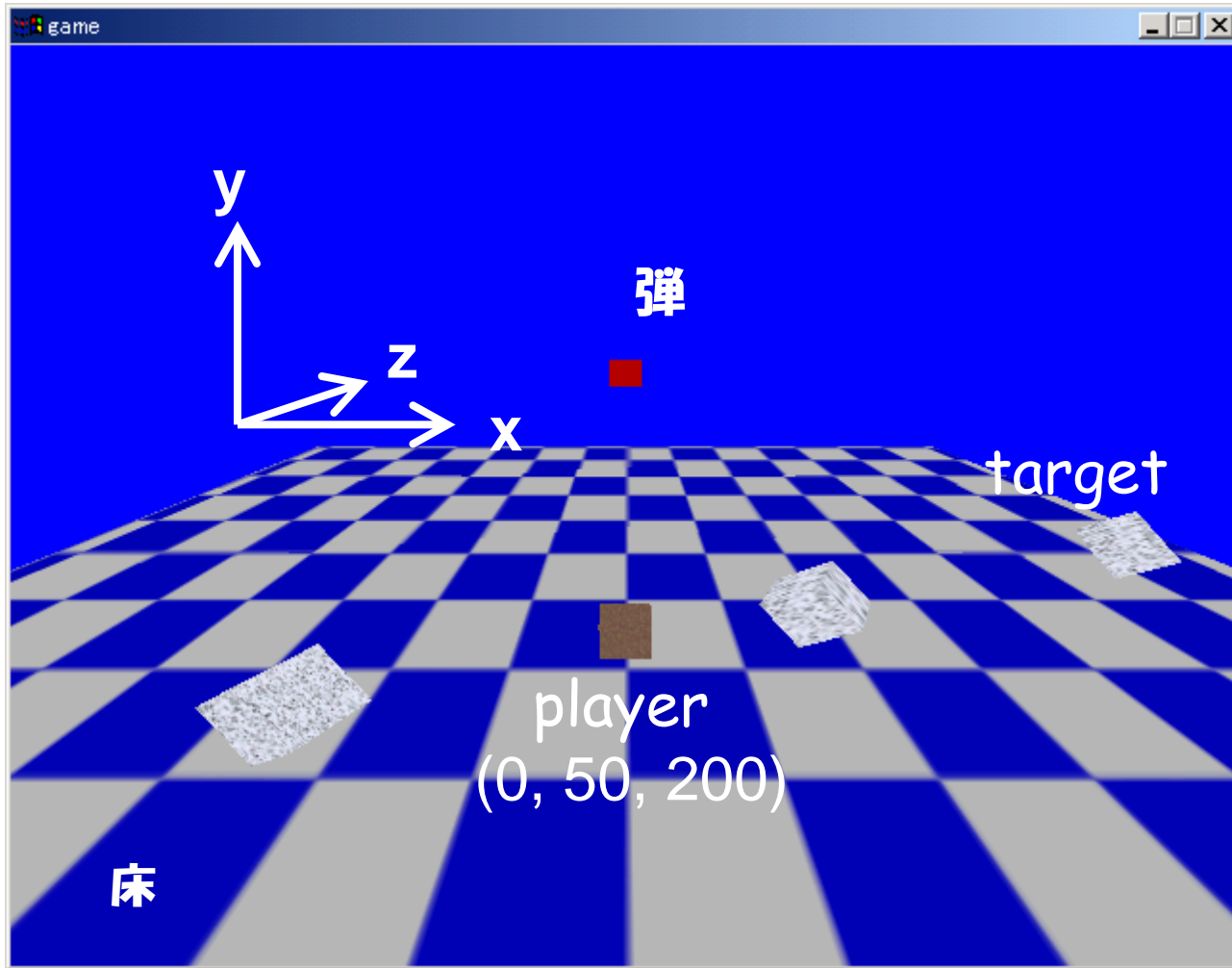


# DirectX勉強会 第5回

# 座標、物体の位置について



# 移動する物体について (target)

- 位置の管理
  - `_TpositionX[j]` : j番目のtargetのx座標
  - `_TpositionY[j]` : j番目のtargetのy座標
  - `_TpositionZ[j]` : j番目のtargetのz座標
- 移動 (プログラム Line468~)
  - 一回描画する度に、x座標を更新していく
    - `_TpositionX[j] = _TpositionX[j] + “移動距離”;`

# 移動する物体について (target)

- 描画 (プログラム Line446~)

```
D3DXMatrixRotationY(&matRotateY,(float)timeGetTime() / 1800.0f);  
D3DXMatrixRotationX(&matRotateX,(float)timeGetTime() / 920.0f);  
D3DXMatrixTranslation(&matTrans,_TpositionX[j], _TpositionY[j],  
_TpositionZ[j]);
```

```
matWorld = matRotateX * matRotateY * matTrans;
```

```
lpD3DDEV->SetTransform(D3DTS_WORLD,&matWorld);
```

```
// 使用する頂点バッファの設定
```

```
lpD3DDEV->SetStreamSource(0,vb[i],sizeof(LVERTEX));
```

```
// 描画する
```

```
lpD3DDEV->DrawPrimitive(D3DPT_TRIANGLEFAN,0,2);
```

# 移動する物体について (弾)

- 位置の管理
  - `_BpositionX[j]` : j番目の弾のx座標
  - `_BpositionY[j]` : j番目の弾のy座標
  - `_BpositionZ[j]` : j番目の弾のz座標
- 移動 (プログラム Line517~)
  - 一回描画する度に、y座標を更新していく
    - `_BpositionY[j] = _BpositionY[j] + “移動距離”;`
- 描画 (プログラム Line500~)

# キー入力の処理について

- プログラム全体の流れを説明
- 主な関数
  - WinMain()
  - procMain()
  - Move()
  - DrawMain()

# 主な関数

WinMain()

ウィンドウの初期化  
Dinputの初期化 など

メインウィンドウループ

後処理

procMain()

キー入力はあるか？

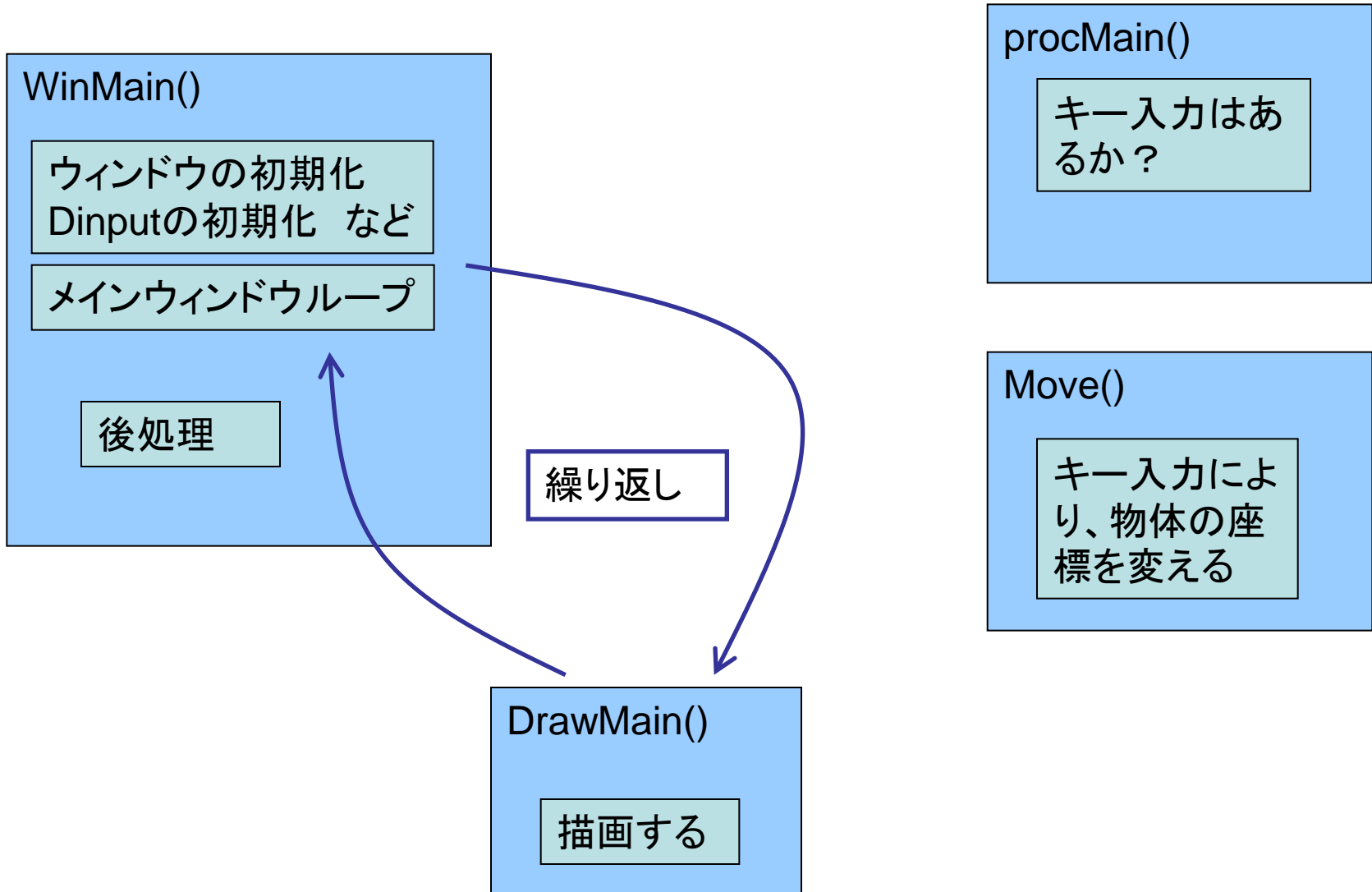
Move()

キー入力により、物体の座標を変える

DrawMain()

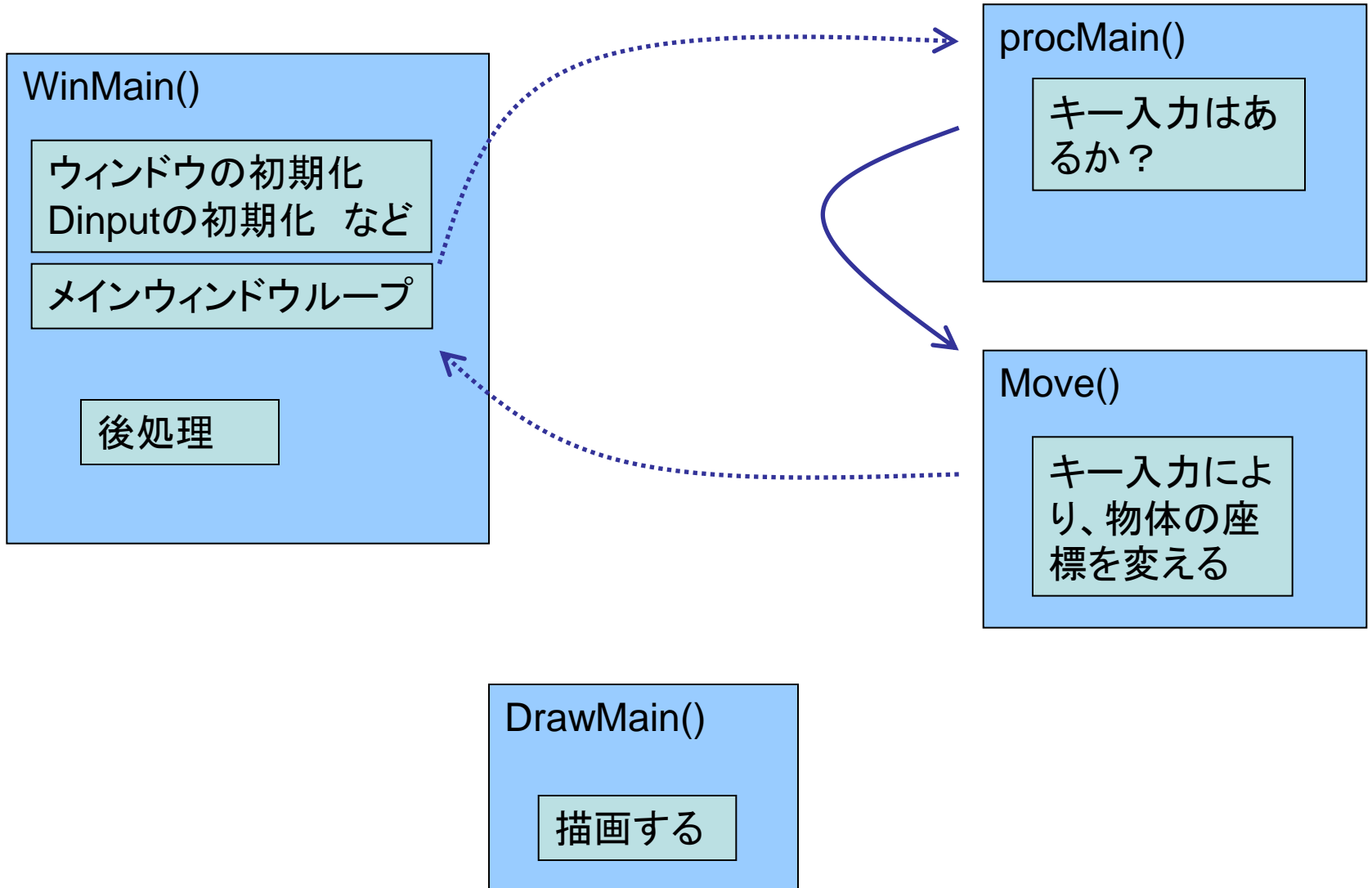
描画する

# キー入力が無い間





# キー入力があった時



# メインウインドウループ (WinMain()の中)

- PeekMessage()  
メッセージが送られてきているかどうか調べる

# メインウインドウループ (WinMain()の中)

```
MSG msg;
while (!end)
{
    // メッセージが来ているか?
    if (PeekMessage(&msg, NULL, 0, 0, PM_REMOVE))
    {
        // メッセージの処理
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }
    else
    {
        // 描画処理
        DrawMain();
    }
}
```

windows経由で、  
procMain()を呼び出す

DrawMain()を  
呼び出す

# procMain()

- メッセージがあると、Windowsから、この関数が呼ばれる
- msgに入っている値により、処理を決定できる
  - WM\_CLOSE :  
ウィンドウ上部の[×]ボタンを押したとき等に、この値になる。  
→アプリケーションを終了する処理をする
  - WM\_KEYDOWN :  
キーが押されたときに、この値になる  
→今回は、Move()を呼び出している

# procMain()

```
LRESULT CALLBACK procMain(HWND hWnd,UINT msg,WPARAM  
wParam,LPARAM lParam)  
{  
    switch (msg)  
    {  
        case WM_CLOSE:  
            // 終了フラグ  
            end = 1;  
            return 1;  
        case WM_KEYDOWN:  
            Move();  
            break;  
    }  
    return DefWindowProc(hWnd,msg,wParam,lParam);  
}
```

キーが押されたので、  
Move()を呼び出す

# Move()

- DirectInputを利用して、どのキーが押されたかを判定する
- 押されたキーに応じて、playerの位置を動かしたり、弾を発射したりする

# Move()

```
// キー入力を受けとって、物体の位置を移動する
int Move() {
    ...
    BYTE diKeyState[256]; //この構造体にデータを取得
    hr = g_pDIDevice->GetDeviceState(sizeof(BYTE)*256, &diKeyState[0]);
    if (SUCCEEDED(hr)) {
        //直接データの取得に成功
        if (diKeyState[DIK_Q] & 0x80) {
            ...
        }
        else if (diKeyState[DIK_H] & 0x80) {
            //「h」が入力された場合は、playerが左へ移動
            _PpositionX = _PpositionX - 50.0f;
        }
        ...
        else if (diKeyState[DIK_SPACE] & 0x80) {
            ...
        }
    }
}
```

キーqが押されたので、  
終了処理をする

キーhが押されたので、  
playerの位置を変える

スペースキーが押され  
たので、弾を発射する

# キー入力の処理について (player)

- 位置の管理
  - `_PpositionX` : playerのx座標
  - `_PpositionY` : playerのy座標
  - `_PpositionZ` : playerのz座標
- 移動 (プログラム Line326~)
  - 押されたキーによって、x座標(左右の移動)かz座標(手前・奥の移動)を更新する
    - `_PpositionX = _PpositionX + “移動距離”;`
    - `_PpositionZ = _PpositionZ + “移動距離”;`



# キー入力の処理について (player)

- 描画 (プログラム Line483~)

```
// World 行列設定
D3DXMatrixTranslation(&matTrans, _PpositionX , _PpositionY, _PpositionZ);
matWorld = matTrans;
lpD3DDEV->SetTransform(D3DTS_WORLD,&matWorld);

// 使用する頂点バッファの設定
lpD3DDEV->SetStreamSource(0,vb[i],sizeof(LVERTEX));
// 描画する
lpD3DDEV->DrawPrimitive(D3DPT_TRIANGLEFAN,0,2);
```

# 実習

- game.cppを改良してみる。

## – 例

- 物体のtexture、透過度、形を変えてみる
- 視点を変えてみる
- 弾、targetの速さを変えてみる
- 弾の軌道を変えてみる(斜めに飛んでいく等)
- playerを移動させるキーを変えてみる

これ以外にもいろいろとあると思うので、自由にやってみてください