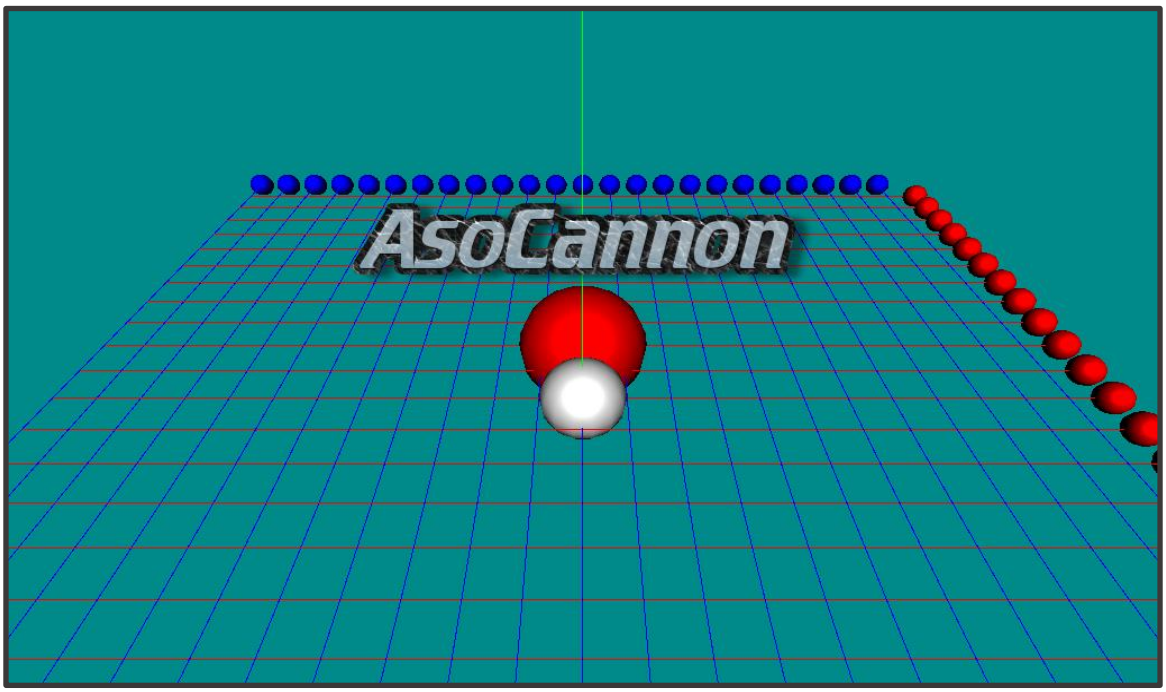
グリッドの作成

３Ｄの初期設定や、カメラ設定を行っていく上で、
３Ｄワールドのイメージが難しく、迷子になることもしばしばあります。
そんな時は、立体空間を把握しやすくするためのグリッド表示を行います。

グリッドとは？　⇒　格子、方眼状のもの。

イメージ的にはこのような線と、正方向の先には球体を表示していきます。
赤がX軸、緑がY軸、青がZ軸です。



３Ｄの線を引くためには、球体と同じようにDxLibの機能を使います。

```
int DrawLine3D( VECTOR Pos1, VECTOR Pos2, unsigned int Color ) ;
```

３Ｄ空間に線分を描画する。

VECTOR Pos1　：　線分の始点の座標
VECTOR Pos2　：　線分の終点の座標
unsigned int Color　：　線分の色

プロジェクトが変わっても、サッと導入できるように、
専用のクラスを作っていきましょう。

Grid.h

```
#pragma once
class Grid
{

public:

    // 線の長さ
    static constexpr float LEN = 1200.0f;

    // 線の長さの半分
    static constexpr float HLEN = LEN / 2.0f;

    // 線の間隔
    static constexpr float TERM = 50.0f;

    // 線の数
    static const int NUM = static_cast<int>(LEN / TERM);

    // 線の数の半分
    static const int HNUM = NUM / 2;

    // コンストラクタ
    Grid(void);

    // デストラクタ
    ~Grid(void);

    void Init(void);
    void Update(void);
    void Draw(void);
    void Release(void);

};
```

これらの定数は使っても、
使わなくとも大丈夫です。
ご参考までに。

最初は練習で、Z座標0.0fに赤色の横一本の線を描画してみましょう。

Grid.cpp

```
void Grid::Draw(void)
{
    // 【練習】最初の１本
    VECTOR sPos = { 0.0f, 0.0f, 0.0f };
    VECTOR ePos = { HLEN, 0.0f, 0.0f };
    DrawLine3D(sPos, ePos, 0xff0000);
};
```

サンプルでは、長さ1200の線を描画していますので、
反対の左側まで線を伸ばしていきます。

```
    VECTOR sPos = { -HLEN, 0.0f, 0.0f };
    VECTOR ePos = {  HLEN, 0.0f, 0.0f };
```

これで上手くX軸の線が描画されました。
まずは、X軸の描画を完成させて、
次にZ軸の描画に取り組んでいきましょう。