３Ｄの初期設定と球体の画面表示

手始めに、DxLibで標準装備されている３Ｄ球体の表示機能を使って、
画面に３Ｄ球体を表示してみましょう。

1. Titleシーンで、３Ｄ球体の表示

Draw関数内で、

```
//VECTOR pos = VGet(0.0f, 0.0f, 0.0f);
VECTOR pos = { 0.0f, 0.0f, 0.0f };

// ３Ｄ球体の描画
//（位置、半径、ポリゴン数、拡散反射光、鏡面反射光、塗りつぶし）
// 拡散反射光：オブジェクト自体が持つ色
// 鏡面反射光：光の入射方向とは反対方向へと反射する光
DrawSphere3D(pos, 50.0f, 10, 0xffffff, 0xffffff, true);
```

VECTORとは、１年生の時に使っていた、Vector2F(float型のxとy)に近い存在です。
DxLibで定義されている構造体で、float型のxとyとzを持っています。

```
struct VECTOR
{
   float        x, y, z ;
};
```

今後、３Ｄ制御(大きさ、回転、座標)を行うにあたり、
相当な頻度で使用していきますので、早めに慣れておきましょう。
構造体になりますので、３つの数字を設定する際に、

```
VECTOR pos = { 0.0f, 0.0f, 0.0f };
```

このように省略して、左から順番に、x、y、z と代入できますが、
キチンと型を明示的にしたい場合は、VGetを使用しましょう。

```
VECTOR pos = VGet(0.0f, 0.0f, 0.0f);
```

DrawSphere3Dは、DxLibの球体描画関数ですが、
引数の意味はリファレンスか、コメントをご参照ください。

画面左下にうっすらと球体が見えますでしょうか？
これが３Ｄの嫌なところです。
いくつかの条件をクリアしないと、画面に表示されず、不具合の直し方も
わからず、ストレスが溜まり、妥協してしまいがちです。

そんな時は、素直に周りの人たちに頼って、パターンを掴んでいきましょう。

2. ３Ｄに関する初期設定(背景色)

SceneManagerで３Ｄ専用の初期設定を行いましょう。
Init関数から新しく作成したInit3Dを呼び出しましょう。

```
void SceneManager::Init3D(void)
{
    // 背景色設定
    SetBackgroundColor(0, 139, 139);
}
```

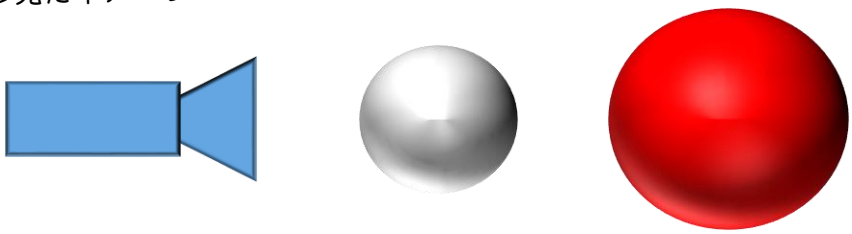
背景色のデフォルトが黒色になっているので、緑色にしてみましょう。

今度は、先ほどよりも３Ｄの球体がハッキリ見えるようになりました。
３Ｄはライトの関係で、何も設定しないとモデルが黒や白に近い色で
表示されてしまいますので、慣れないうちは、黒や白が見えやすいように
背景色の色を変えておきましょう。

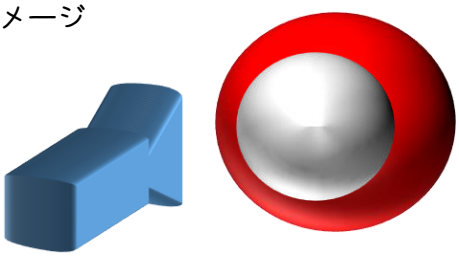
3. ３Ｄに関する初期設定(Zバッファ)

次に、Titleシーンに戻って、
最初の球体よりも奥側に、もっと大きな赤い球体を追加しましょう。

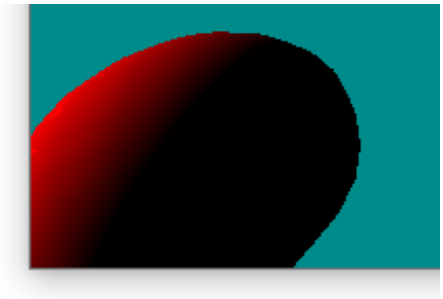
真横から見たイメージ



斜めから見たイメージ

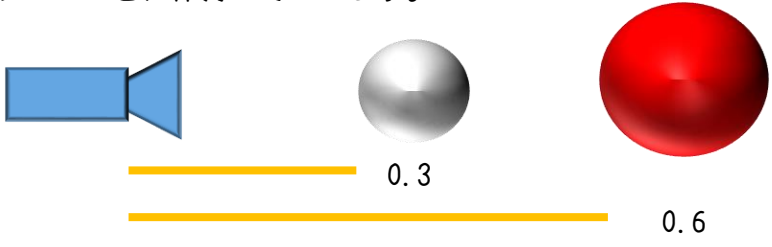


赤い球体を追加したら、実行してみて、表示を確認しましょう。

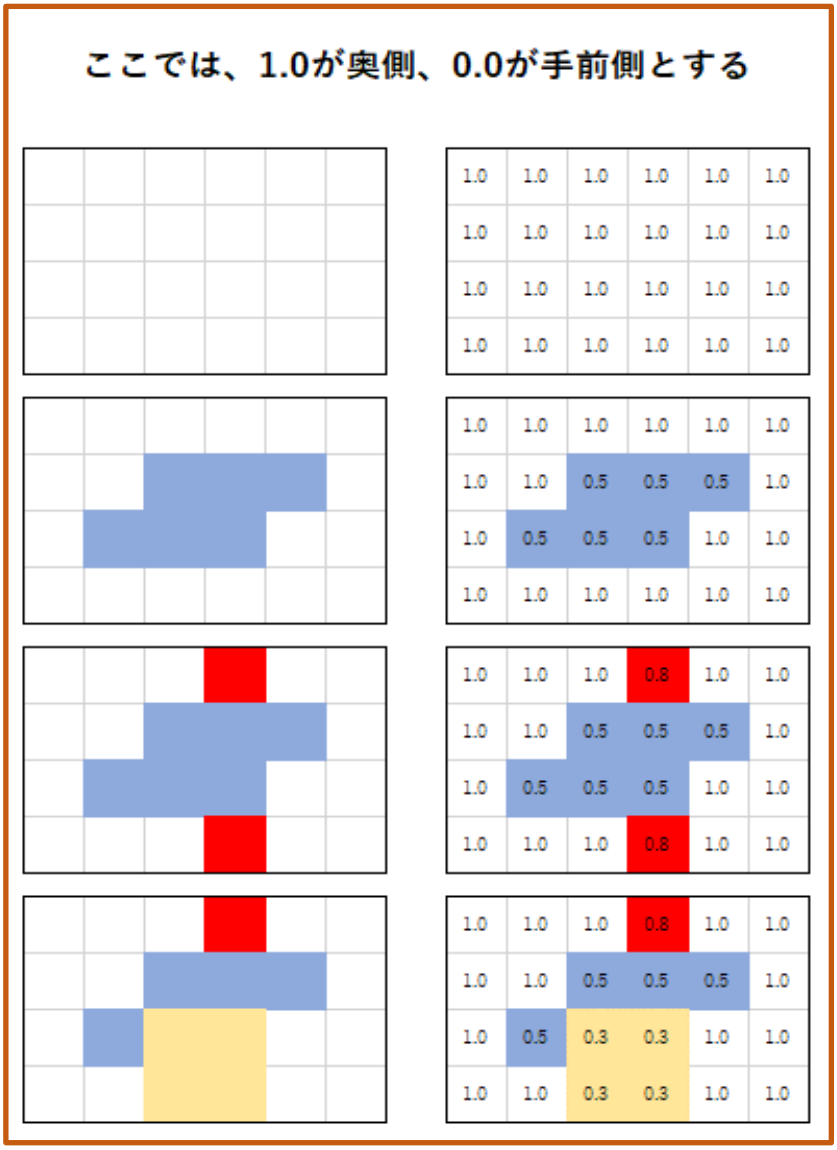


白い球体が手前に来ているはずなのに、赤色の球体で塗り潰されています。
描画順を変更すれば、白い球体が手前に表示されますが、
今回の対応としては正しくありません。
Zバッファという、奥行きの制御を行う必要があります。

カメラの位置から、頂点座標までの奥行の長さをZ値(深度値)として、
ピクセルごとに保持しておきます。



そうすると、このように
描画順が入れ子になっていてもキチンと前後関係を保てます。

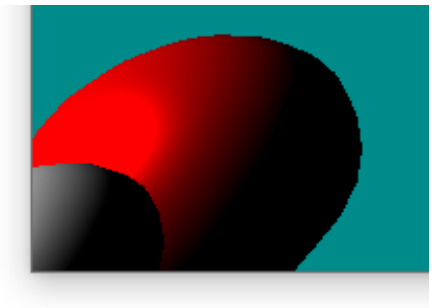


このZバッファを有効にするため、Init3D関数に下記設定を追加します。

```
// Zバッファを有効にする
SetUseZBuffer3D(true);

// Zバッファへの書き込みを有効にする
SetWriteZBuffer3D(true);
```

赤色の球体の手前に、白色の球体が描画されました。

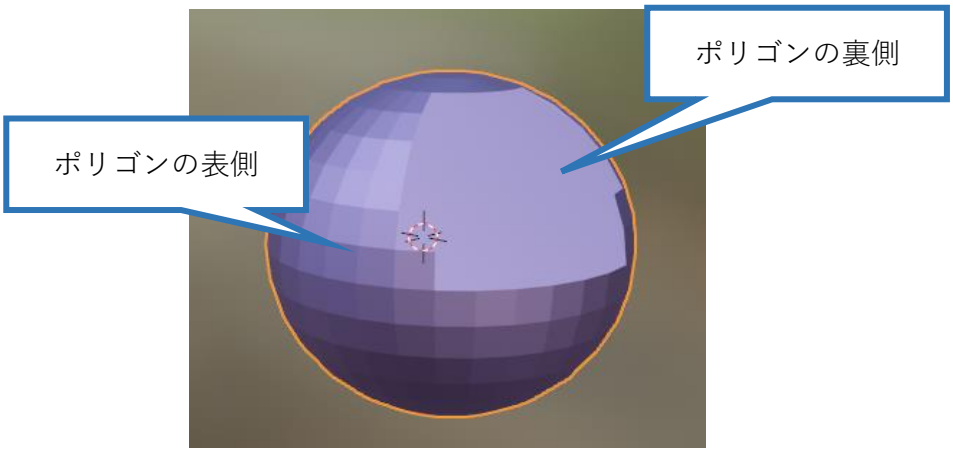


4. ３Ｄに関する初期設定(カリング)

描画する必要がないポリゴンは描画しないようにする設定のことを
カリングといいます。
描画しないようにすることで、処理負荷を軽減できます。

視錐台カリング、オクルージョンカリングなどありますが、
今回は、ポリゴンの裏面を描画しないという設定を行います。
これを、バックカリングといいます。

イメージしづらいかもしれませんが、



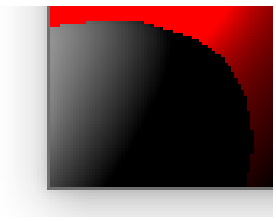
カメラから見て、面の表側は描画する、裏側は描画しない設定です。
DxLibの設定は以下のとおり。

```
// バックカリングを有効にする
SetUseBackCulling(true);
```

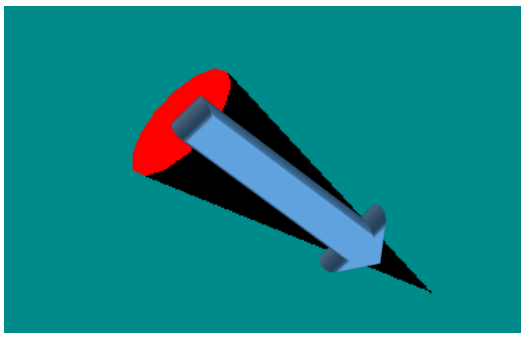
見た目は変わっていないかと思いますが、処理負荷は軽減されています。

5. ３Ｄに関する初期設定(ライト)

手前に描画されている白の球体ですが、白色に見えません。



これは、ライトによる影響です。
デフォルトでは、{ 0.578f, -0.578f, 0.578f }の方向を向いた、
ディレクショナルライトが設定されています。
ライトの種類については、また別紙で解説しますが、

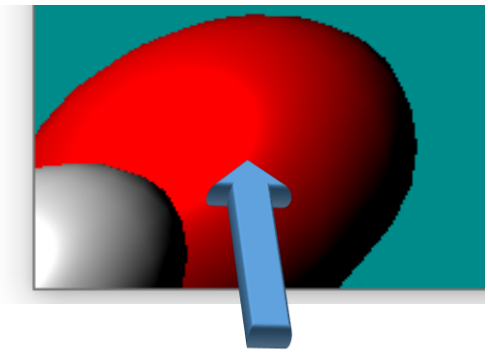


矢印の方向に光が当たるような設定になっていますので、
球体の右側が影に隠れ、黒くなっている状態になっています。

これを、正面斜め下にライトの向きを変えてみます。

```
// ライトを有効にする
SetUseLighting(true);

// ディレクショナルライト方向の設定(正規化されていなくても良い)
// 正面から斜め下に向かったライト
ChangeLightTypeDir({ 0.00f, -1.00f, 1.00f });
```



正面からライトが当たるようになりました。
ライトとカメラの向きと近くなり、白っぽくなりました。

ある程度、まともに３Ｄ球体が表示されるようになりましたが、
位置が｛ 0.0f, 0.0f, 0.0f ｝に設定されているのに
なぜ左下に表示されているのでしょうか。

次は、カメラの設定に移ります。