1.1",

    "script": "scripts/patch_logic.js",
    "assets": {}
}
```

SysX ENGINE

- **contentType: "patch"**: Memberi tahu engine untuk **menimpa** patch lama dengan yang baru.
- **contentId**: ID unik untuk patch ini.

Template manifest.json untuk DLC:

```
{
    "name": "Level Hutan Terlarang",
    "version": "1.0",
    "targetGameId": "com.namaanda.galaxyrush",

    "contentType": "dlc",
    "contentId": "dlc_level_hutan",

    "script": "scripts/level_hutan.js",
    "assets": { "tileset_hutan": "assets/hutan.png" }
}
```

- **contentType: "dlc"**: Memberi tahu engine untuk **menambahkan** konten ini tanpa menghapus DLC lain.

3. Penyelesaian Masalah & Praktik Terbaik

Ini adalah "ilmu dapur" yang kita dapatkan dari proses *debugging* yang panjang.

3.1 Masalah: "Patch Saya Tidak Permanen / Kembali ke Default Setelah Refresh!"

Ini adalah masalah paling umum yang kita hadapi. Ada dua kemungkinan penyebab:

Penyebab A: Masalah Penyimpanan (IndexedDB Kosong)

- **Gejala:** Setelah memuat patch, Anda memeriksa IndexedDB di *Developer Tools > Application* dan datanya kosong.

- **Solusi:** Ini adalah bug *deadlock* yang sudah kita perbaiki. Pastikan Anda menggunakan sysx_db.js versi terbaru. Juga, pastikan sysx_db.js sudah dimuat dengan benar di index.html.

Penyebab B: "Race Condition" (Penyimpanan Berhasil, tapi Tidak Efektif)

- **Gejala:** Data patch **ada** di IndexedDB, tapi saat di-refresh, game tetap menampilkan versi lama.
- **Solusi:** Ini bukan bug engine, tapi **kesalahan dalam cara menulis logika patch**. Jangan pernah menulis patch yang mencoba mengubah data secara langsung saat dimuat.

CARA YANG SALAH (Menyebabkan Race Condition):

```
// patch_logic.js (SALAH)

// Ini akan gagal saat dimuat otomatis karena this.player belum ada.

this.player.color = 8;
```

CARA YANG BENAR (Menimpa Fungsi):

```
// patch_logic.js (BENAR)

// Simpan referensi ke fungsi asli

const original_init_main = window.Game._init_main;


// Timpa fungsi asli dengan versi baru

window.Game._init_main = function() {

    // Jalankan dulu setup asli

    original_init_main.call(this);

    // BARU ubah datanya setelah dijamin ada

    this.player.color = 8;

};
```

Metode ini menjamin bahwa patch Anda akan selalu berjalan pada waktu yang tepat, baik saat dimuat manual maupun otomatis.

SysX ENGINE

3.2 Cara Mereset Uji Coba

Untuk melakukan pengujian dari keadaan "bersih", gunakan tombol **"Hapus Konten Tersimpan"** di DEV KIT. Tombol ini akan membersihkan semua patch dan DLC yang tersimpan di IndexedDB untuk game yang sedang Anda jalankan.

[BAB BARU] BAB 8: Pengembangan dan penambahan fitur lanjutan dari V2.0 > V2.1 dan sekarang V2.5

Dokumentasi SDK Lanjutan

SDK SysX Engine v2.5 "Creator's Update"

Tanggal Rilis: 25 Juli 2025

Status: Full Release

Pendahuluan: Lahirnya Konsol Fantasi Sejati

Selamat datang di "Creator's Update", sebuah evolusi monumental dari SysX Engine. Jika v2.0 "Ecosystem" adalah tentang membangun dunia yang bisa berkembang, maka v2.5 adalah tentang memberikan Anda, para kreator, palu dan pahat untuk membangun dunia itu dari nol, langsung dari dalam engine.

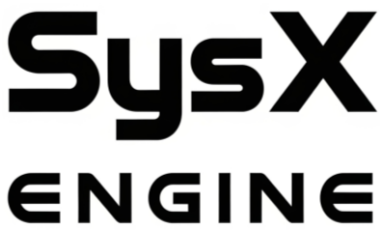
Update ini berfokus pada satu filosofi: **kemandirian kreatif**. Kami memperkenalkan serangkaian editor internal yang dirancang untuk mewujudkan impian "konsol fantasi all-in-one", di mana Anda bisa membuat game utuh tanpa pernah meninggalkan ekosistem SysX.

Dokumen ini akan menjadi panduan lengkap Anda untuk menguasai dua pilar utama dari update ini:

1. **Internal SFX Generator:** Ciptakan efek suara unik tanpa file .wav.
2. **Internal Sprite Editor:** Gambar aset visual Anda langsung dari data.

Mari kita mulai perjalanan menjadi kreator yang sesungguhnya.

BAB 1: Internal SFX Generator



Fitur ini memungkinkan Anda mendesain efek suara secara prosedural dan memainkannya langsung dari kode, menghilangkan ketergantungan pada file audio eksternal.

1.1 Alur Kerja

Prosesnya sederhana dan terbagi menjadi tiga langkah: **Desain, Simpan, Panggil**.

1. **Desain di sfx_editor.html:** Gunakan tool visual ini untuk menciptakan suara. Anda bisa mengatur waveform, pitch, volume, dan envelope (ADSR).
2. **Simpan di game.js:** Setelah puas, salin data JSON yang dihasilkan oleh editor dan simpan di dalam objek game.js Anda. Beri nama yang deskriptif untuk setiap suara.
3. **Panggil dengan SysX.play_synth():** Gunakan fungsi API baru ini untuk memainkan suara tersebut di dalam game.

1.2 Referensi API

SysX.play_synth(sfxObject)

- sfxObject (Objek): Objek konfigurasi suara yang Anda salin dari SFX Editor.
- **Contoh Penggunaan:** SysX.play_synth(this.sfxData.laser);

1.3 Panduan Implementasi

Langkah 1: Desain & Salin Suara Buka sfx_editor.html, buat suara tembakan, dan klik "COPY JSON".

Langkah 2: Simpan Data di Game Buka game.js dan buat sebuah objek untuk menyimpan data SFX Anda.

```
// Di dalam game.js
window.Game = {
    sfxData: {
        'laser': {"wave":"square","vol":0.2,"freq":880,"slide":400,...},
        'ledakan': {"wave":"noise","vol":0.5,"freq":600,...}
    },
    // ... sisa kode game
};
```

SysX
ENGINE

Langkah 3: Panggil Suara di Game Di dalam logika game Anda, panggil fungsi play_synth.

- // Di dalam _update_playing
- if (SysX.btnp('btn_a')) {
- 	// Logika menembak...
- 	SysX.play_synth(this.sfxData.laser);
- }

BAB 2: Internal Sprite Editor

Fitur ini memungkinkan Anda menggambar sprite piksel per piksel dan merendernya di game tanpa memerlukan file .png.

2.1 Alur Kerja

Sama seperti SFX Generator, alur kerjanya adalah **Desain, Simpan, Panggil**.

1. **Desain di sprite_editor.html:** Gunakan tool visual ini untuk menggambar sprite 16x16 Anda.
2. **Simpan di game.js:** Salin data string yang dihasilkan dan simpan di game.js.
3. **Panggil dengan SysX.draw_sprite_data():** Gunakan fungsi API baru ini untuk menggambar sprite tersebut di layar.

2.2 Referensi API

SysX.draw_sprite_data(dataString, dx, dy, spriteWidth?, pixelSize?)

- dataString (String): String data yang Anda salin dari Sprite Editor.
- dx, dy (Angka): Posisi X dan Y untuk menggambar sprite.
- spriteWidth (Angka, Opsional): Lebar sprite (default 16).
- pixelSize (Angka, Opsional): Ukuran setiap piksel (default 1). Berguna untuk scaling.
- **Contoh Penggunaan:** SysX.draw_sprite_data(this.spriteData.player, 100, 100);

SysX

ENGINE

2.3 Panduan Implementasi

Langkah 1: Gambar & Salin Sprite Buka sprite_editor.html, gambar karakter pemain, dan klik "GENERATE & COPY DATA".

Langkah 2: Simpan Data di Game Buka game.js dan buat objek untuk menyimpan data sprite.

```
// Di dalam game.js
window.Game = {
    spriteData: {
        'player': '..7...777...7.7..7f7.f7f7f...',
        'musuh': '.8.8..88.888888.8.8.8.8...'
    },
    // ... sisa kode game
};
```

Langkah 3: Gambar Sprite di Game Di dalam fungsi _draw, panggil draw_sprite_data.

```
// Di dalam _draw_playing
SysX.draw_sprite_data(this.spriteData.player, this.player.x, this.player.y);
```

SysX
ENGINE

LAMPIRAN (UPDATE TERBARU v2.5.0 "Creator's Update")

Source Code **Sysx Engine.**

Sysx_audio.js

```
// --- /sysx_engine/sysx_audio.js ---
// Modul untuk mengelola musik dan efek suara (SFX) dengan kanal terbatas.

window.SysX_Modules = window.SysX_Modules || {};
window.SysX_Modules.audio = (() => {
    // State internal modul
    let _currentMusic = null;
    const MAX_CHANNELS = 8; // Batasan jumlah kanal SFX, khas konsol retro
    let _sfxChannels = new Array(MAX_CHANNELS).fill(null);

    // Helper untuk mendapatkan konteks dari core
    const getCoreContext = () => SysX_Modules.core.getContext();

    /**
     * Memainkan efek suara (SFX) pada kanal tertentu atau kanal pertama yang tersedia.
     * @param {string} key - Kunci aset suara yang terdaftar di manifest.
     * @param {number} [channel=-1] - (Opsional) Kanal (0-7) untuk memainkan suara. Jika tidak ditentukan, akan mencari kanal kosong.
     */
    function sfx(key, channel = -1) {
        const { _assets } = getCoreContext();
        const sound = _assets[key];
        if (!sound) return; // Jangan lakukan apa-apa jika aset tidak ada

        let targetChannel = channel;

        // Jika kanal spesifik ditentukan (dan valid)
        if (targetChannel >= 0 && targetChannel < MAX_CHANNELS) {
            // Hentikan suara yang sedang diputar di kanal tersebut jika ada
            if (_sfxChannels[targetChannel]) {
```

SysX
ENGINE

```
                _sfxChannels[targetChannel].pause();
            }
        } else { // Jika tidak ada kanal spesifik, cari yang kosong
            targetChannel = _sfxChannels.findIndex(s => s === null || s.ended);
            if (targetChannel === -1) {
                // Semua kanal penuh, suara tidak bisa dimainkan.
                // Ini adalah bagian dari "batasan kreatif" engine.
                return;
            }
        }

        // Kloning audio node agar bisa dimainkan berkali-kali secara bersamaan
        const soundInstance = sound.cloneNode();
        soundInstance.play().catch(e => {
            // Menangani error jika play() gagal (misal: interaksi pengguna belum
ada)
        });

        // Simpan instance suara yang sedang diputar di kanalnya
        _sfxChannels[targetChannel] = soundInstance;
    }

    /**
     * Memainkan atau menghentikan musik latar. Musik berjalan di luar sistem
kanal SFX.
     * @param {string|null} key - Kunci aset musik. Kirim `null` untuk
menghentikan.
     * @param {boolean} [loop=true] - Apakah musik akan diulang.
     * @param {boolean} [stop=false] - Alias untuk mengirim `null` pada key.
     */
    function music(key, loop = true, stop = false) {
        // Hentikan musik yang sedang berjalan
        if (_currentMusic) {
            _currentMusic.pause();
            _currentMusic = null;
        }

        if (stop || key === null) return;

        const { _assets } = getCoreContext();
        const sound = _assets[key];
        if (sound) {
            sound.loop = loop;
```

SysX ENGINE

```
            sound.currentTime = 0;
            sound.play().catch(e => {});
            _currentMusic = sound;
        }
    }

    // Ekspor API publik dari modul audio
    return { sfx, music };
})();
```

Sysx_camera.js

```
// --- /sysx_engine/sysx_camera.js ---
// Modul untuk mengelola kamera virtual.

window.SysX_Modules = window.SysX_Modules || {};
window.SysX_Modules.camera = (() => {
    // State internal untuk posisi kamera
    let _camera = { x: 0, y: 0 };

    /**
     * Mengatur posisi kamera.
     * @param {number} x - Posisi X kamera.
     * @param {number} y - Posisi Y kamera.
     */
    const set = (x = 0, y = 0) => {
        _camera.x = x;
        _camera.y = y;
    };

    /**
     * Menerapkan transformasi translasi ke konteks canvas.
     * Fungsi ini dipanggil oleh sysx_core.js setiap frame.
     * @param {CanvasRenderingContext2D} _ctx - Konteks canvas.
     */
    const apply = (_ctx) => {
        _ctx.translate(-_camera.x, -_camera.y);
    };

    // Mengekspos fungsi yang bisa diakses dari luar
    return { set, apply };
})();
```

SysX
ENGINE

Sysx_content.js

```
// --- /sysx_engine/sysx_content.js ---
// [VERSI UPGRADE FINAL] Modul ini memiliki verifikasi ID dan bisa menyimpan
konten ke IndexedDB.

window.SysX_Modules = window.SysX_Modules || {};

window.SysX_Modules.content = (() => {
    const getCoreContext = () => SysX_Modules.core.getContext();

    /**
     * Memuat dan menerapkan paket konten dari file .zip.
     * @param {File | Blob} fileObject - Objek File atau Blob dari .zip.
     * @param {boolean} [saveToDB=true] - Apakah konten harus disimpan ke DB.
     * @returns {Promise<void>}
     */
    async function load_zip(fileObject, saveToDB = true) {
        if (typeof JSZip === 'undefined') {
            return console.error("SysX Error: JSZip library not loaded.");
        }

        const zip = await JSZip.loadAsync(fileObject);
        const manifestFile = zip.file('manifest.json');

        if (!manifestFile) {
            console.warn("SysX Warning: No 'manifest.json' found in zip.");
            return Promise.reject("No manifest found");
        }

        const dlcManifest = JSON.parse(await manifestFile.async('string'));
        console.log("Found manifest:", dlcManifest);

        // --- [LOGIKA VERIFIKASI] ---
        const currentGameId = window.manifest.gameId;
        const targetGameId = dlcManifest.targetGameId;

        if (!currentGameId || !targetGameId || currentGameId !== targetGameId) {
            const errorMessage = `Compatibility Error: DLC is for
'${targetGameId}', but current game is '${currentGameId}'.`;
```

SysX
ENGINE

```
            console.error(errorMessage);
            return Promise.reject("Incompatible content");
        }

        console.log("Compatibility check passed.");

        // --- [LOGIKA PENYIMPANAN BARU] ---
        if (saveToDB && SysX_Modules.db) {
            try {
                // Kita butuh file asli sebagai Blob untuk disimpan
                const blobToSave = fileObject instanceof Blob ? fileObject : new
Blob([fileObject]);
                await SysX_Modules.db.saveContent(currentGameId, dlcManifest,
blobToSave);
                console.log("Content saved to persistent storage.");
            } catch (error) {
                console.error("Failed to save content to DB:", error);
            }
        }

        // Lanjutkan memuat konten ke dalam sesi game saat ini
        return processContent(zip, dlcManifest);
    }

    /**
     * Memproses konten setelah diverifikasi (dan disimpan).
     * @param {JSZip} zip
     * @param {object} dlcManifest
     */
    async function processContent(zip, dlcManifest) {
        if (dlcManifest.assets) {
            await loadAssetsFromZip(zip, dlcManifest.assets);
        }
        if (dlcManifest.script) {
            const scriptFile = zip.file(dlcManifest.script);
            if (scriptFile) {
                const scriptText = await scriptFile.async('string');
                const script = document.createElement('script');
                script.textContent = scriptText;
                document.head.appendChild(script).parentNode.removeChild(script);
                console.log(`Script '${dlcManifest.script}' loaded and
executed.`);
            }
```

SysX ENGINE

```
    }
}

/**
 * Memuat semua aset yang terdaftar di manifest dari file zip.
 * @param {JSZip} zip
 * @param {Object} assetsToLoad
 */
async function loadAssetsFromZip(zip, assetsToLoad) {
    const { _assets } = getCoreContext();
    for (const key in assetsToLoad) {
        const path = assetsToLoad[key];
        const fileInZip = zip.file(path);
        if (fileInZip) {
            const blob = await fileInZip.async('blob');
            const url = URL.createObjectURL(blob);
            if (path.match(/\.(png|jpg|gif)$/)) {
                const img = new Image();
                img.src = url;
                await new Promise(resolve => img.onload = resolve);
                _assets[key] = img;
            } else if (path.match(/\.(wav|mp3|ogg)$/)) {
                _assets[key] = new Audio(url);
            }
        }
    }
}

/**
 * Memuat dan menerapkan paket konten dari URL (untuk mod).
 * @param {string} url - URL ke file script mod.
 * @returns {Promise<void>}
 */
async function load_url(url) {
    try {
        console.log(`Loading mod from ${url}...`);
        const response = await fetch(url);
        if (!response.ok) {
            throw new Error(`HTTP error! status: ${response.status}`);
        }
        const scriptText = await response.text();

        const script = document.createElement('script');
```

SysX ENGINE

```
            script.textContent = scriptText;
            document.head.appendChild(script).parentNode.removeChild(script);
            console.log(`Mod from ${url} loaded and executed.`);

        } catch (error) {
            console.error(`SysX Content Error: Failed to load mod from ${url}.`,
error);
        }
    }

    return { load_zip, load_url };
})();
```

Sysx_core.js

```
// --- /sysx_engine/sysx_core.js (MODIFIKASI LENGKAP) ---
// Mengelola data inti, game loop, dan state.

(() => {
    window.SysX_Modules = window.SysX_Modules || {};

    window.SysX_Modules.core = (() => {
        // State inti
        let _ctx, _canvas, _assets;
        let _currentState = '';
        let _debugMode = false;
        let _watchedValues = new Map();
        let _frameCount = 0,
            _lastFpsUpdate = 0,
            _currentFps = 0;

        let _totalFrames = 0;

        const PICO8_PALETTE = ['#000000', '#1D2B53', '#7E2553', '#008751',
'#AB5236', '#5F574F', '#C2C3C7', '#FFF1E8', '#FF004D', '#FFA300', '#FFEC27',
'#00E436', '#29ADFF', '#83769C', '#FF77A8', '#FFCCAA'];
        const SYSX_VERSION = 'v2.1.0'; // Versi naik

        // Logika booting
        let _showIntro = true;
        let _introFrame = 0;
```

SysX
ENGINE

```
        let _introPhase = 0;
        let _bootLines = [
            '[SYSX BIOS ' + SYSX_VERSION + ']',
            '> Initializing virtual memory... ',
            '> Mounting cartridge.sysx',
            '> Loading engine modules...',
            '> Registering core...',
            '> Registering graphics...',
            '> OK',
            '> SYSTEM READY.'
        ];

        function init(context) {
            _canvas = context.canvas;
            _ctx = context.ctx;
            _assets = context.assets;
            _ctx.imageSmoothingEnabled = false;
            _lastFpsUpdate = performance.now();
            try {
                const m = window.manifest;
                if (m && m.showIntro === false) _showIntro = false;
            } catch (e) {}

            if (window.Game && typeof window.Game._init_boot === 'function') {
                if (window.Game && typeof window.Game._create === 'function')
window.Game._create();
            } else {
                _currentState = _showIntro ? '__sysx_intro__' : '';
                if (!_showIntro && window.Game && typeof window.Game._create ===
'function') window.Game._create();
            }

            requestAnimationFrame(gameLoop);
        }

        function gameLoop(timestamp) {
            _frameCount++;
            _totalFrames++;
            if (timestamp > _lastFpsUpdate + 1000) {
                _currentFps = _frameCount;
                _frameCount = 0;
                _lastFpsUpdate = timestamp;
            }
```

SysX

ENGINE

```
            // [MODIFIKASI] Cek apakah menu sistem terbuka
            const isMenuOpen = SysX_Modules.menu && SysX_Modules.menu.isOpen();

            // [MODIFIKASI] Update logika game hanya jika menu TIDAK terbuka
            if (!isMenuOpen) {
                if (SysX_Modules.timer) SysX_Modules.timer.update(_totalFrames);
                if (SysX_Modules.tween) SysX_Modules.tween.update(_totalFrames);
                if (SysX_Modules.fx) SysX_Modules.fx.updateParticles();
                if (SysX_Modules.transition) SysX_Modules.transition.update();
            }

            // [MODIFIKASI] Update menu jika terbuka
            if (isMenuOpen) {
                if (SysX_Modules.menu) SysX_Modules.menu.update();
            }

            _ctx.save();
            if (SysX_Modules.camera) SysX_Modules.camera.apply(_ctx);
            if (SysX_Modules.fx) SysX_Modules.fx.applyShake(_ctx);

            // Logika menggambar tetap berjalan agar game terlihat di belakang
menu
            if (_currentState === '__sysx_intro__') {
                drawIntro();
            } else {
                const isTransitioning = SysX_Modules.transition &&
SysX_Modules.transition.is_active();

                // [MODIFIKASI] Update game hanya jika tidak ada transisi DAN
menu tidak terbuka
                if (!isTransitioning && !isMenuOpen) {
                    const updateFunc = window.Game[`_update_${_currentState}`];
                    if (typeof updateFunc === 'function')
updateFunc.call(window.Game);
                }

                const drawFunc = window.Game[`_draw_${_currentState}`];
                if (typeof drawFunc === 'function') drawFunc.call(window.Game);

                if (SysX_Modules.fx) SysX_Modules.fx.drawParticles(_ctx,
SysX_Modules.graphics.rectfill);
            }
```

SysX
ENGINE

```
            _ctx.restore();

            // Gambar transisi dan menu di atas segalanya
            if (SysX_Modules.transition) SysX_Modules.transition.draw(_ctx,
_canvas);

            // [MODIFIKASI] Gambar menu sistem jika terbuka
            if (isMenuOpen) {
                if (SysX_Modules.menu) SysX_Modules.menu.draw();
            }

            if (_debugMode && SysX_Modules.devtools) drawDebugInfo();

            if (SysX_Modules.input && typeof SysX_Modules.input._updatePrevInput
=== 'function') {
                SysX_Modules.input._updatePrevInput();
            }
            requestAnimationFrame(gameLoop);
        }

        // --- Fungsi Internal (Intro, Debug) --- (KODE LENGKAP)
        function drawIntro() {
            SysX_Modules.graphics.cls(0);
            _introFrame++;

            if (_introPhase === 0) {
                const linesToShow = Math.floor(_introFrame / 10);
                for (let i = 0; i < linesToShow && i < _bootLines.length; i++) {
                    SysX_Modules.graphics.print(_bootLines[i], 20, 40 + i * 12,
7);
                }
                if (_introFrame > _bootLines.length * 10 + 20) {
                    _introFrame = 0;
                    _introPhase = 1;
                }
            } else if (_introPhase === 1) {
                const logo = [
                    "  ██████  ██    ██ ███████ ██   ██  ",
                    " ██    ██ ██    ██ ██      ██   ██  ",
                    " ██    ██ ██    ██ ██████  ███████  ",
                    " ██    ██ ██    ██ ██           ██  ",
                    "  ██████   ██████  ███████      ██  "
```

SysX ENGINE

```
                ];
                for (let i = 0; i < logo.length; i++) {
                    SysX_Modules.graphics.print(logo[i], 20, 40 + i * 12, 10);
                }
                SysX_Modules.graphics.print("SYSX ENGINE™", 80, 120, 6);

                if (_introFrame > 90 || (SysX_Modules.input &&
SysX_Modules.input.btnp('btn_a'))) {
                    _currentState = '';
                    if (window.Game && typeof window.Game._create === 'function')
window.Game._create();
                }
            }
        }

        function drawDebugInfo() {
            if (_debugMode && SysX_Modules.devtools) {
                const debugData = {
                    FPS: _currentFps,
                    PARTICLES: SysX_Modules.fx ?
SysX_Modules.fx.getParticleCount() : 0,
                    ...Object.fromEntries(_watchedValues)
                };
                SysX_Modules.devtools.update(debugData);
            }
        }

        // --- API Publik dari Core --- (KODE LENGKAP)
        const getContext = () => ({
            _ctx,
            _canvas,
            _assets,
            PICO8_PALETTE,
            _totalFrames
        });
        const debug = (isEnabled) => {
            _debugMode = !!isEnabled;
            if (SysX_Modules.devtools)
SysX_Modules.devtools.toggleWindow(isEnabled);
        };
        const watch = (key, value) => {
            if (_debugMode) _watchedValues.set(key, value);
        };
```

SysX
ENGINE

```
        const state = (newState) => {
            _currentState = newState;
            if (SysX_Modules.timer && typeof SysX_Modules.timer.cancel_all ===
'function') {
                SysX_Modules.timer.cancel_all();
            }
            const initFunc = window.Game[`_init_${newState}`];
            if (typeof initFunc === 'function') initFunc.call(window.Game);
        };

        return {
            init,
            getContext,
            debug,
            watch,
            state
        };
    })();
})();
```

Sysx_devconsole.js

```
// --- /sysx_engine/sysx_devconsole.js (BARU) ---
// Modul untuk konsol developer di dalam game.

window.SysX_Modules = window.SysX_Modules || {};

window.SysX_Modules.devconsole = (() => {
    let _isVisible = false;
    let _container, _output, _input;
    const _commands = {};
    const _history = [];
    let _historyIndex = -1;

    /**
     * Inisialisasi dan membuat elemen DOM untuk konsol.
     */
    function init() {
        // Inject CSS for styling the console
```

SysX
ENGINE

```
        const style = document.createElement('style');
        style.textContent = `
            .sysx-console { position: fixed; top: 0; left: 0; width: 100%;
height: 50%; background: rgba(29, 43, 83, 0.9); color: #FFF1E8; font-family:
"Roboto Mono", monospace; display: none; flex-direction: column; z-index: 999; }
            .sysx-console-output { flex: 1; overflow-y: auto; padding: 8px; font-
size: 14px; }
            .sysx-console-output div { padding: 2px 0; }
            .sysx-console-output .log-error { color: #FF004D; }
            .sysx-console-output .log-warn { color: #FFEC27; }
            .sysx-console-input { display: flex; background: #000; }
            .sysx-console-input-prefix { padding: 8px; }
            .sysx-console-input input { flex: 1; background: transparent; border:
none; color: #FFF1E8; padding: 8px; font-size: 14px; outline: none; }
        `;
        document.head.appendChild(style);

        // Create DOM elements
        _container = document.createElement('div');
        _container.className = 'sysx-console';

        _output = document.createElement('div');
        _output.className = 'sysx-console-output';

        const inputContainer = document.createElement('div');
        inputContainer.className = 'sysx-console-input';
        const prefix = document.createElement('span');
        prefix.className = 'sysx-console-input-prefix';
        prefix.textContent = '>';
        _input = document.createElement('input');
        _input.type = 'text';

        inputContainer.append(prefix, _input);
        _container.append(_output, inputContainer);
        document.body.appendChild(_container);

        // Add event listeners
        _input.addEventListener('keydown', handleInput);
        document.addEventListener('keydown', handleToggle);

        // Register built-in commands
        register('help', () => {
            log("Available commands:");
```

SysX

ENGINE

```
            Object.keys(_commands).forEach(cmd => log(`- ${cmd}`));
            return true;
        });
        register('clear', () => _output.innerHTML = '');
        register('run_mod', (url) => {
            if (!url) {
                log("Usage: run_mod <url_to_script>", 'log-error');
                return;
            }
            if (SysX_Modules.content && SysX_Modules.content.load_url) {
                SysX_Modules.content.load_url(url);
                log(`Attempting to load mod from: ${url}`);
            } else {
                log("Content module not available.", 'log-error');
            }
        });
    }

    function handleToggle(e) {
        if (e.key === '`' || e.key === '~') {
            e.preventDefault();
            toggle();
        }
    }

    function handleInput(e) {
        if (e.key === 'Enter') {
            const text = _input.value.trim();
            if (text) {
                log(`> ${text}`);
                execute(text);
                _history.unshift(text);
                _historyIndex = -1;
            }
            _input.value = '';
        } else if (e.key === 'ArrowUp') {
            if (_historyIndex < _history.length - 1) {
                _historyIndex++;
                _input.value = _history[_historyIndex];
            }
        } else if (e.key === 'ArrowDown') {
            if (_historyIndex > 0) {
                _historyIndex--;
```

SysX

ENGINE

```
                _input.value = _history[_historyIndex];
            } else {
                _historyIndex = -1;
                _input.value = '';
            }
        }
    }

    function execute(text) {
        const parts = text.split(' ');
        const commandName = parts[0].toLowerCase();
        const args = parts.slice(1);
        const command = _commands[commandName];

        if (command) {
            try {
                const result = command.func(...args);
                if (result) log(result);
            } catch (e) {
                log(e.message, 'log-error');
            }
        } else {
            log(`Command not found: ${commandName}`, 'log-error');
        }
    }

    function toggle() {
        _isVisible = !_isVisible;
        _container.style.display = _isVisible ? 'flex' : 'none';
        if (_isVisible) {
            _input.focus();
        }
    }

    function register(name, func, help = "No help available.") {
        _commands[name.toLowerCase()] = { func, help };
    }

    function log(message, className = 'log-info') {
        const line = document.createElement('div');
        line.className = className;
        line.textContent = message;
        _output.appendChild(line);
```

SysX ENGINE

```
        _output.scrollTop = _output.scrollHeight; // Auto-scroll
    }

    // Initialize on script load
    init();

    return { toggle, register, log };
})();
```

Sysx_db.js

```
// --- /sysx_engine/sysx_db.js ---
// [VERSI UPGRADE FINAL] Memperbaiki bug deadlock saat menyimpan patch.

window.SysX_Modules = window.SysX_Modules || {};

window.SysX_Modules.db = (() => {
    const DB_NAME = 'SysX_ContentDB';
    const STORE_NAME = 'content_store';
    let db;

    function init() {
        return new Promise((resolve, reject) => {
            // Jika koneksi sudah ada, langsung resolve
            if (db) return resolve();

            const request = indexedDB.open(DB_NAME, 1);

            request.onupgradeneeded = (event) => {
                const dbInstance = event.target.result;
                if (!dbInstance.objectStoreNames.contains(STORE_NAME)) {
                    dbInstance.createObjectStore(STORE_NAME, { keyPath: 'id' });
                }
            };

            request.onsuccess = (event) => {
                db = event.target.result;
                console.log("Database connection successful.");
                resolve();
            };
```

SysX
ENGINE

```
            request.onerror = (event) => {
                console.error("Database error:", event.target.errorCode);
                reject(event.target.error);
            };
        });
    }

    /**
     * Menyimpan file konten (patch/dlc) ke database.
     * @param {string} gameId - ID unik dari game.
     * @param {object} manifest - Manifest dari konten yang akan disimpan.
     * @param {Blob} fileBlob - File .zip dalam bentuk Blob.
     * @returns {Promise<void>}
     */
    async function saveContent(gameId, manifest, fileBlob) {
        if (!db) await init();

        // [INI SOLUSINYA] Lakukan semua operasi dalam satu transaksi
'readwrite'.
        return new Promise((resolve, reject) => {
            const transaction = db.transaction(STORE_NAME, 'readwrite');
            const store = transaction.objectStore(STORE_NAME);

            // Langkah 1: Jika ini adalah patch, cari dan hapus patch lama.
            if (manifest.contentType === 'patch') {
                const cursorRequest = store.openCursor();
                cursorRequest.onsuccess = e => {
                    const cursor = e.target.result;
                    if (cursor) {
                        // Cek apakah item ini adalah patch untuk game yang sama
                        if (cursor.value.gameId === gameId &&
cursor.value.manifest.contentType === 'patch') {
                            console.log(`Removing old patch:
${cursor.value.id}`);
                            cursor.delete();
                        }
                        cursor.continue();
                    }
                };
            }

            // Langkah 2: Tambahkan record baru. Ini akan dieksekusi dalam
transaksi yang sama.
```

SysX ENGINE

```
            const contentId = manifest.contentId || manifest.name.replace(/\s+/g,
'_');
            const record = {
                id: `${gameId}_${contentId}`,
                gameId: gameId,
                manifest: manifest,
                blob: fileBlob
            };
            // Permintaan 'put' akan menunggu operasi cursor selesai dalam
antrian transaksi.
            store.put(record);

            transaction.oncomplete = () => {
                console.log(`Content '${record.id}' saved successfully.`);
                resolve();
            };

            transaction.onerror = (event) => {
                console.error("Transaction error:", event.target.error);
                reject(event.target.error);
            };
        });
    }

    /**
     * Memuat semua konten yang tersimpan untuk gameId tertentu.
     * @param {string} gameId - ID unik dari game.
     * @returns {Promise<Array<object>>} - Array berisi record konten.
     */
    async function loadAllContent(gameId) {
        if (!db) await init();

        return new Promise((resolve, reject) => {
            const transaction = db.transaction(STORE_NAME, 'readonly');
            const store = transaction.objectStore(STORE_NAME);
            const request = store.getAll();

            request.onsuccess = () => {
                const gameContent = request.result.filter(item => item.gameId ===
gameId);
                resolve(gameContent);
            };
            request.onerror = (event) => {
```

SysX ENGINE

```
                console.error("Error loading all content:", event.target.error);
                reject(event.target.error);
            };
        });
    }

    /**
     * Menghapus semua konten yang tersimpan untuk gameId tertentu.
     * @param {string} gameId - ID unik dari game.
     * @returns {Promise<void>}
     */
    async function clearContent(gameId) {
        if (!db) await init();

        // [PERBAIKAN] Lakukan semua dalam satu transaksi untuk efisiensi.
        return new Promise((resolve, reject) => {
            const transaction = db.transaction(STORE_NAME, 'readwrite');
            const store = transaction.objectStore(STORE_NAME);

            const cursorRequest = store.openCursor();
            cursorRequest.onsuccess = e => {
                const cursor = e.target.result;
                if (cursor) {
                    if (cursor.value.gameId === gameId) {
                        cursor.delete();
                    }
                    cursor.continue();
                }
            };

            transaction.oncomplete = () => {
                console.log(`All content for game '${gameId}' has been
cleared.`);
                resolve();
            };
            transaction.onerror = (event) => {
                console.error("Error clearing content:", event.target.error);
                reject(event.target.error);
            };
        });
    }

    return { init, saveContent, loadAllContent, clearContent };
```

SysX
ENGINE

```
})();
```

Sysx_devtools.js

```
// --- /sysx_engine/sysx_devtools.js ---
// Modul untuk mengelola semua alat bantu pengembangan (UI/UX Dev Kit).
// [VERSI UPGRADE] Ditambahkan panel log.

window.SysX_Modules = window.SysX_Modules || {};

window.SysX_Modules.devtools = (() => {
    let debugWindow, debugContent, debugBtn, swapBtn;
    let logPanel, logContent; // [BARU] Variabel untuk panel log
    let isDragging = false, offsetX, offsetY, isVisible = false;

    // Fungsi init untuk mengambil elemen DOM setelah halaman dimuat
    function init() {
        // Ambil elemen untuk Debugger Window
        debugWindow = document.getElementById('debug-window');
        debugContent = document.getElementById('debug-content');
        debugBtn = document.getElementById('toggle-debug-btn');
        swapBtn = document.getElementById('swap-controls-btn');

        // [BARU] Ambil elemen untuk Log Panel
        logPanel = document.getElementById('log-panel');
        logContent = document.getElementById('log-content');

        // Tambahkan event listener ke tombol-tombol
        if(debugBtn) debugBtn.onclick = () => toggleWindow();
        if(swapBtn) swapBtn.onclick = () => {
            const ui = document.getElementById('mobile-ui-container');
            if(ui) ui.classList.toggle('layout-swapped');
        };

        // Logika untuk membuat debug window bisa digeser
        const header = document.getElementById('debug-header');
        if(header) {
            header.onmousedown = e => {
                isDragging = true;
                offsetX = e.clientX - debugWindow.offsetLeft;
                offsetY = e.clientY - debugWindow.offsetTop;
```

SysX
ENGINE

```
                e.preventDefault();
            };
            document.onmousemove = e => {
                if (isDragging) {
                    debugWindow.style.left = `${e.clientX - offsetX}px`;
                    debugWindow.style.top = `${e.clientY - offsetY}px`;
                }
            };
            document.onmouseup = () => { isDragging = false; };
        }

        // [BARU] Ganti console.log default untuk menangkap semua log
        const oldLog = console.log;
        console.log = function(...args) {
            log(args.join(' '), 'info');
            oldLog.apply(console, args);
        }
        const oldWarn = console.warn;
        console.warn = function(...args) {
            log(args.join(' '), 'warn');
            oldWarn.apply(console, args);
        }
        const oldError = console.error;
        console.error = function(...args) {
            log(args.join(' '), 'error');
            oldError.apply(console, args);
        }
    }

    // Fungsi untuk mengupdate data yang ditampilkan di debug window
    function update(data) {
        if (!isVisible || !debugContent) return;
        let content = '';
        for (const key in data) {
            content += `<div class="debug-item"><span class="debug-
key">${key}:</span><span class="debug-value">${data[key]}</span></div>`;
        }
        debugContent.innerHTML = content;
    }

    // Fungsi untuk menampilkan/menyembunyikan debug window
    function toggleWindow(force) {
        if (!debugWindow) return;
```

SysX ENGINE

```
        isVisible = (force !== undefined) ? force : !isVisible;
        debugWindow.style.display = isVisible ? 'flex' : 'none';
        if(debugBtn) debugBtn.classList.toggle('active', isVisible);
    }

    /**
     * [FUNGSI BARU] Menambahkan pesan ke panel log.
     * @param {string} message - Pesan yang akan ditampilkan.
     * @param {string} type - Tipe log ('info', 'warn', 'error').
     */
    function log(message, type = 'info') {
        if (!logContent) return;
        const logEntry = document.createElement('div');
        logEntry.className = `log-entry log-${type}`;

        const timestamp = new Date().toLocaleTimeString();
        logEntry.innerHTML = `<span class="log-
timestamp">${timestamp}</span><span class="log-message">${message}</span>`;

        logContent.appendChild(logEntry);
        // Auto-scroll ke bawah
        logPanel.scrollTop = logPanel.scrollHeight;
    }

    return { init, update, toggleWindow, log };
})();
```

Sysx_fx.js

```
// --- /sysx_engine/sysx_fx.js ---
window.SysX_Modules = window.SysX_Modules || {};
window.SysX_Modules.fx = (() => {
    let _particles = [];
    let _shakeEndTime = 0,
        _shakeIntensity = 0;

    function collide(r1, r2) {
        if (!r1 || !r2) return false;
        return r1.x < r2.x + r2.w && r1.x + r1.w > r2.x && r1.y < r2.y + r2.h &&
r1.y + r1.h > r2.y;
    }
```

SysX

ENGINE

```
/**
 * [BARU v1.3] Memeriksa tabrakan antara dua objek lingkaran.
 * Objek harus memiliki properti x, y, dan radius.
 * @param {object} c1 - Lingkaran pertama.
 * @param {object} c2 - Lingkaran kedua.
 * @returns {boolean} - True jika bertabrakan.
 */
function collide_circ(c1, c2) {
    if (!c1 || !c2) return false;
    const dx = c1.x - c2.x;
    const dy = c1.y - c2.y;
    const distance = Math.sqrt(dx * dx + dy * dy);
    return distance < c1.radius + c2.radius;
}

/**
 * [BARU v1.3] Memeriksa tabrakan antara objek lingkaran dan persegi panjang.
 * Lingkaran harus memiliki x, y, radius. Persegi harus memiliki x, y, w, h.
 * @param {object} circ - Objek lingkaran.
 * @param {object} rect - Objek persegi panjang.
 * @returns {boolean} - True jika bertabrakan.
 */
function collide_circ_rect(circ, rect) {
    if (!circ || !rect) return false;
    const testX = Math.max(rect.x, Math.min(circ.x, rect.x + rect.w));
    const testY = Math.max(rect.y, Math.min(circ.y, rect.y + rect.h));
    const dx = circ.x - testX;
    const dy = circ.y - testY;
    const distance = Math.sqrt(dx * dx + dy * dy);
    return distance < circ.radius;
}

function shake(intensity, duration) {
    _shakeIntensity = intensity;
    _shakeEndTime = Date.now() + duration;
}

function emit(x, y, count = 10, color = 9, life = 30) {
    for (let i = 0; i < count; i++) _particles.push({
        x,
        y,
        vx: (Math.random() - 0.5) * 4,
```

SysX
ENGINE

```
            vy: (Math.random() - 0.5) * 4 - 1,
            size: Math.random() * 2 + 1,
            life: Math.random() * life,
            color
        });
    }

    function updateParticles() {
        for (let i = _particles.length - 1; i >= 0; i--) {
            const p = _particles[i];
            p.x += p.vx;
            p.y += p.vy;
            p.vy += 0.1;
            p.life--;
            if (p.life <= 0) _particles.splice(i, 1);
        }
    }

    function drawParticles(_ctx, rectfillFunc) {
        for (const p of _particles) rectfillFunc(p.x, p.y, p.size, p.size,
p.color);
    }

    function applyShake(_ctx) {
        if (Date.now() < _shakeEndTime) {
            const dx = (Math.random() - 0.5) * _shakeIntensity * 2;
            const dy = (Math.random() - 0.5) * _shakeIntensity * 2;
            _ctx.translate(dx, dy);
        } else {
            _shakeIntensity = 0;
        }
    }
    const getParticleCount = () => _particles.length;

    // Ekspor semua fungsi, termasuk fungsi collide yang baru
    return {
        collide,
        shake,
        emit,
        updateParticles,
        drawParticles,
        applyShake,
        getParticleCount,
```

SysX
ENGINE

```
        collide_circ,
        collide_circ_rect
    };
})();
```

Sysx_graphics.js

```
// --- /sysx_engine/sysx_graphics.js ---
// Semua fungsi yang berhubungan dengan menggambar.

window.SysX_Modules = window.SysX_Modules || {};

window.SysX_Modules.graphics = (() => {
    const TILE_SIZE = 16;
    const SPRITESHEET_KEY = 'main_spritesheet';

    // Helper untuk mendapatkan konteks dari core
    const getCoreContext = () => SysX_Modules.core.getContext();

    function cls(c) {
        const { _ctx, _canvas, PICO8_PALETTE } = getCoreContext();
        _ctx.fillStyle = PICO8_PALETTE[c] || '#000';
        _ctx.fillRect(0, 0, _canvas.width, _canvas.height);
    }

    function rectfill(x, y, w, h, c) {
        const { _ctx, PICO8_PALETTE } = getCoreContext();
        _ctx.fillStyle = PICO8_PALETTE[c] || '#FFF';
        _ctx.fillRect(x, y, w, h);
    }

    function line(x1, y1, x2, y2, c) {
        const { _ctx, PICO8_PALETTE } = getCoreContext();
        _ctx.strokeStyle = PICO8_PALETTE[c] || '#FFF';
        _ctx.beginPath();
        _ctx.moveTo(x1, y1);
        _ctx.lineTo(x2, y2);
        _ctx.stroke();
    }

    function circ(x, y, r, c) {
```

SysX
ENGINE

```
        const { _ctx, PICO8_PALETTE } = getCoreContext();
        _ctx.strokeStyle = PICO8_PALETTE[c] || '#FFF';
        _ctx.beginPath();
        _ctx.arc(x, y, r, 0, 2 * Math.PI);
        _ctx.stroke();
    }

    function circfill(x, y, r, c) {
        const { _ctx, PICO8_PALETTE } = getCoreContext();
        _ctx.fillStyle = PICO8_PALETTE[c] || '#FFF';
        _ctx.beginPath();
        _ctx.arc(x, y, r, 0, 2 * Math.PI);
        _ctx.fill();
    }

    function print(t, x, y, c) {
        const { _ctx, PICO8_PALETTE } = getCoreContext();
        _ctx.font = '14px "Roboto Mono"';
        _ctx.fillStyle = PICO8_PALETTE[c] || '#FFF';
        _ctx.fillText(t, x, y);
    }

    function spr(idx, dx, dy, flipX = false, flipY = false, angle = 0) {
        const { _ctx, _assets } = getCoreContext();
        const sheet = _assets[SPRITESHEET_KEY];
        if (!sheet || !sheet.complete) return;

        const cols = Math.floor(sheet.width / TILE_SIZE);
        const sx = (idx % cols) * TILE_SIZE;
        const sy = Math.floor(idx / cols) * TILE_SIZE;

        if (!flipX && !flipY && angle === 0) {
            _ctx.drawImage(sheet, sx, sy, TILE_SIZE, TILE_SIZE, dx, dy,
TILE_SIZE, TILE_SIZE);
            return;
        }

        _ctx.save();
        _ctx.translate(dx + TILE_SIZE / 2, dy + TILE_SIZE / 2);
        _ctx.rotate(angle * Math.PI / 180);
        _ctx.scale(flipX ? -1 : 1, flipY ? -1 : 1);
        _ctx.drawImage(sheet, sx, sy, TILE_SIZE, TILE_SIZE, -TILE_SIZE / 2, -
TILE_SIZE / 2, TILE_SIZE, TILE_SIZE);
```

SysX

ENGINE

```
        _ctx.restore();
    }

    function sspr(sx, sy, sw, sh, dx, dy, dw, dh) {
        const { _ctx, _assets } = getCoreContext();
        const sheet = _assets[SPRITESHEET_KEY];
        if (!sheet || !sheet.complete) return;

        const destW = dw === undefined ? sw : dw;
        const destH = dh === undefined ? sh : dh;

        _ctx.drawImage(sheet, sx, sy, sw, sh, dx, dy, destW, destH);
    }

    function map(tm, mx = 0, my = 0) {
        if (!tm) return;
        for (let r = 0; r < tm.length; r++)
            for (let c = 0; c < tm[r].length; c++)
                if (tm[r][c] >= 0) spr(tm[r][c], mx + c * TILE_SIZE, my + r *
TILE_SIZE);
    }

    /**
     * [BARU v2.5] Menggambar sprite dari data string yang digenerate oleh Sprite
Editor.
     * @param {string} dataString - String data sprite (misal: '..7..cc..7..').
     * @param {number} dx - Posisi X tujuan pada canvas.
     * @param {number} dy - Posisi Y tujuan pada canvas.
     * @param {number} [spriteWidth=16] - Lebar sprite.
     * @param {number} [pixelSize=1] - Ukuran setiap piksel.
     */
    function draw_sprite_data(dataString, dx, dy, spriteWidth = 16, pixelSize =
1) {
        if (!dataString) return;
        const { _ctx, PICO8_PALETTE } = getCoreContext();

        for (let i = 0; i < dataString.length; i++) {
            const char = dataString[i];
            if (char === '.') continue; // '.' berarti transparan

            const colorIndex = parseInt(char, 36); // Konversi dari Base36 ke
angka

            const x = i % spriteWidth;
```

SysX ENGINE

```
            const y = Math.floor(i / spriteWidth);

            _ctx.fillStyle = PICO8_PALETTE[colorIndex] || '#FFF';
            _ctx.fillRect(dx + (x * pixelSize), dy + (y * pixelSize), pixelSize,
pixelSize);
        }
    }

    // Ekspor semua fungsi, termasuk fungsi baru 'draw_sprite_data'
    return { cls, rectfill, print, spr, map, line, circ, circfill, sspr,
draw_sprite_data };
})();
```

Sysx_input.js

```
// --- /sysx_engine/sysx_input.js (MODIFIKASI LENGKAP) ---
// Modul untuk mengelola semua input: Keyboard, Mouse, dan Kontroler Mobile.

window.SysX_Modules = window.SysX_Modules || {};
window.SysX_Modules.input = (() => {
    // State internal untuk semua jenis input
    let _inputState = {};
    let _prevInputState = {};
    let _mouseState = { x: 0, y: 0, buttons: {} };
    let _prevMouseState = { buttons: {} };
    // [FIX] Flag baru untuk mencegah panggilan menu berulang pada touch
    let _isTouchPowerDown = false;

    // Peta untuk mengubah input keyboard menjadi nama tombol standar engine
    const KEY_MAP = {
        'ArrowUp': 'up', 'ArrowDown': 'down', 'ArrowLeft': 'left', 'ArrowRight':
'right',
        'w': 'up', 's': 'down', 'a': 'left', 'd': 'right',
        'z': 'btn_b', 'x': 'btn_a',
        'Shift': 'btn_b', ' ': 'btn_a',
        'Enter': 'start', 'Escape': 'power'
    };

    /**
     * Menginisialisasi semua event listener.
     * @param {HTMLCanvasElement} canvas - Elemen canvas game.
```

SysX ENGINE

```
     */
    function init(canvas) {
        // Keyboard listeners
        document.addEventListener('keydown', e => {
            if (e.repeat) return;
            const key = KEY_MAP[e.key];
            if (key) {
                if (key === 'power') {
                    if (SysX.toggle_menu) SysX.toggle_menu();
                } else {
                    _inputState[key] = true;
                }
                e.preventDefault();
            }
        });
        document.addEventListener('keyup', e => {
            const key = KEY_MAP[e.key];
            if (key && key !== 'power') {
                _inputState[key] = false;
                e.preventDefault();
            }
        });

        // Mouse listeners
        canvas.addEventListener('mousemove', e => {
            const rect = canvas.getBoundingClientRect();
            _mouseState.x = Math.round((e.clientX - rect.left) / rect.width *
canvas.width);
            _mouseState.y = Math.round((e.clientY - rect.top) / rect.height *
canvas.height);
        });
        canvas.addEventListener('mousedown', e => { _mouseState.buttons[e.button]
= true; });
        canvas.addEventListener('mouseup', e => { _mouseState.buttons[e.button] =
false; });

        if ('ontouchstart' in window) {
            createMobileUI();
        }
    }

    /**
     * Membuat dan menambahkan UI kontroler mobile ke halaman.
```

SysX

ENGINE

```
     */
    function createMobileUI() {
        const style = document.createElement('style');
        style.textContent = `
            .sysx-mobile-controls { position: fixed; bottom: 0; left: 0; width:
100%; height: 140px; user-select: none; -webkit-user-select: none; z-index: 100;
}
            .sysx-dpad, .sysx-buttons { position: absolute; bottom: 20px;
display: grid; opacity: 0.6; }
            .sysx-dpad { left: 20px; grid-template-columns: 44px 44px 44px; grid-
template-rows: 44px 44px 44px; gap: 5px; }
            .sysx-buttons { right: 20px; grid-template-columns: 55px 55px; gap:
25px; align-items: center; }
            .sysx-btn { background: rgba(100, 100, 100, 0.7); border: 2px solid
#fff; text-align: center; color: #fff; font-family: monospace; font-weight: bold;
font-size: 24px; }
            .sysx-dpad .sysx-btn { border-radius: 8px; line-height: 40px; }
            .sysx-buttons .sysx-btn { border-radius: 50%; height: 55px; line-
height: 51px; }
            .sysx-dpad-up { grid-column: 2; grid-row: 1; }
            .sysx-dpad-left { grid-column: 1; grid-row: 2; }
            .sysx-dpad-right { grid-column: 3; grid-row: 2; }
            .sysx-dpad-down { grid-column: 2; grid-row: 3; }
            .sysx-power-btn { position: absolute; top: 10px; right: 10px; width:
40px; height: 30px; border-radius: 5px; line-height: 28px; font-size: 14px; }
        `;
        document.head.appendChild(style);

        const container = document.createElement('div');
        container.className = 'sysx-mobile-controls';

        const dpadHTML = `<div class="sysx-dpad"><div class="sysx-btn sysx-dpad-
up" data-key="up">▲</div><div class="sysx-btn sysx-dpad-left" data-
key="left">◀</div><div class="sysx-btn sysx-dpad-right" data-
key="right">▶</div><div class="sysx-btn sysx-dpad-down" data-
key="down">▼</div></div>`;
        const buttonsHTML = `<div class="sysx-buttons"><div class="sysx-btn"
data-key="btn_b">B</div><div class="sysx-btn" data-key="btn_a">A</div></div>`;
        const powerHTML = `<div class="sysx-btn sysx-power-btn" data-
key="power">MENU</div>`;

        container.innerHTML = dpadHTML + buttonsHTML + powerHTML;
        document.body.appendChild(container);
```

SysX

ENGINE

```
        // [FIX] Menggunakan satu handler yang lebih pintar untuk semua event
touch
        container.addEventListener('touchstart', handleTouch, { passive: false
});
        container.addEventListener('touchend', handleTouch, { passive: false });
        container.addEventListener('touchmove', handleTouch, { passive: false });
    }

    // [FIX] Handler tunggal yang diperbarui untuk semua event touch
    function handleTouch(e) {
        e.preventDefault();

        const VIRTUAL_KEYS = ['up', 'down', 'left', 'right', 'btn_a', 'btn_b'];
        const currentlyPressed = new Set();

        // 1. Deteksi semua tombol yang sedang disentuh
        for (let i = 0; i < e.touches.length; i++) {
            const touch = e.touches[i];
            const elem = document.elementFromPoint(touch.clientX, touch.clientY);
            if (elem && elem.dataset.key) {
                currentlyPressed.add(elem.dataset.key);
            }
        }

        // 2. Perbarui state untuk tombol D-Pad dan A/B
        VIRTUAL_KEYS.forEach(k => {
            _inputState[k] = currentlyPressed.has(k);
        });

        // 3. Logika khusus untuk tombol 'power' agar hanya terpanggil sekali
        const isPowerCurrentlyDown = currentlyPressed.has('power');
        // Jika tombol 'power' baru saja ditekan (sebelumnya tidak)
        if (isPowerCurrentlyDown && !_isTouchPowerDown) {
            if (SysX.toggle_menu) SysX.toggle_menu();
        }
        // Perbarui status tombol 'power' untuk event berikutnya
        _isTouchPowerDown = isPowerCurrentlyDown;
    }


    // --- API Publik ---
    function btn(k) { return _inputState[k] || false; }
```

SysX

ENGINE

```
    function btnp(k) { return _inputState[k] && !_prevInputState[k]; }

    function mouse_btn(b) { return _mouseState.buttons[b] || false; }
    function mouse_btnp(b) { return _mouseState.buttons[b] &&
!_prevMouseState.buttons[b]; }

    function _updatePrevInput() {
        _prevInputState = { ..._inputState };
        _prevMouseState.buttons = { ..._mouseState.buttons };
    }

    return {
        init,
        btn, btnp,
        mouse_x: () => _mouseState.x,
        mouse_y: () => _mouseState.y,
        mouse_btn, mouse_btnp,
        _updatePrevInput
    };
})();
```

Sysx_main.js

```
// --- /sysx_engine/sysx_main.js ---
// File ini bertugas sebagai "lem", menyatukan semua modul
// menjadi satu objek global `SysX` yang akan digunakan oleh developer.

const SysX = {};

// --- TAHAP 1: Integrasi Modul-Modul Dasar ---
Object.assign(
    SysX,
    SysX_Modules.core,
    SysX_Modules.graphics,
    SysX_Modules.audio,
    SysX_Modules.fx
);

// --- TAHAP 2: Integrasi Modul Input (Secara Spesifik) ---
if (SysX_Modules.input) {
    SysX.btn = SysX_Modules.input.btn;
```

SysX

ENGINE

```
    SysX.btnp = SysX_Modules.input.btnp;
    SysX.mouse = {
        get x() { return SysX_Modules.input.mouse_x(); },
        get y() { return SysX_Modules.input.mouse_y(); },
        btn: SysX_Modules.input.mouse_btn,
        btnp: SysX_Modules.input.mouse_btnp,
    };
    SysX._init_input = SysX_Modules.input.init;
}

// --- TAHAP 3: Integrasi Modul dengan Pola Facade ---
if (SysX_Modules.camera) {
    SysX.camera = SysX_Modules.camera.set;
}
if (SysX_Modules.tween) {
    SysX.tween = (target, properties, duration, ease, onComplete) => {
        const { _totalFrames } = SysX_Modules.core.getContext();
        SysX_Modules.tween.create(target, properties, duration, ease, onComplete,
_totalFrames);
    };
}
if (SysX_Modules.timer) {
    SysX.timer_set = (name, duration, callback, isLooping) => {
        const { _totalFrames } = SysX_Modules.core.getContext();
        SysX_Modules.timer.set(name, duration, callback, isLooping,
_totalFrames);
    };
    SysX.timer_is_running = SysX_Modules.timer.is_running;
    SysX.timer_cancel = SysX_Modules.timer.cancel;
}
if (SysX_Modules.transition) {
    SysX.transition = SysX_Modules.transition.start;
}

// [BARU v2.0] Integrasi Modul Konten & Konsol
if (SysX_Modules.content) {
    SysX.content_load_zip = SysX_Modules.content.load_zip;
    SysX.content_load_url = SysX_Modules.content.load_url;
}
if (SysX_Modules.devconsole) {
    SysX.console_toggle = SysX_Modules.devconsole.toggle;
    SysX.console_register = SysX_Modules.devconsole.register;
    SysX.console_log = SysX_Modules.devconsole.log;
```

SysX

ENGINE

```
}

// [BARU v2.5 "Creator's Update"] Integrasi Modul Synth
if (SysX_Modules.synth) {
    SysX.play_synth = SysX_Modules.synth.play;
}

// ============================================================================
// [BARU v3.0 "Professional"] Integrasi Modul Menu Sistem
// ============================================================================
if (SysX_Modules.menu) {
    SysX.toggle_menu = SysX_Modules.menu.toggle;
}
```

Sysx_timer.js

```
// --- /sysx_engine/sysx_timer.js ---
// Modul untuk sistem timer/alarm.

window.SysX_Modules = window.SysX_Modules || {};
window.SysX_Modules.timer = (() => {
    let _timers = new Map();

    /**
     * Mengatur timer baru.
     * @param {string} name - Nama unik timer.
     * @param {number} duration - Durasi dalam frame.
     * @param {function} callback - Fungsi yang dipanggil.
     * @param {boolean} isLooping - Apakah timer berulang.
     * @param {number} totalFrames - Frame saat ini dari core.
     */
    const set = (name, duration, callback, isLooping = false, totalFrames) => {
        if (!name || !callback) return;
        _timers.set(name, {
            duration,
            endFrame: totalFrames + duration,
            callback,
            isLooping
        });
    };
```

SysX
ENGINE

```
    const is_running = (name) => _timers.has(name);
    const cancel = (name) => _timers.delete(name);

    /**
     * Mengupdate semua timer aktif.
     * Dipanggil oleh sysx_core.js setiap frame.
     * @param {number} totalFrames - Frame saat ini dari core.
     */
    const update = (totalFrames) => {
        for (const [name, timer] of _timers.entries()) {
            if (totalFrames >= timer.endFrame) {
                timer.callback.call(window.Game);
                if (timer.isLooping) {
                    timer.endFrame = totalFrames + timer.duration;
                } else {
                    _timers.delete(name);
                }
            }
        }
    };

    return { set, is_running, cancel, update };
})();
```

Sysx_menu.js

```
// --- /sysx_engine/sysx_menu.js (MODERN STYLE V3.1) ---
// Modul untuk mengelola menu sistem default yang profesional.

window.SysX_Modules = window.SysX_Modules || {};

window.SysX_Modules.menu = (() => {
    // State internal modul menu
    let _isOpen = false;
    let _currentView = 'main'; // 'main', 'options', 'controls'
    let _selectedIndex = 0;

    // State untuk audio
    let _masterVolume = 1.0;
    let _soundEnabled = true;
```

SysX
ENGINE

```
    // State untuk animasi
    let _selectionAnimOffset = 0;
    let _openCloseAnim = 0; // 0 = closed, 1 = open

    const _menuViews = {
        main: {
            title: "- GAME PAUSED -",
            options: [
                { label: "CONTINUE", action: () => toggle() },
                { label: "RESET CART", action: () => {
                    toggle();
                    if (window.Game && typeof window.Game._create === 'function')
{
                        window.Game._create();
                    }
                }},
                { label: "OPTIONS", action: () => { _currentView = 'options';
_selectedIndex = 0; playSfx('select'); } },
                // [BARU] Menambahkan opsi CHEAT
                { label: "CHEAT", action: () => {
                    toggle(); // Tutup menu pause
                    // Buka developer console jika ada
                    if (SysX.console_toggle) {
                        SysX.console_toggle();
                    }
                }},
            ]
        },
        options: {
            title: "- OPTIONS -",
            options: [
                { label: "SOUND", type: 'toggle' },
                { label: "VOLUME", type: 'slider' },
                { label: "CONTROLS", action: () => { _currentView = 'controls';
_selectedIndex = 0; playSfx('select'); } },
                { label: "BACK", action: () => { _currentView = 'main';
_selectedIndex = 2; playSfx('select'); } }, // Kembali dan sorot OPTIONS
            ]
        },
        controls: {
            title: "- CONTROLS -",
        }
    };
```

SysX

ENGINE

```
    const _sfx = {
        openClose: { wave: "square", vol: 0.3, freq: 440, slide: 200, attack:
0.01, sustain: 0.08, decay: 0.1 },
        navigate: { wave: "triangle", vol: 0.2, freq: 880, slide: 0, attack:
0.01, sustain: 0.01, decay: 0.05 },
        select: { wave: "sine", vol: 0.3, freq: 660, slide: 0, attack: 0.01,
sustain: 0.1, decay: 0.15 }
    };

    function playSfx(sfxName) {
        if (!_soundEnabled) return;
        if (SysX.play_synth && _sfx[sfxName]) {
            const sfxData = { ..._sfx[sfxName] };
            sfxData.vol *= _masterVolume;
            SysX.play_synth(sfxData);
        }
    }

    function toggle() {
        _isOpen = !_isOpen;
        if (!_isOpen) {
             _currentView = 'main';
        }
        _selectedIndex = 0;
        playSfx('openClose');
    }

    function update() {
        if (!_isOpen && _openCloseAnim <= 0) return;

        if (_isOpen && _openCloseAnim < 1) _openCloseAnim += 0.15;
        else if (!_isOpen && _openCloseAnim > 0) _openCloseAnim -= 0.15;
        _openCloseAnim = Math.max(0, Math.min(1, _openCloseAnim));

        if (_selectionAnimOffset > 0) _selectionAnimOffset -= 0.5;

        if (!_isOpen) return;

        const view = _menuViews[_currentView];

        if (SysX.btnp('btn_b') || (SysX.btnp('power') && _currentView !==
'main')) {
```

SysX ENGINE

```
            if (_currentView === 'options') {
                _currentView = 'main';
                _selectedIndex = 2; // Sorot 'OPTIONS' lagi
                playSfx('openClose');
                return;
            } else if (_currentView === 'controls') {
                _currentView = 'options';
                _selectedIndex = 2; // Sorot 'CONTROLS' lagi
                playSfx('openClose');
                return;
            }
        }

        if (_currentView === 'controls') return;

        if (SysX.btnp('down')) {
            _selectedIndex = (_selectedIndex + 1) % view.options.length;
            _selectionAnimOffset = 3;
            playSfx('navigate');
        }
        if (SysX.btnp('up')) {
            _selectedIndex = (_selectedIndex - 1 + view.options.length) %
view.options.length;
            _selectionAnimOffset = 3;
            playSfx('navigate');
        }

        const selectedOption = view.options[_selectedIndex];

        if (selectedOption.type === 'slider') {
            if (SysX.btn('right')) _masterVolume = Math.min(1.0, _masterVolume +
0.02);
            if (SysX.btn('left')) _masterVolume = Math.max(0.0, _masterVolume -
0.02);
        }

        if (SysX.btnp('btn_a') || SysX.btnp('start')) {
            if (selectedOption.action) {
                selectedOption.action();
            } else if (selectedOption.type === 'toggle') {
                _soundEnabled = !_soundEnabled;
                playSfx('select');
            }
```

SysX ENGINE

```
        }
    }

    function draw() {
        if (_openCloseAnim <= 0) return;

        const { _ctx, _canvas } = SysX_Modules.core.getContext();
        const { rectfill, print } = SysX_Modules.graphics;
        const view = _menuViews[_currentView];

        const grad = _ctx.createLinearGradient(0, 0, 0, _canvas.height);
        grad.addColorStop(0, `rgba(29, 43, 83, ${0.9 * _openCloseAnim})`);
        grad.addColorStop(1, `rgba(0, 0, 0, ${0.95 * _openCloseAnim})`);
        _ctx.fillStyle = grad;
        _ctx.fillRect(0, 0, _canvas.width, _canvas.height);

        const menuWidth = 180;
        const menuHeight = 120;
        const animY = -menuHeight + (menuHeight * _openCloseAnim);
        const menuX = (_canvas.width - menuWidth) / 2;
        const menuY = (_canvas.height - menuHeight) / 2;

        rectfill(menuX + 4, menuY + 4 + animY, menuWidth, menuHeight, 0);
        rectfill(menuX, menuY + animY, menuWidth, menuHeight, 1);
        rectfill(menuX + 2, menuY + 2 + animY, menuWidth - 4, menuHeight - 4, 0);

        _ctx.globalAlpha = _openCloseAnim;
        print(view.title, menuX + 10, menuY + 15 + animY, 7);

        if (_currentView === 'controls') {
            drawControlsView(print, menuX, menuY + animY);
        } else {
            drawOptionsView(print, rectfill, view.options, menuX, menuY + animY);
        }
        _ctx.globalAlpha = 1.0;
    }

    function drawOptionsView(print, rectfill, options, menuX, menuY) {
        options.forEach((option, index) => {
            const y = menuY + 40 + (index * 18); // [UBAH] Kurangi jarak antar
item
            let label = option.label;
            let animX = 0;
```

SysX
ENGINE

```
            if (index === _selectedIndex) {
                animX = Math.sin(_selectionAnimOffset) * 2;
                print("▶", menuX + 15 + animX, y, 10);
            }

            if (option.type === 'toggle') {
                label += _soundEnabled ? ": ON" : ": OFF";
            }

            print(label, menuX + 30 + animX, y, 7);

            if (option.type === 'slider') {
                const barStartX = menuX + 95;
                const numBlocks = 8;
                const blockWidth = 6;
                const blockGap = 2;
                const activeBlocks = Math.round(_masterVolume * numBlocks);
                for (let i = 0; i < numBlocks; i++) {
                    const color = i < activeBlocks ? 11 : 5;
                    rectfill(barStartX + i * (blockWidth + blockGap), y - 4,
blockWidth, 10, color);
                }
            }
        });
    }

    function drawControlsView(print, menuX, menuY) {
        const buttons = ['up', 'down', 'left', 'right', 'btn_a', 'btn_b',
'start'];
        print("PRESS ANY BUTTON", menuX + 25, menuY + 40, 6);

        buttons.forEach((btnKey, index) => {
            const col = index > 3 ? 1 : 0;
            const row = index % 4;
            const x = menuX + 30 + (col * 70);
            const y = menuY + 60 + (row * 12);

            const color = SysX.btn(btnKey) ? 11 : 5;
            print(btnKey.toUpperCase(), x, y, color);
        });
        print("PRESS (B) TO GO BACK", menuX + 15, menuY + 105, 6);
    }
```

SysX

ENGINE

```
    return {
        toggle,
        update,
        draw,
        isOpen: () => _isOpen
    };

})();
```

Sysx_synth.js

```
// --- /sysx_engine/sysx_synth.js (BARU) ---
// Modul untuk memainkan efek suara secara prosedural menggunakan Web Audio API.
// Ini adalah jantung dari fitur SFX internal.

window.SysX_Modules = window.SysX_Modules || {};

window.SysX_Modules.synth = (() => {

    // AudioContext dibuat sekali dan digunakan kembali untuk efisiensi.
    let audioCtx = null;

    // Fungsi untuk menginisialisasi AudioContext.
    // Harus dipicu oleh interaksi pengguna (misal: klik) untuk kebijakan browser modern.
    function initAudioContext() {
        if (!audioCtx) {
            try {
                audioCtx = new (window.AudioContext || window.webkitAudioContext)();
            } catch (e) {
                console.error("SysX Synth Error: Web Audio API is not supported.");
            }
        }
    }

    /**
     * Memainkan sebuah efek suara berdasarkan objek konfigurasi.
     * @param {object} sfxObject Objek yang berisi parameter suara.
```

SysX ENGINE

```
     * Contoh: { wave: 'square', vol: 0.5, freq: 440, slide: -200, attack: 0.01,
sustain: 0.1, decay: 0.2 }
     */
    function play(sfxObject) {
        // Jika AudioContext belum siap, coba inisialisasi.
        // Jika gagal (misal: belum ada interaksi pengguna), suara tidak akan
dimainkan.
        initAudioContext();
        if (!audioCtx) return;

        // Jika context dalam keadaan 'suspended', coba lanjutkan.
        if (audioCtx.state === 'suspended') {
            audioCtx.resume();
        }

        // Ambil parameter dengan nilai default untuk keamanan
        const p = {
            wave: sfxObject.wave || 'sine',
            vol: sfxObject.vol ?? 0.5,
            freq: sfxObject.freq || 440,
            slide: sfxObject.slide || 0,
            attack: sfxObject.attack || 0.01,
            sustain: sfxObject.sustain || 0.1,
            decay: sfxObject.decay || 0.2,
        };

        const now = audioCtx.currentTime;
        const totalDuration = p.attack + p.sustain + p.decay;

        // 1. Buat GainNode untuk mengontrol volume (envelope)
        const gainNode = audioCtx.createGain();
        gainNode.connect(audioCtx.destination);

        // Atur envelope ADSR (Attack, Decay, Sustain, Release)
        // Di sini kita sederhanakan menjadi Attack -> Sustain -> Decay
        gainNode.gain.setValueAtTime(0, now); // Mulai dari volume 0
        gainNode.gain.linearRampToValueAtTime(p.vol, now + p.attack); // Naik ke
volume puncak (Attack)
        // Setelah attack dan sustain, volume akan turun ke 0 (Decay)
        gainNode.gain.linearRampToValueAtTime(0, now + totalDuration);

        // 2. Buat sumber suara (Oscillator atau Noise)
        if (p.wave === 'noise') {
```

SysX ENGINE

```
            const bufferSize = audioCtx.sampleRate * totalDuration;
            const buffer = audioCtx.createBuffer(1, bufferSize,
audioCtx.sampleRate);
            const data = buffer.getChannelData(0);
            for (let i = 0; i < bufferSize; i++) {
                data[i] = Math.random() * 2 - 1; // Isi buffer dengan white noise
            }
            const noiseSource = audioCtx.createBufferSource();
            noiseSource.buffer = buffer;
            noiseSource.connect(gainNode);
            noiseSource.start(now);
            // Tidak perlu stop, karena durasi buffer sudah menentukannya.
        } else {
            const oscillator = audioCtx.createOscillator();
            oscillator.type = p.wave;
            oscillator.connect(gainNode);

            // Atur frekuensi (pitch) dan perubahannya (slide)
            const endFreq = p.freq + p.slide;
            oscillator.frequency.setValueAtTime(p.freq, now); // Frekuensi awal
            if (p.slide !== 0) {
                // Frekuensi tidak boleh 0 atau negatif
                oscillator.frequency.linearRampToValueAtTime(endFreq > 0 ?
endFreq : 1, now + totalDuration);
            }

            oscillator.start(now);
            oscillator.stop(now + totalDuration);
        }
    }

    // Ekspor API publik dari modul synth
    return {
        play
    };

})();
```

SysX
ENGINE

Sysx_transition.js

```
// --- /sysx_engine/sysx_transition.js (BARU) ---
// Modul untuk mengelola transisi layar antar state.

window.SysX_Modules = window.SysX_Modules || {};

window.SysX_Modules.transition = (() => {
    // State internal modul transisi
    let _active = false;
    let _progress = 0;
    let _duration = 0;
    let _type = 'fade'; // Tipe transisi, misal: 'fade', 'wipe', 'slide'
    let _onComplete = null; // Callback setelah transisi selesai

    /**
     * Memulai sebuah transisi baru.
     * @param {string} type - Tipe transisi ('fade').
     * @param {number} duration - Durasi dalam frame.
     * @param {function} onCompleteCallback - Fungsi yang dipanggil setelah
transisi selesai.
     */
    function start(type, duration, onCompleteCallback) {
        // Jangan mulai transisi baru jika sudah ada yang aktif
        if (_active) return;

        _type = type || 'fade';
        _duration = duration || 30; // Default 30 frame
        _onComplete = onCompleteCallback || null;
        _progress = 0;
        _active = true;
    }

    /**
     * Mengupdate logika transisi setiap frame.
     * Dipanggil oleh game loop di sysx_core.js.
     */
    function update() {
        if (!_active) return;

        _progress++;
        if (_progress >= _duration) {
            _active = false;
```

SysX ENGINE

```
            _progress = 0;
            // Panggil callback jika ada, lalu hapus
            if (_onComplete) {
                _onComplete();
                _onComplete = null;
            }
        }
    }

    /**
     * Menggambar efek transisi ke layar.
     * Dipanggil oleh game loop di sysx_core.js.
     * @param {CanvasRenderingContext2D} _ctx - Konteks canvas.
     * @param {HTMLCanvasElement} _canvas - Elemen canvas.
     */
    function draw(_ctx, _canvas) {
        if (!_active) return;

        // Normalisasi progres dari 0 ke 1
        const t = _progress / _duration;

        switch (_type) {
            case 'fade':
                // Gambar persegi hitam dengan transparansi sesuai progres
                _ctx.fillStyle = `rgba(0, 0, 0, ${t})`;
                _ctx.fillRect(0, 0, _canvas.width, _canvas.height);
                break;
            // Tipe transisi lain bisa ditambahkan di sini (wipe, slide, dll.)
        }
    }

    /**
     * Memeriksa apakah ada transisi yang sedang berjalan.
     * @returns {boolean}
     */
    function is_active() {
        return _active;
    }

    // Ekspor API publik dari modul
    return { start, update, draw, is_active };
})();
```

SysX ENGINE

Sysx_tween.js

```
// --- /sysx_engine/sysx_tween.js ---
// Modul untuk mesin animasi (tweening).

window.SysX_Modules = window.SysX_Modules || {};
window.SysX_Modules.tween = (() => {
    let _tweens = [];
    const Easing = {
        linear: t => t,
        easeIn: t => t * t,
        easeOut: t => t * (2 - t),
        easeInOut: t => t < 0.5 ? 2 * t * t : -1 + (4 - 2 * t) * t
    };

    /**
     * Membuat tween baru.
     * @param {object} target - Objek yang dianimasikan.
     * @param {object} properties - Nilai akhir yang dituju.
     * @param {number} duration - Durasi dalam frame.
     * @param {string} ease - Tipe easing.
     * @param {function} onComplete - Callback setelah selesai.
     * @param {number} totalFrames - Frame saat ini dari core.
     */
    const create = (target, properties, duration, ease = 'linear', onComplete =
null, totalFrames) => {
        if (!target) return;
        const startValues = {};
        for (const prop in properties) {
            if (target[prop] !== undefined) {
                startValues[prop] = target[prop];
            }
        }
        _tweens.push({
            target,
            startValues,
            endValues: properties,
            duration,
            ease: Easing[ease] ? ease : 'linear',
            startFrame: totalFrames,
```

SysX
ENGINE

```
            onComplete
        });
    };

    /**
     * Mengupdate semua tween aktif.
     * Dipanggil oleh sysx_core.js setiap frame.
     * @param {number} totalFrames - Frame saat ini dari core.
     */
    const update = (totalFrames) => {
        for (let i = _tweens.length - 1; i >= 0; i--) {
            const t = _tweens[i];
            const elapsed = totalFrames - t.startFrame;
            const progress = Math.min(elapsed / t.duration, 1);
            const easedProgress = Easing[t.ease](progress);

            for (const prop in t.endValues) {
                const startVal = t.startValues[prop];
                const endVal = t.endValues[prop];
                t.target[prop] = startVal + (endVal - startVal) * easedProgress;
            }

            if (progress >= 1) {
                if(t.onComplete) t.onComplete.call(window.Game);
                _tweens.splice(i, 1);
            }
        }
    };

    return { create, update };
})();
```

SysX
ENGINE

LAMPIRAN (UPDATE TERBARU)

Source Code **Contoh *Template* Penerapan Game.**

Template game.js (Mulai Dari Sini)

Template manifest.js

SysX
ENGINE