## 1. Architektura Potoku Graficznego (Graphics Pipeline)

Współczesne programowanie grafiki opiera się na programowalnym potoku renderowania, w którym kluczową rolę odgrywają jednostki cieniujące (**Shaders**). Są to programy wykonywane równolegle na procesorze graficznym (GPU), operujące na danych geometrycznych i rastrowych.

- **Vertex Shader:** Realizuje transformacje wierzchołków z przestrzeni lokalnej modelu do przestrzeni rzutowania (Clip Space). Odpowiada za obliczenia geometryczne, transformacje macierzowe ($MVP$) oraz przekazywanie danych do kolejnych etapów.
- **Fragment (Pixel) Shader:** Wykonywany dla każdego wygenerowanego fragmentu (potencjalnego piksela). Jego zadaniem jest wyznaczenie końcowej wartości koloru ($RGBA$) na podstawie danych interpolowanych, tekstur oraz parametrów oświetlenia.

---

## 2. Formalny Model Blendingu

Blending jest operacją realizowaną w etapie **Output Merger (OM)**, następującym po rasteryzacji i wykonaniu fragment shaderów. Polega on na matematycznym połączeniu koloru wyjściowego z shadera (źródło) z kolorem aktualnie znajdującym się w buforze ramki (cel).

Podstawowe równanie blendingu definiuje się jako:

$$C_{out} = (C_{src} \cdot F_{src}) \text{ op } (C_{dst} \cdot F_{dst})$$

Gdzie:

- $C_{src}$ (**Source Color**): Wektor koloru zwrócony przez fragment shader.
- $F_{src}$ (**Source Factor**): Współczynnik wagowy dla koloru źródłowego.
- $C_{dst}$ (**Destination Color**): Kolor aktualnie zapisany w buforze ramki.
- $F_{dst}$ (**Destination Factor**): Współczynnik wagowy dla koloru docelowego.
- $\text{op}$ (**Blend Operation**): Operacja binarna (zazwyczaj Add).

### Zarządzanie Buforem Głębokości i Kolejkami

Zastosowanie blendingu wymaga precyzyjnej kontroli nad buforem głębokości (**Z-Buffer**). W przypadku obiektów półprzezroczystych (Transparent) konieczne jest:

1. **Wyłączenie zapisu do Z-Buffera** (ZWrite Off).
2. **Sortowanie (Back-to-Front)**: Obiekty muszą być rysowane w kolejności od najdalszego do najbliższego.
3. **Render Queues**: W Unity proces ten kontroluje tag "Queue"="Transparent".

---

### 3. Implementacja w ShaderLab (Sekcja Praktyczna)

W środowisku Unity shadery tekstowe definiowane są w języku **ShaderLab**, stanowiącym opakowanie dla kodu **HLSL**. Poniżej znajduje się struktura shadera z zaimplementowanym mechanizmem przezroczystości.

### Szkielet Shadera

```
Shader "Folder/ShaderName"
{
  Properties
  {
    _MainTex ("Texture", 2D) = "white" {}
    _Color ("Main Color", Color) = (1,1,1,1)
  }
  SubShader
  {
    Tags { "Queue" = "Transparent" "RenderType" = "Transparent" }

    ZWrite Off

    Pass
    {
      CGPROGRAM
      #pragma vertex vert
      #pragma fragment frag
      #include "UnityCG.cginc"

      struct appdata {
        float4 vertex : POSITION;
        float2 uv : TEXCOORD0;
      };

      struct v2f {
        float4 pos : SV_POSITION;
        float2 uv : TEXCOORD0;
      };

      sampler2D _MainTex;
      float4 _Color;

      v2f vert (appdata v) {
        v2f o;
        o.pos = UnityObjectToClipPos(v.vertex);
        o.uv = v.uv;
        return o;
      }

      fixed4 frag (v2f i) : SV_Target {
        fixed4 col = tex2D(_MainTex, i.uv) * _Color;
        return col;
      }
      ENDCG
    }
  }
}
```

# 4. Zadania do wykonania

## Zadanie 1

Wykorzystując parametry SrcAlpha oraz OneMinusSrcAlpha, zaimplementuj shader pozwalający na uzyskanie efektu przezroczystości typowej dla szkła.

**Wymaganie**: Obiekt musi poprawnie wyświetlać tło, a stopień jego widoczności musi być płynnie sterowany z poziomu inspektora Unity

## Zadanie 2

Zrealizuj dwa warianty shadera (lub dwa osobne materiały), wykorzystując alternatywne operacje mieszania kolorów.

**Wariant A (Addytywny)**: Uzyskaj efekt, w którym czarne fragmenty tekstury są całkowicie ignorowane, a jasne rozjaśniają tło (efekt światła).

**Wariant B (Multiplikatywny)**: Uzyskaj efekt, w którym białe fragmenty tekstury są przezroczyste, a ciemne barwią lub przyciemniają tło (efekt filtru/cienia).

## Zadanie 3

Doprowadź do sytuacji, w której obiekt półprzezroczysty błędnie przesłania inne obiekty znajdujące się za nim.

**Wymaganie**: Przy aktywnym blendingu (Zadanie 1) zmień konfigurację bufora głębokości tak, aby karta graficzna odrzucała fragmenty tła znajdujące się za obiektem przezroczystym.