第6回　当たり判定3

今回はジャンプと重力について解説します。

アクションの基本動作であるジャンプですが、基本動作なのに解説するのがこんな後半になってしまったのは、ジャンプするときの条件が少々厄介なためです。

ジャンプの条件

無限ジャンプできるゲームを作りたい場合は別ですが、ジャンプ回数を制限する場合は制限するための条件文を書かないといけません。

例えばジャンプ回数を１回に制限したい場合を考えてみます。ジャンプを１回したら地面に着くまでジャンプ出来なくすればいいのです。つまり自機が地面に接地している時のみジャンプできるようにすればジャンプ回数を１回に制限できるのですが、地面に接地しているかどうかというのは、前回やったマップチップとの当たり判定が無いと分かりません。そのためジャンプを解説するのがこんなに後半になってしまったのです。

マップの作成

では条件を説明したのでせっかくなのでジャンプのプログラムを書きましょう・・・といきたいところですが、今の状態でジャンプのプログラムを書いても地面(マップチップ)が少なすぎて、地面に乗れずジャンプが出来ないと思うのでまずはマップを作成します。

マップの描画は最初にやりましたが下の関数で描画しています。

```
void g_tip_draw( void){
    //チップの描画をする関数
    int i;
    int j;
        for( i = 0; i <= TIP_MAX_X; i++){
            for( j = 0; j <= TIP_MAX_Y; j++){
                if( tip[i][j] == 1){
                    //本来ならば、画面内のチップしか描画をする必要はない。
                    //が、面倒なので全部描画させます。

                    DrawGraph( i * TIP_HABA - TIP_HABA / 2, j * TIP_TAKASA –
                    TIP_TAKASA / 2, tip_handle, false);
                    //幅、及び高さの1/2をx、yから引いてるのは、
                    //画像の中心をx、yにするため。
                    }
            }
        }
}
```

関数を見てもらえば分かりますがtip[i][j] == 1の時、マップチップを描画します。なので、tip[0][0]の位置にマップを作成したいならtip[0][0]=1を関数g_loopの中に追加すればいいのです。

今回はジャンプするための地面を作りたいので、画面の一番下の行にマップを作成しましょう。

画面の縦のサイズは480、TIP_TAKASAは32なので、tip[i][j]のjの部分が15にすれば画面の

一番下の行にマップを作成できます。

```
        for( i = 0; i <= TIP_MAX_X; i++){
                tip[i][15] = 1;
        }
```

これをg_loopの中に追加してください。
追加したらちゃんと表示されるか動かしてみましょう。
ちゃんと表示ができたら上以外のマップチップも自分で追加してみてください。

これで準備が整ったので実際にジャンプのプログラムを書いてみましょう。
まずは今回使う下記の２つの変数をgame.cppの先頭に追加します。

```
#define JIKI_JUMP 24
#define JUURYOKU 2
```

変数の意味は名前のままでJIKI_JUMPは自機のジャンプ力、JUURYOKUは重力です。

それでは早速上の変数を使ってジャンプするプログラムを作ります。

```
if( jiki.flag == 1 && (key_b > 0 && key_b < 12)){
                jiki.kasoku_y -= JIKI_JUMP;
        }
        //bボタンが押されるとジャンプする。
        //加速度がy座標のマイナス方向
        //地面に着いていなくてもジャンプができるのは仕様です

        if( jiki.flag == 0) jiki.kasoku_y += JUURYOKU;
        jiki.flag = 0;
        //jiki.flagは地面と接地しているかどうか。で接地している。で接地していない
        //接地していなければ、重力をかける(本当は常に重力があったほうがいいんだけどね)
        //jiki.flag = 1にするところはtipの判定のところで

        if( jiki.sokudo_y > 20) jiki.sokudo_y = 20;
        else if( jiki.sokudo_y < -30) jiki.sokudo_y = -30;
        //y座標は摩擦がないので速度がオーバーしやすいので、あまり上がりすぎないようにリミッター。
```

このプログラムを関数g_jiki_sousa内の摩擦の計算の前の行に追加しといてください。
最初のif文でkey_b＜12となっているのは、bボタンをずっと押し続けた場合、地面に着くとジャンプを永遠に繰り返すのを防ぐためです。
最後のy方向の速度のリミッターはコメントアウトの通り無いと、高いところからジャンプして降りるだけっでもかなりのスピードになります、どのくらいのスピードになるか確かめたい場合は、リミッターのところだけ削除して試してみるのも面白いかもしれません。