Chapter 1

キーボードのＡが押されたらプレイヤーの移動速度を倍にしてみよう。

まず､キーボードのＡが押されたということを記録できるようにする必要があります。Packman のプロジェクトでキーボードの入力を記録しているプログラムは下記の二のファイルです。

ソースファイル

.¥Packman¥tkEngine¥Input¥tkInput¥tkKeyInput.cpp

ヘッダーファイル

.¥Packman¥tkEngine¥Input¥tkInput¥tkKeyInput.h

ではヘッダーファイルを下記のように編集してください。太字になっている部分が変更点です。

```
/*!
 * @brief	キー入力。
 */

#ifndef _TKKEYINPUT_H_
#define _TKKEYINPUT_H_

namespace tkEngine{
	class CKeyInput{
		enum EnKey{
			enKeyUp,
			enKeyDown,
			enKeyRight,
			enKeyLeft,
			enKeyA,
			enKeyNum,
		};
	public:
		/*!
		 * @brief	コンストラクタ。
		 */
		CKeyInput();
		/*!
		 * @brief	デストラクタ。
		 */
		~CKeyInput();
		/*!
		 * @brief	キー情報の更新。
		 */
		void Update();
		/*!
		 * @brief	上キーが押されている。
		 */
		bool IsUpPress() const
		{
			return m_keyPressFlag[enKeyUp];
		}
		/*!
		 * @brief	右キーが押されている。
		 */
		bool IsRightPress() const
		{
			return m_keyPressFlag[enKeyRight];
		}
```

```
		/*!
		 * @brief		左キーが押されている。
		 */
		bool IsLeftPress() const
		{
			return m_keyPressFlag[enKeyLeft];
		}
		/*!
		 * @brief		下キーが押されている。
		 */
		bool IsDownPress() const
		{
			return m_keyPressFlag[enKeyDown];
		}
		/*!
		* @brief		キーボードのAが押された。
		*/
		bool IsAPress() const
		{
			return m_keyPressFlag[enKeyA];
		}
	private:
		bool	m_keyPressFlag[enKeyNum];
	};
}
#endif //_TKKEYINPUT_H_
```

続いてソースファイルを下記のように変更します。

```
/*!
 * @brief		キー入力
 */

#include "tkEngine/tkEnginePreCompile.h"
#include "tkEngine/Input/tkKeyInput.h"

namespace tkEngine{
	/*!
	 * @brief		コンストラクタ。
	 */
	CKeyInput::CKeyInput()
	{
		memset(m_keyPressFlag, 0, sizeof(m_keyPressFlag));
	}
	/*!
	 * @brief		デストラクタ。
	 */
	CKeyInput::~CKeyInput()
	{
	}
	/*!
	 * @brief		キー情報の更新。
	 */
	void CKeyInput::Update()
	{
		if (GetAsyncKeyState(VK_UP) & 0x8000) {
			m_keyPressFlag[enKeyUp] = true;
		}
		else {
			m_keyPressFlag[enKeyUp] = false;
		}
		if (GetAsyncKeyState(VK_DOWN) & 0x8000) {
			m_keyPressFlag[enKeyDown] = true;
		}
```

```
                    else {
                        m_keyPressFlag[enKeyDown] = false;
                    }
                    if (GetAsyncKeyState(VK_RIGHT) & 0x8000) {
                        m_keyPressFlag[enKeyRight] = true;
                    }
                    else {
                        m_keyPressFlag[enKeyRight] = false;
                    }
                    if (GetAsyncKeyState(VK_LEFT) & 0x8000) {
                        m_keyPressFlag[enKeyLeft] = true;
                    }
                    else {
                        m_keyPressFlag[enKeyLeft] = false;
                    }
                    if ((GetAsyncKeyState('A') & 0x8000)
                        | (GetAsyncKeyState('a') & 0x8000) ) {
                        m_keyPressFlag[enKeyA] = true;
                    }
                    else {
                        m_keyPressFlag[enKeyA] = false;
                    }
            }
}
```

これでキーボードの A が入力されると、m_keyPressFlag[enKeyA]に true という値が記録されるようになりました。

では本当にキーボードの A が入力されたら true が設定されるか確認してみましょう。

先ほどの書き換えた部分にコードを下記の黒字のコードを追加してみて下さい。

```
if ((GetAsyncKeyState('A') & 0x8000)
   | (GetAsyncKeyState('a') & 0x8000) ) {
   m_keyPressFlag[enKeyA] = true;
   MessageBox(NULL, "A ボタンが押されたよ！", "成功", MB_OK);
}else {
   m_keyPressFlag[enKeyA] = false;
}
```

A を押したらダイアログボックスがでましたか？

では、確認ができたらメッセージボックスを表示するコードは削除してください。

では、正しくキーボードから入力を受け取ることができることが確認できたのでプレイヤーの移動速度を倍にしてみましょう。プレイヤーの移動処理は下記のファイルに記述されています。

ソースファイル

Packman¥Packman¥game¥Player¥ CPlayer.cpp

CPlayer.cpp を下記のように書き換えてみて下さい。

```
/*!
 *@brief        プレイヤー
 */

#include "stdafx.h"
#include "Packman/game/Player/CPlayer.h"
```

```
#include "Packman/game/CGameManager.h"

/*!
 *@brief	Update が初めて呼ばれる直前に一度だけ呼ばれる処理。
 */
void CPlayer::Start()
{
}
/*!
*@brief Update 関数が実行される前に呼ばれる更新関数。
*/
void CPlayer::PreUpdate()
{
	Move();
}
/*!
 *@brief	更新処理。60fps なら 16 ミリ秒に一度。30fps なら 32 ミリ秒に一度呼ばる。
 */
void CPlayer::Update()
{
	m_sphere.SetPosition(m_position);
	m_sphere.UpdateWorldMatrix();
	CGameManager& gm = CGameManager::GetInstance();
	CMatrix mMVP = gm.GetGameCamera().GetViewProjectionMatrix();
	const CMatrix& mWorld = m_sphere.GetWorldMatrix();
	m_wvpMatrix.Mul(mWorld, mMVP);
	m_idMapModel.SetWVPMatrix(m_wvpMatrix);
	IDMap().Entry(&m_idMapModel);
	m_shadowModel.SetWorldMatrix(mWorld);
	ShadowMap().Entry(&m_shadowModel);
}
/*!
*@brief 移動処理。
*/
void CPlayer::Move()
{
	float moveSpeed = 0.02f;  //移動速度。
	if (KeyInput().IsAPress()) {
		//キーボードの A 押されていたら速度を倍にする。
		moveSpeed *= 2.0f;
	}
	if (KeyInput().IsUpPress()) {
		m_position.z += moveSpeed;
	}
	if (KeyInput().IsDownPress()) {
		m_position.z -= moveSpeed;
	}
	if (KeyInput().IsRightPress()) {
		m_position.x += moveSpeed;
	}
	if (KeyInput().IsLeftPress()) {
		m_position.x -= moveSpeed;
	}
}
/*!
 *@brief	描画処理。60fps なら 16 ミリ秒に一度。30fps なら 32 ミリ秒に一度呼ばれる。
 */
void CPlayer::Render(tkEngine::CRenderContext& renderContext)
{
	CGameManager& gm = CGameManager::GetInstance();
	m_sphere.RenderLightWVP(
		renderContext,
		m_wvpMatrix,
		gm.GetFoodLight(),
		false,
		true
	);
}
/*!
 *@brief	構築。
```

```
 *必ず先に CreateShape を一度コールしておく必要がある。
 */
void CPlayer::Build( const CVector3& pos )
{
	m_sphere.Create(0.08f, 10, 0xffff0000, true );
	m_idMapModel.Create(m_sphere.GetPrimitive());
	m_shadowModel.Create(m_sphere.GetPrimitive());
	m_position = pos;
}
```

Chapter 2

ジャンプできるようにしてみよう。

Chapter1 のプログラムを改造して、キーボードの A が入力されたらプレイヤーがジャンプするようにしてみましょう。今回編集するプログラムは下記のファイルになります。

ソースファイル

Packman¥Packman¥game¥Player¥ CPlayer.cpp

ヘッダーファイル

Packman¥Packman¥game¥Player¥ CPlayer.h

CPlayer.h を下記のように編集してください。

```
/*!
 * @brief プレイヤー
 */
#ifndef _CPLAYER_H_
#define _CPLAYER_H_

#include "tkEngine/shape/tkSphereShape.h"

class CPlayer : public tkEngine::IGameObject{
public:
	CPlayer() :
		m_position(CVector3::Zero)
	{
	}
	~CPlayer(){}
	/*!
	 *@brief		Update が初めて呼ばれる直前に一度だけ呼ばれる処理。
	 */
	void Start() override final;
	/*!
	*@brief Update 関数が実行される前に呼ばれる更新関数。
	*/
	void PreUpdate() override final;
	/*!
	 *@brief		更新処理。60fps なら 16 ミリ秒に一度。30fps なら 32 ミリ秒に一度呼ばれる。
	 */
	void Update() override final;
	/*!
	 *@brief		描画処理。60fps なら 16 ミリ秒に一度。30fps なら 32 ミリ秒に一度呼ばれる。
	 */
	void Render(tkEngine::CRenderContext& renderContext) override final;
	/*!
	 *@brief		構築。
	 *必ず先に CreateShape を一度コールしておく必要がある。
	 */
	void Build( const CVector3& pos );
	/*!
```

```
    *@brief 移動処理。
    */
    void Move();
    /*!
    *@brief 座標を取得。
    */
    const CVector3& GetPosition() const
    {
        return m_position;
    }
private:
    tkEngine::CSphereShape    m_sphere;
    CMatrix                   m_wvpMatrix;      //<ワールドビュープロジェクション行列。
    tkEngine::CIDMapModel     m_idMapModel;
    CVector3                  m_position;
    tkEngine::CShadowModel    m_shadowModel;    //!<シャドウマップへの書き込み用のモデル。
    CVector3                  m_moveSpeed;      //!<移動速度。
};

#endif
```

プレイヤーに m_moveSpeed という移動速度を覚えるためのメンバ変数が追加されました。

では、続いて tkPlayer.cpp の Move 関数を下記のように編集してください。

```
void CPlayer::Move()
{
    //XZ平面での移動速度。
    m_moveSpeed.x = 0.02f;
    m_moveSpeed.z = 0.02f;
    if (KeyInput().IsAPress()) {
        //キーボードのAが押されていたら速度を倍にする。
        m_moveSpeed.x *= 2.0f;
        m_moveSpeed.z *= 2.0f;
        m_moveSpeed.y = 0.1f;
    }
    //Y方向への移動速度。
    if (KeyInput().IsUpPress()) {
        m_position.z += m_moveSpeed.z;
    }
    if (KeyInput().IsDownPress()) {
        m_position.z -= m_moveSpeed.z;
    }
    if (KeyInput().IsRightPress()) {
        m_position.x += m_moveSpeed.x;
    }
    if (KeyInput().IsLeftPress()) {
        m_position.x -= m_moveSpeed.x;
    }
    m_position.y += m_moveSpeed.y;
    //重力とかは考えない。
    m_moveSpeed.y -= 0.01f;
}
```

これでプレイヤーは A を押すとジャンプするようになりました。ただし地面と衝突判定などは行っていないため、このプレイヤーは A を押さないと奈落の底に落下していきます。次の Chapter ではプレイヤーが落下しないようにプログラムを変更してみます。

Tips

ゲーム制作において、プレイヤーの挙動はそれっぽく見えれば OK なのでまじめに物理計算を行う必要があるわけではありません。ですが先ほどのジャンプ処理を重力を考慮してプログラムを書いてみましたので、せっかくですのでご紹介します。変更点は

CPlayer の Move 関数のみです。

```
void CPlayer::Move()
{
	//Moveが呼ばれる感覚は16ミリ秒で固定で考える。
	static const float deltaTime = 1.0f / 60.0f;
	//速度の単位をm/sに変更する。
	m_moveSpeed.x = 1.f;			//XZ平面での移動速度。
	m_moveSpeed.z = 1.f;
	if (KeyInput().IsAPress()) {
		//ジャンプ。
		//初速度を2m/sで与える。
		m_moveSpeed.y = 2.0f;
	}
	CVector3 add(0.0f, 0.0f, 0.0f);
	add = m_moveSpeed;
	add.Scale(deltaTime);
	if (KeyInput().IsUpPress()) {
		m_position.z += add.z;
	}
	if (KeyInput().IsDownPress()) {
		m_position.z -= add.z;
	}
	if (KeyInput().IsRightPress()) {
		m_position.x += add.x;
	}
	if (KeyInput().IsLeftPress()) {
		m_position.x -= add.x;
	}
	m_position.y += add.y;
	//速度に重力加速度の影響を与える。
	//重力加速度 9.8m/s^2
	static const CVector3 gravity(0.0f, -9.8f, 0.0f);
	CVector3 addVelocity = gravity;
	addVelocity.Scale(deltaTime);
	m_moveSpeed.y += addVelocity.y;
}
```

変更を加えたプログラムを下記のパスに上げました。Debug モードで実行すると処理が遅いのでもっさりした挙動になるので、Release モードで確認するといいでしょう。

(影も壁に落ちるように改良してます・・・)

¥¥mmnas01¥student¥GC2016¥02_授業¥ゲーム PG Ⅰ¥Lesson02¥Packman_JumpGravity

Chapter 3

地面に立てるようにしてみよう。

Chapter2 でパックマンがジャンプできるようになりましたが、A ボタンを押さないとパックマンは地面を突き抜けて自由落下していたはずです。ではこれを地面に立てるように改造してみましょう。地面の位置は Y 座標で 0 の位置にあるので、パックマンの Y 座標が 0 より小さくなれば落下を行わないようにすれば地面に立てるはずです。ではプログラムを見ていきましょう。

今回編集するソースは下記のファイルになります。

ソースファイル

Packman¥Packman¥game¥Player¥ CPlayer.cpp

Packmap¥Packman¥game¥CCamera.cpp

```
/*!
*@brief  移動処理。
*/
void CPlayer::Move()
{
	//XZ 平面での移動速度。
	m_moveSpeed.x = 0.02f;
	m_moveSpeed.z = 0.02f;
	if (KeyInput().IsAPress()) {
		//キーボードの A が押されていたら速度を倍にする。
		m_moveSpeed.x *= 2.0f;
		m_moveSpeed.z *= 2.0f;
		m_moveSpeed.y = 0.1f;
	}
	//Y 方向への移動速度。
	if (KeyInput().IsUpPress()) {
		m_position.z += m_moveSpeed.z;
	}
	if (KeyInput().IsDownPress()) {
		m_position.z -= m_moveSpeed.z;
	}
	if (KeyInput().IsRightPress()) {
		m_position.x += m_moveSpeed.x;
	}
	if (KeyInput().IsLeftPress()) {
		m_position.x -= m_moveSpeed.x;
	}
	m_position.y += m_moveSpeed.y;
	//重力とかは考えない。
	m_moveSpeed.y -= 0.01f;
	if (m_position.y < 0.0f) {
		//座標が 0 以下になったので座標を補正。
		m_position.y = 0.0f;
	}
}
```

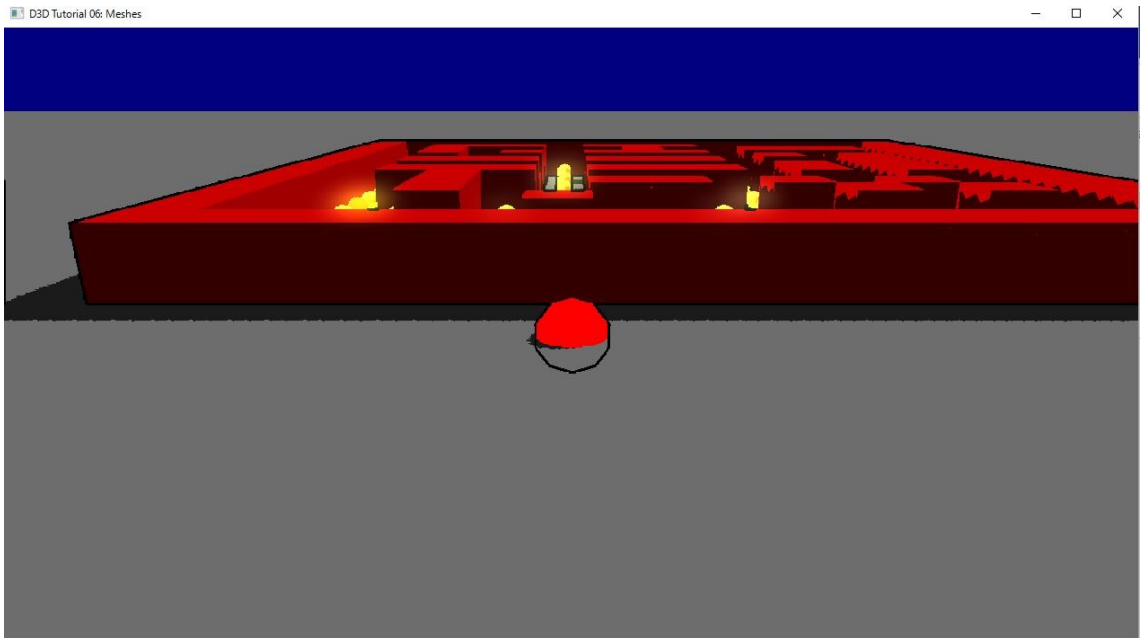
黒字の部分が今回変更した部分になります。

さて、この変更でパックマンは地面をすり抜けて落下しなくなりましたが、実はまだ問題が残っています。カメラのプログラムを下記のように書き換えてみてください。

CCamera.cpp

```
void CGameCamera::Start()
{
	CVector3 cameraTarget;
	m_playerDist.Set(0.0f, 0.5f, -1.5f);
	cameraTarget = m_playerDist;
	cameraTarget.z = 0.0f;
	m_camera.SetPosition(m_playerDist);
	m_camera.SetTarget(CVector3::Zero);
	m_camera.SetUp(CVector3::Up);
	m_camera.SetFar(100000.0f);
	m_camera.SetNear(0.1f);
	m_camera.SetViewAngle(CMath::DegToRad(45.0f));
	m_camera.Update();
}
```

カメラが下記の図のようにプレイヤーの後方に移動したはずです。



実はパックマンの座標は球体の中心を指しています。そのため Y 座標 0 で判定を行うとパックマンが地面にめり込んでしまいます。つまり判定する Y 座標をパックマンの半径分押し上げてやればめり込まなくなるはずです。ではめり込まないようにプログラムを改良し

てみましょう。パックマンの半径は 0.08 とします。

CPlayer.cpp

```
/*!
*@brief  移動処理。
*/
void CPlayer::Move()
{
	//XZ 平面での移動速度。
	m_moveSpeed.x = 0.02f;
	m_moveSpeed.z = 0.02f;
	if (KeyInput().IsAPress()) {
		//キーボードの A が押されていたら速度を倍にする。
		m_moveSpeed.x *= 2.0f;
		m_moveSpeed.z *= 2.0f;
		m_moveSpeed.y = 0.1f;
	}
	//Y 方向への移動速度。
	if (KeyInput().IsUpPress()) {
		m_position.z += m_moveSpeed.z;
	}
	if (KeyInput().IsDownPress()) {
		m_position.z -= m_moveSpeed.z;
	}
	if (KeyInput().IsRightPress()) {
		m_position.x += m_moveSpeed.x;
	}
	if (KeyInput().IsLeftPress()) {
		m_position.x -= m_moveSpeed.x;
	}
	m_position.y += m_moveSpeed.y;
	//重力とかは考えない。
	m_moveSpeed.y -= 0.01f;
	if (m_position.y < 0.08f) {
		//座標が半径以下になったので座標を補正。
		m_position.y = 0.08f;
	}
}
```

いかがでしょうか？パックマンが地面にめり込まなくなったはずです。