# 07 Pengayaan #2

## Gambaran Umum

Tidak seperti materi – materi sebelumnya yang menggunakan tampilan *terminal* (atau *Command Line*, CLI), pengayaan ini berisi beberapa *project* yang menggunakan tampilan grafik. Untuk dapat (mempermudah) menggunakan tampilan grafik harus menyertakan *library* eksternal.

Ada banyak *library* grafik yang biasanya digunakan dalam pembuatan video game. Yang paling terkenal adalah OpenGL, Direct3D, SDL, SFML, dan lain – lain. Tiap *library* memiliki karakteristik tertentu berkaitan dengan cara pengaturan, kelengkapan, dan lain – lain.

Materi pengayaan ini menggunakan *library* kecil bernama **SIGIL**. *Library* ini berukuran kecil dan sangat mudah untuk dipelajari. Penjelasan lebih lanjut dapat dilihat pada *slide* **07 Pengayaan 2.pptx.**

## Tujuan

Pengayaan ke-2 ini bertujuan untuk memahami penggunaan *library* dalam sebuah *project* sesuai kebutuhan. Materi ini berfokus pada:

1. Cara mendapatkan *external library.*
2. Pengaturan *library* dalam sebuah *project.*
3. Penggunaan beberapa contoh fungsi dari *library* yang sudah dimasukkan dalam *project*.

## Percobaan

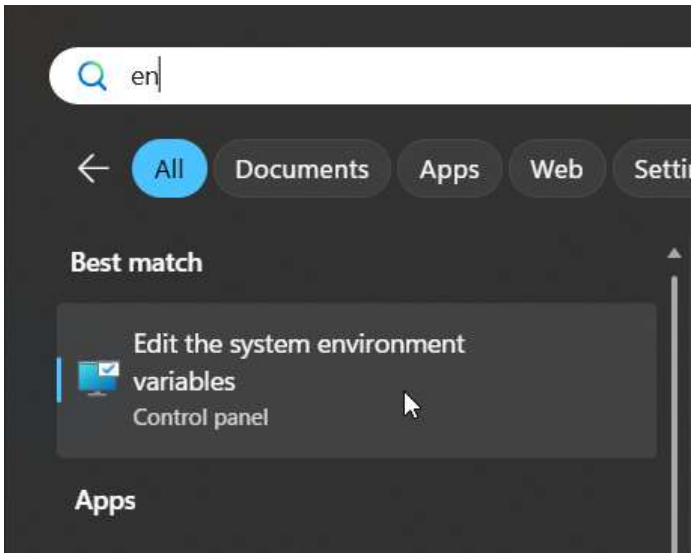
Langkah pertama sebelum membuat beberapa *project* adalah mengunduh dan mengatur konfigurasi *library* SIGIL. *File* dapat diunduh melalui *link* https://reps.yipsip.my.id/libs/sigil.zip. Kemudian ekstraklah di tempat yang dipilih. Dalam contoh ini *library* SIGIL berada di D:\libs\sigil.

Dalam folder sigil terdapat:

1. Folder **include** yang berisi *file* **sl.h.** *File* ini adalah *header* yang nantinya akan dimasukkan (**#include <sl.h>**) dalam kode program.
2. Folder **lib** berisi file **sigil.lib** yang merupakan kode bahasa mesin yang nantinya akan digabung dengan program yang dibuat.
3. *File – file* dengan ekstensi ***.dll** yang digunakan sebagai pendukung agar program yang dibuat menggunakan *library* SIGIL dapat berjalan.
4. File **SIGIL_API.pdf** sebagai referensi (manual) penggunaan *library*.

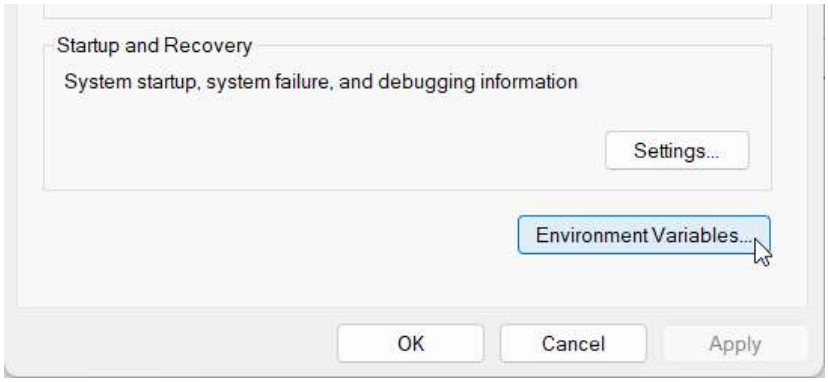
Langkah berikutnya adalah melakukan pengaturan **PATH** atau lokasi di mana *library* SIGIL berada. Pengaturan ini diperlukan agar sistem dapat mengenali ketika ada program yang menggunakan *library* tersebut sedang berjalan. Caranya adalah sebagai berikut:

1. Dari **Start** Windows, cari menu **Edit the system environment variables.** Untuk mempercepat dapat memanfaatkan fitur pencarian dengan mengetikkan '**env**'.



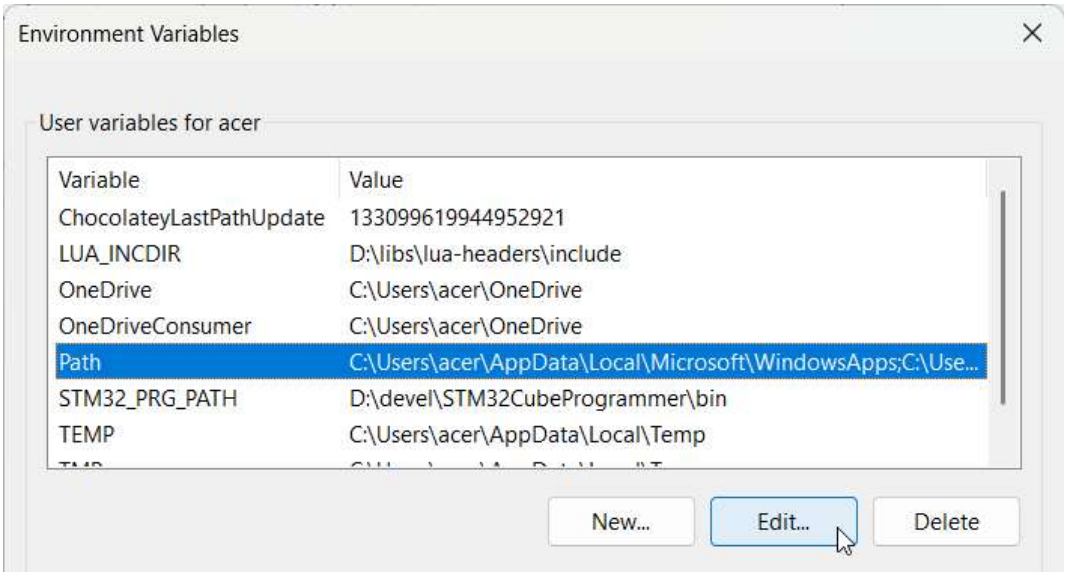

*Gambar 12 Menu Environment Variables*

2. Akan muncul jendela seperti berikut ini, kemudian klik tombol **Environment Variables**.



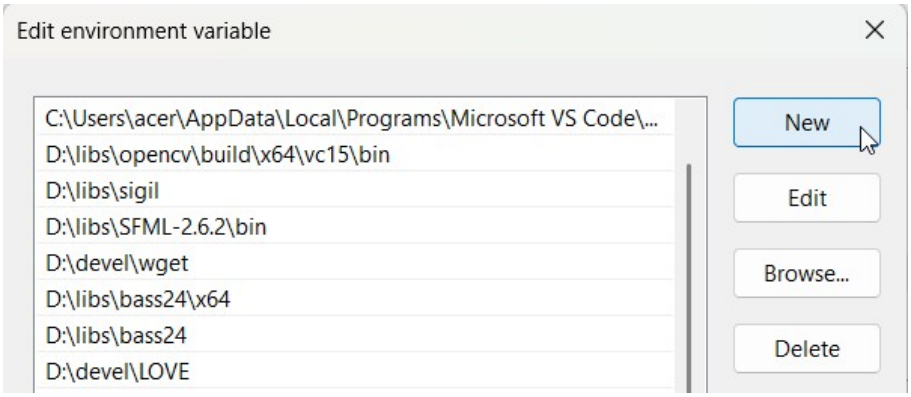

*Gambar 13 Tombol Environment Variables*

3. Akan muncul jendela **Environment Variables**, pilih pada baris **PATH** dan klik tombol **Edit.**




*Gambar 14 Edit Path*

4. Akan muncul lagi jendela **Edit Environment Variables**, klik tombol **New**.Akan muncul baris baru dan tuliskan lokasi folder *library* SIGIL berada. Di contoh ini berada di D:\libs\sigil.




*Gambar 15 Menambah Baris Path*

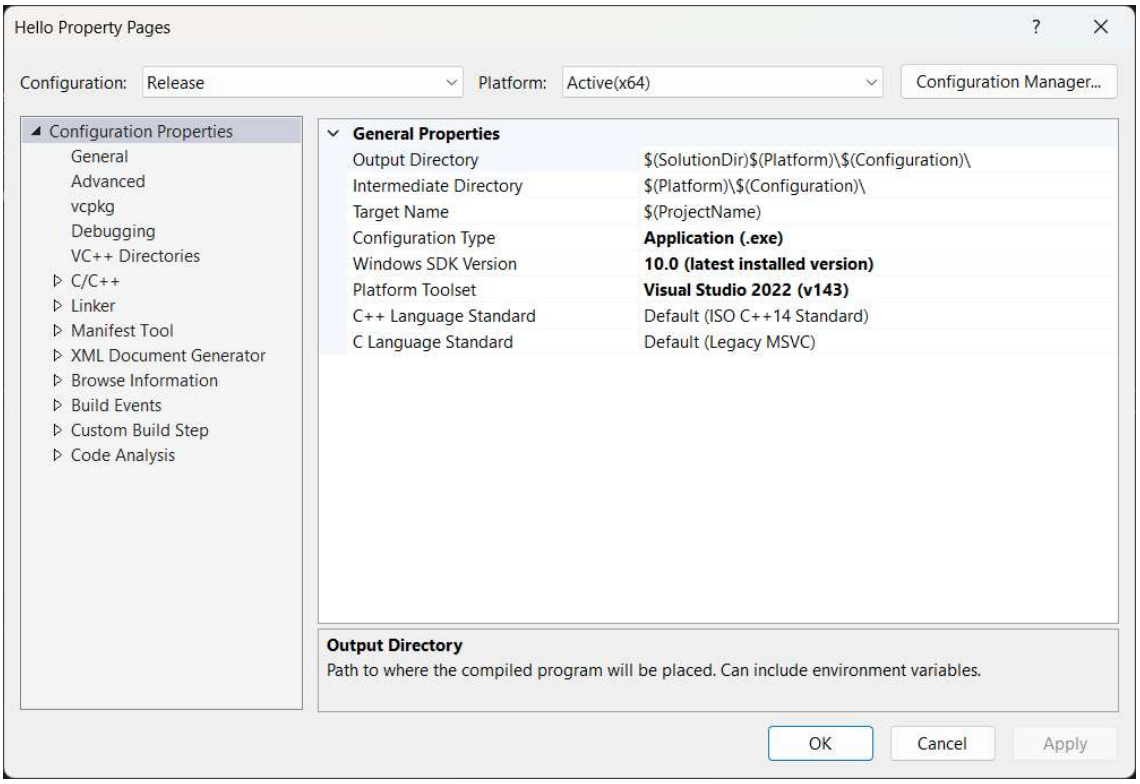
5. Setelah mengisi lokasi folder, klik tombol **OK** di jendela bagian bawah.

Setelah pengaturan *library* selesai, selanjutnya dapat mulai untuk membuat beberapa *project* untuk percobaan.

## Hello

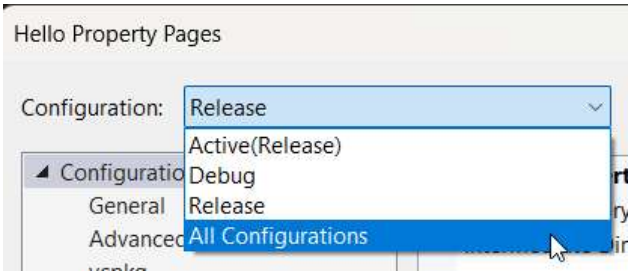
*Project* **Hello** menampilkan sebuah lingkaran di tengah layar. Sebelum menulis kode ada bebrapa hal yang harus di-*setting*. Buatlah sebuah *project* bernama **Hello**, tambahkan file **main.cpp** seperti biasa kemudian lakukan beberapa langkah sebagai berikut:

1. Pilih menu **Project → Hello properties** sehingga muncul tampilan



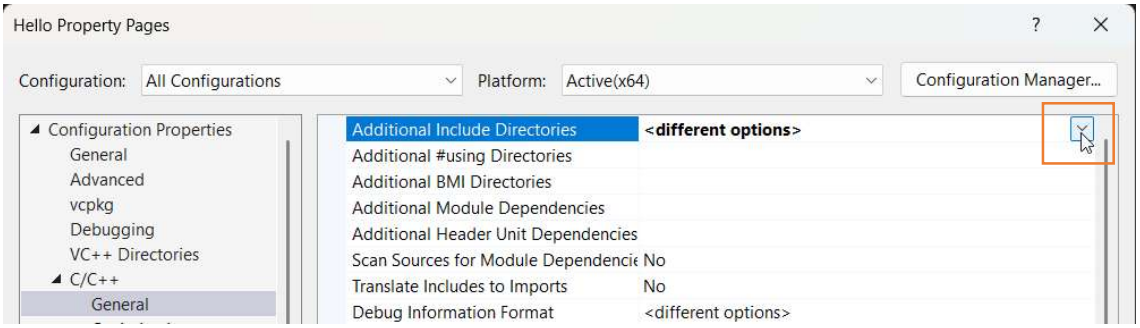

*Gambar 16 Project properties*

2. Di jendela Property pilih **Configuration** ke **All Configuration** seperti di tunjukkan pada Gambar 17 berikut ini:



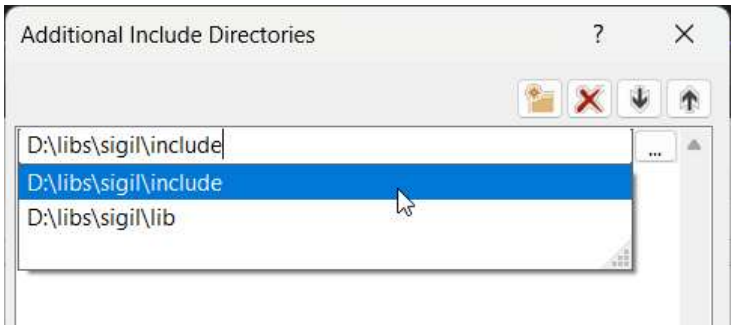

*Gambar 17 Configuration*

3. Pilih tree **C++ → General → Additional Include Directories** kemudian klik tanda panah seperti pada Gambar 19 berikut ini:



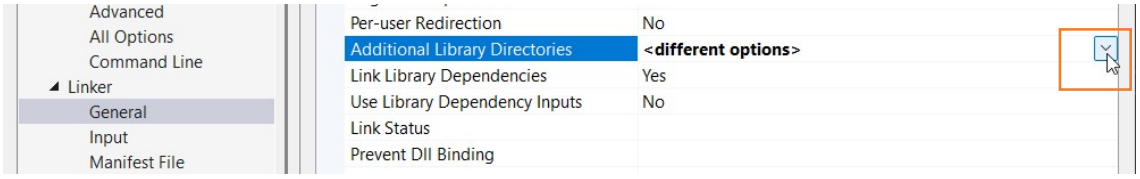

*Gambar 18 Setting include directories*

4. Akan muncul jendela seperti Gambar 19 berikut ini, kemudian klik 2x di baris pertama dan ketikkan lokasi folder **include** berada. Di contoh ini lokasinya adalah di **D:\libs\sigil\include**. Sesuaikan dengan lokasi masing – masing. Setelah itu klik tombol **OK.**



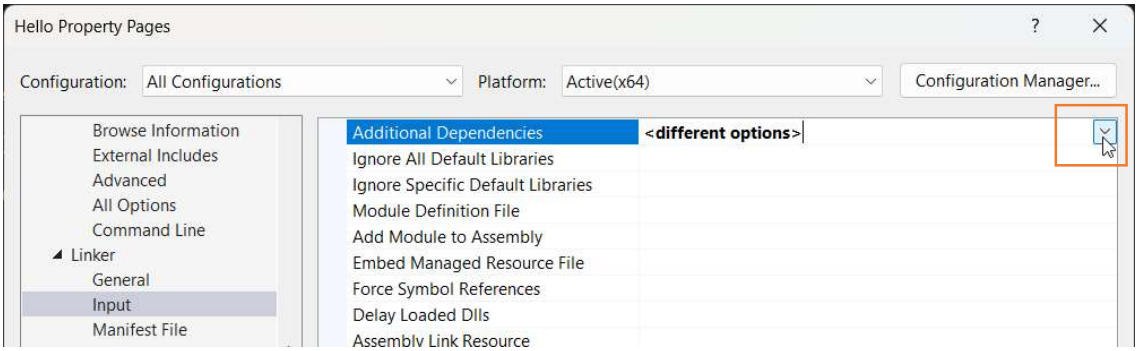

*Gambar 19 Path include directories*

5. Beralih ke tree **Linker → General → Additional Library Directories** kemudian klik tanda panah seperti pada Gambar 20 berikut ini.



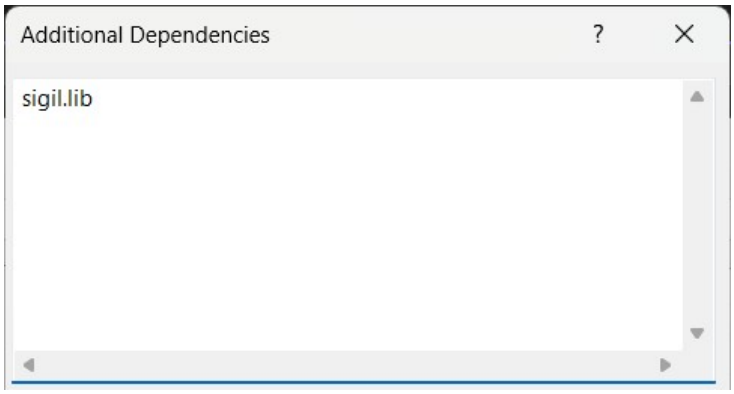

*Gambar 20 Setting library directories*

6. Masukkan lokasi folder **lib** melalui jendela seperti pada **langkah 4**. Pada contoh ini lokasinya berada di D:\libs\sigil\lib. Sesuaikan dengan lokasi masing – masing. Kemudian klik **OK**.
7. Beralih ke tree **Linker → Input → Additional Dependencies** kemudian klik tanda panah seperti Gambar 21 berikut ini.




*Gambar 21 Setting dependencies*

8. Akan muncul jendela untuk mengisikan nama *library*. Isikan **sigil.lib** kemidian klik **OK**.



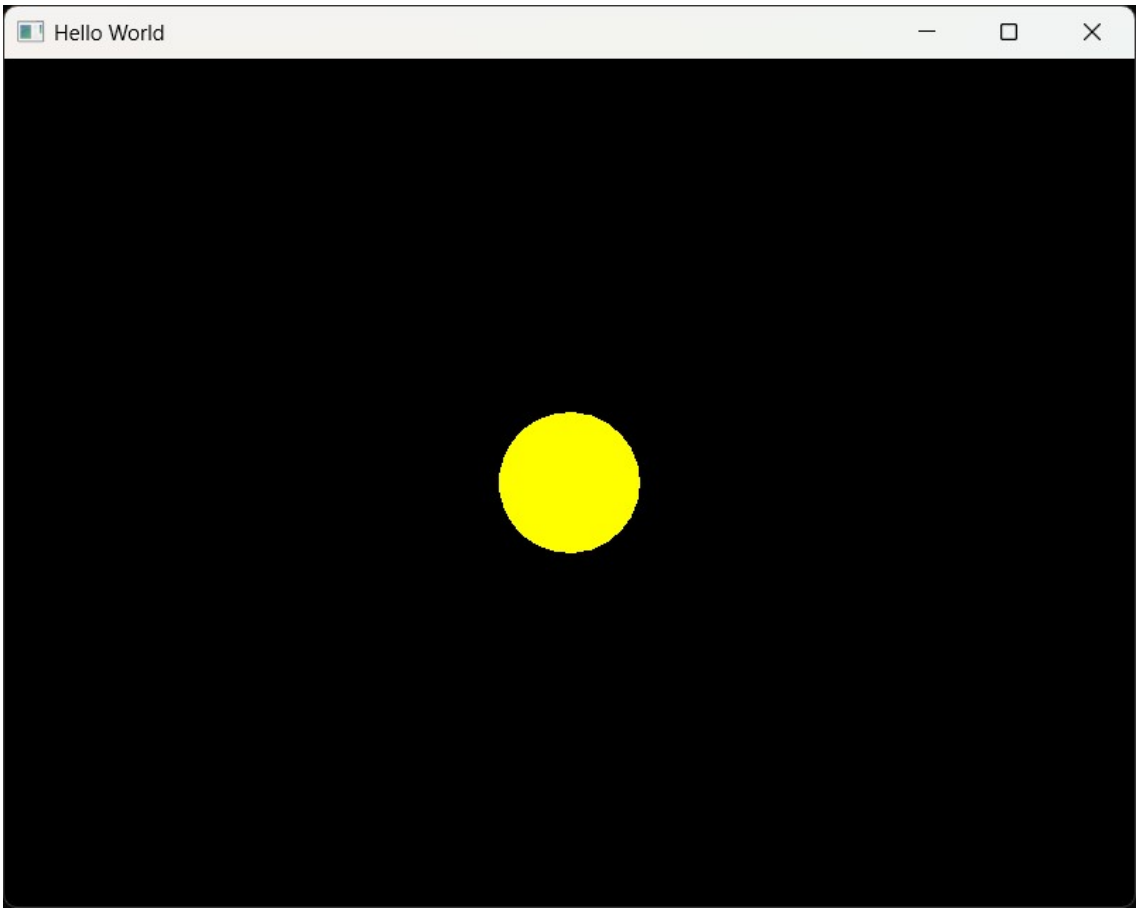

*Gambar 22 Dependency name*

9. Setelah semua selesai akan kembali ke jendela utama Project Properties. Klik **OK** dan akan kembali lagi ke tampilan kode editor.

Setelah semua langkah pengaturan properti selesai tuliskan kodenya, pastikan tidak ada garis merah (*syntax error*). Kode selengkapnya seperti berikut ini.

```
#include <sl.h>

int main()
{
    slWindow(800, 600, "Hello World", false);

    while (!slShouldClose())
    {
        slSetBackColor(0.0, 0.0, 0.0);
        slSetForeColor(1.0, 1.0, 0.0, 1.0);
        slCircleFill(400, 300, 50, 25);

        slRender();
    }

    slClose();

    return 0;
}
```

Jalankan, bila tidak ada kesalahan maka akan muncul tampilan seperti Gambar 23 berikut ini.




*Gambar 23 Hello SIGIL*

Penjelasan kode:

1. slWindow()
   Fungsi untuk membuat jendela dengan **slWindow(width, height, title, fullscreen)**. Untuk kode di atas artinya **width = 800, height 600, title = "Hello World", dan fullscreen = false**.
2. slShouldClose()
   Fungsi untuk mendeteksi apakah ada klik pada tombol silang (*close*) di jendela. Dalam contoh kode di atas tertulis **!slShouldClose()**, ada operator ! (NOT), diletakkan sebagai kondisi pada perulangan **WHILE**. Struktur perulangan tersebut berarti "selama tidak terdeteksi klik tombol *close* lakukan perintah berikut".
3. slSetBackColor()
   Fungsi untuk mengatur warna latar dengan format **slSetBackColor(red, green, blue)**. Komposisi warna tersusun dari kombinasi unsur **red, green,** dan **blue** dengan rentang intensitas antara **0.0 – 1.0**. Dalam contoh kode di atas berarti **red = 0.0, green = 0.0, blue = 0.0** yang akan menghasilkan warna hitam.
4. slSetForeColor()
   Fungsi untuk mengatur warna depan dengan format **slSetForeColor(red, green, blue, alpha)**. Hampir sama dengan warna latar, bedanya di sini ada tambahan unsur **alpha** (tingkat transparansi, 0 = transparan, 1 = solid).
5. slCircleFill()

Fungsi untuk menggambar lingkaran dengan *fill color* dari warna depan (mengikuti pengaturan slSetForeColor). Formatnya adalah **slCircleFill(x, y, radius, verteks).** Dalam contoh kode di atas berarti **x = 400, y = 300, radius = 50, verteks = 25**.
Catatan: verteks adalah jumlah titik penyusun, semakin banyak semakin halus.

6. slRender()
   Fungsi untuk menampilkan semua perintah menggambar agar ditampilkan di layar (render). Fungsi ini biasanya diletakkan paling akhir setelah semua proses menggambar.
7. slClose()
   Fungsi penutup setelah semua proses selesai. Fungsi ini di belakang layar akan menghapus penggunaan semua memori selama program berjalan.

Untuk dokumentasi selengkapnya dapat dilihat di *file* **SIGIL_API.pdf**.

## BouncingBall

*Project* **BouncingBall** akan mensimulasikan bola jatuh yang terkena gaya gravitasi. Bola memilik daya pantul sehingga saat menyentuh tanah akan memantul ke atas, berhenti, jatuh kembali dan seterusnya.

Buatlah *project* BouncingBall ini, tambahkan *file* **main.cpp** dan lakukan pengaturan seperti pada *project* Hello sebelumnya. Untuk mempercepat penulisan kode, bisa menyalin dari *project* sebelumnya. Berikut ini adalah potongan kodenya:

```
slWindow(800, 600, "Hello World", false);

float posX = 400.0f;                    // posisi X
float posY = 600.0f;                    // posisi Y
float velY = 0.0f;                      // kecepatan di sumbu Y
const float gravity = -0.098f;          // gravitasi, bernilai minus karena ke arah bawah

while (!slShouldClose())
{
          slSetBackColor(0, 0, 0);
          slSetForeColor(1.0, 1.0, 0.0, 1.0);
          slCircleFill(posX, posY, 25, 25);

          slRender();

          // perubahan velY (kecepatan Y)
          // dipengaruhi oleh gravity
          velY += gravity;
          // perubahan posY (posisi Y)
          // dipengaruhi oleh velY
          posY += velY;

          if (posY <= 0)
          {
                    // bila bola menyentuh dasar layar
                    // ubah arah kecepatan dengan mengalikan -1
                    velY *= -1;
          }
}

slClose();
```

Ubahlan nilai awal konstanta **gravity** untuk mengubah kecepatan jatuh bula.

## SpaceShip

*Project* ini adalah demo penggunaan *keyboard* untuk input. Pemain dapat menggunakan tombol panah untuk mengontrol pergerakan pesawat. Sebelum membuat *project* **SpaceShip** ini perlu dipersiapkan aset berupa gambar pesawat dan latar langit.

Aset dapat diunduh melalui *link* https://reps.yipsip.my.id/ST068-2025/07/assets.zip kemudian ekstraklah ke folder kerja. Untuk contoh ini aset berada di D:\exps\cpp\ST068\assets.

Berikut ini adalah potongan kodenya:

```
slWindow(600, 800, "Space Ship", false);

// lokasi file texture (gambar)
// sesuaikan dengan lokasi masing - masing
int spaceship = slLoadTexture("D:/exps/cpp/ST068/assets/spaceship.png");
int background = slLoadTexture("D:/exps/cpp/ST068/assets/background.jpg");

float posX = 300.0f;          // posisi X
float posY = 400.0f;          // posisi Y

while (!slShouldClose())
{
          slSetBackColor(0, 0, 0);

          // menggambar sprite yang berasal dari texture
          // sesuaikan dengan ukuran asli (width, height)
          slSprite(background, 300, 400, 600, 800);
          slSprite(spaceship, posX, posY, 64, 63);

          slRender();

          // kontrol pesawat dengan tombol panah
          // bergerak ke kanan, menambah nilai posX
          if (slGetKey(SL_KEY_RIGHT))
                    posX += 2;
          // begerak ke kiri, mengurangi nilai posX
          if (slGetKey(SL_KEY_LEFT))
                    posX -=2;
          // bergerak ke atas, menambah nilai posY
          if (slGetKey(SL_KEY_UP))
                    posY += 2;
          // bergerak ke bawah, mengurangi nilai posY
          if (slGetKey(SL_KEY_DOWN))
                    posY -= 2;
}

slClose();
```

Jalankan program dan coba gerakkan pesawat menggunakan tombol panah. Ubah nilai penambah atau pengurang untuk mengganti kecepatan gerak pesawat. Sebagai contoh untuk mempercepat gerakan di sumbu X bisa diubah menjadi **posX += 4** dan **posY -= 4**, dan seterusnya.

Gambar 24 berikut ini adalah tampilan saat dijalankan:


*Gambar 24 SpaceShip*

Dalam folder **assets** terdapat sebuah gambar bernama **asteroids.png**. Tambahkan asteroid tersebut ke dalam layar, dengan ketentuan:

1. Asteroid bergerak dari posisi **Y = 850** ke **Y = -40**.
2. Bila asteroid sudah mencapai posisi **Y = - 40**, kembalikan lagi posisi **Y = 850**, dan **posisi X** dibuat acak.
3. Manfaatkan fungsi **rand()** untuk mengacak posisi X (lihat kembali project **Dice** untuk melihat kembali penggunaan fungsi acak).