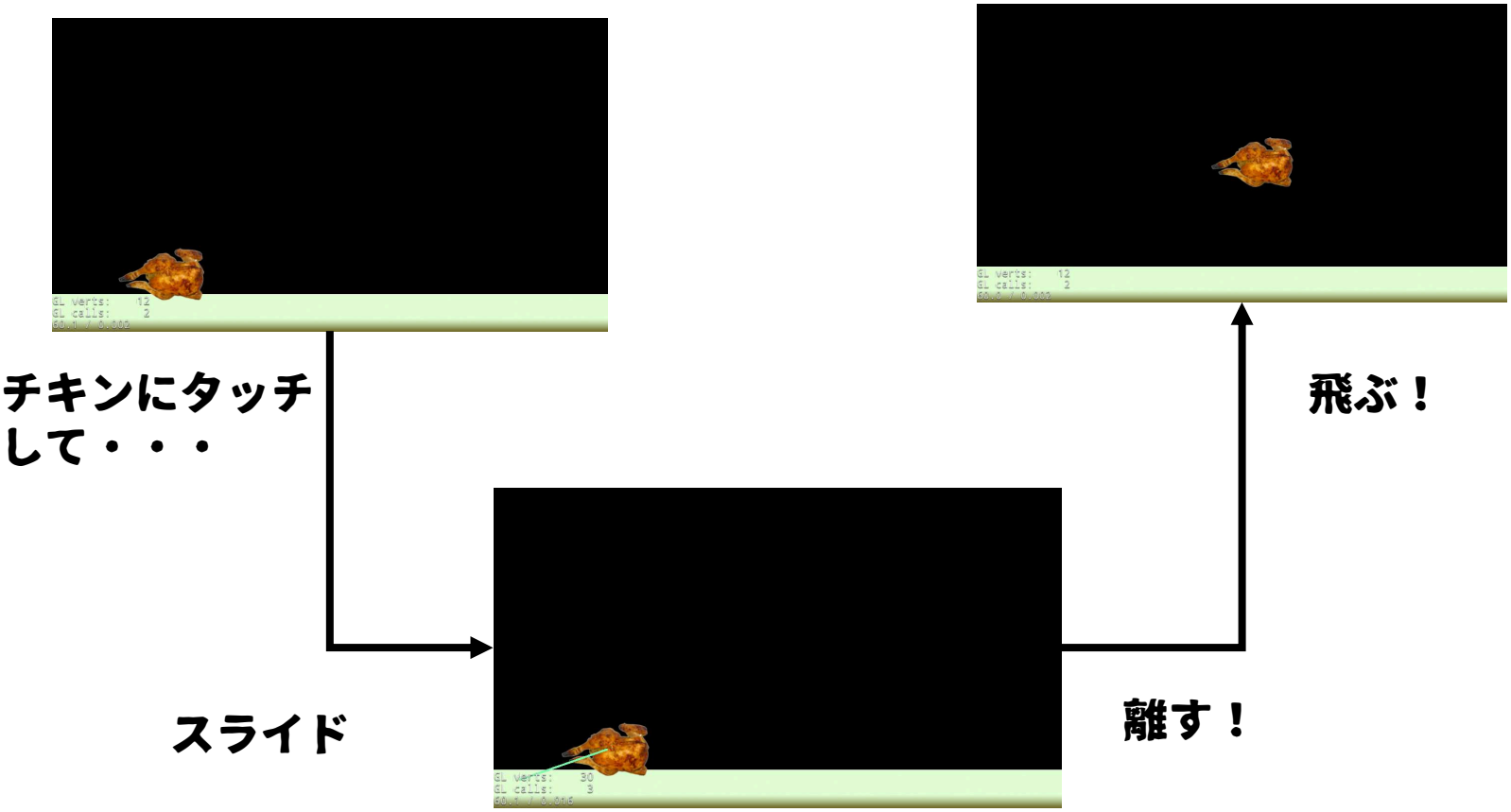
Cocos2d-xで作る物理演算ゲーム
～　スクロールして～編　～

＝　2015年10月08日　＝

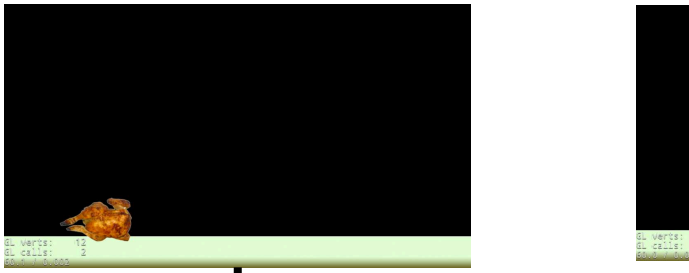
前回のあらすじ

チキンをタップして、引っ張って飛ばすところまで。。



今回はチキンを追跡しながら、
画面をスクロールさせます。

これ



チキンに引っ張って飛ばすと・・・

チキンが中央でホールドされて、画面全体
がスクロールされる。

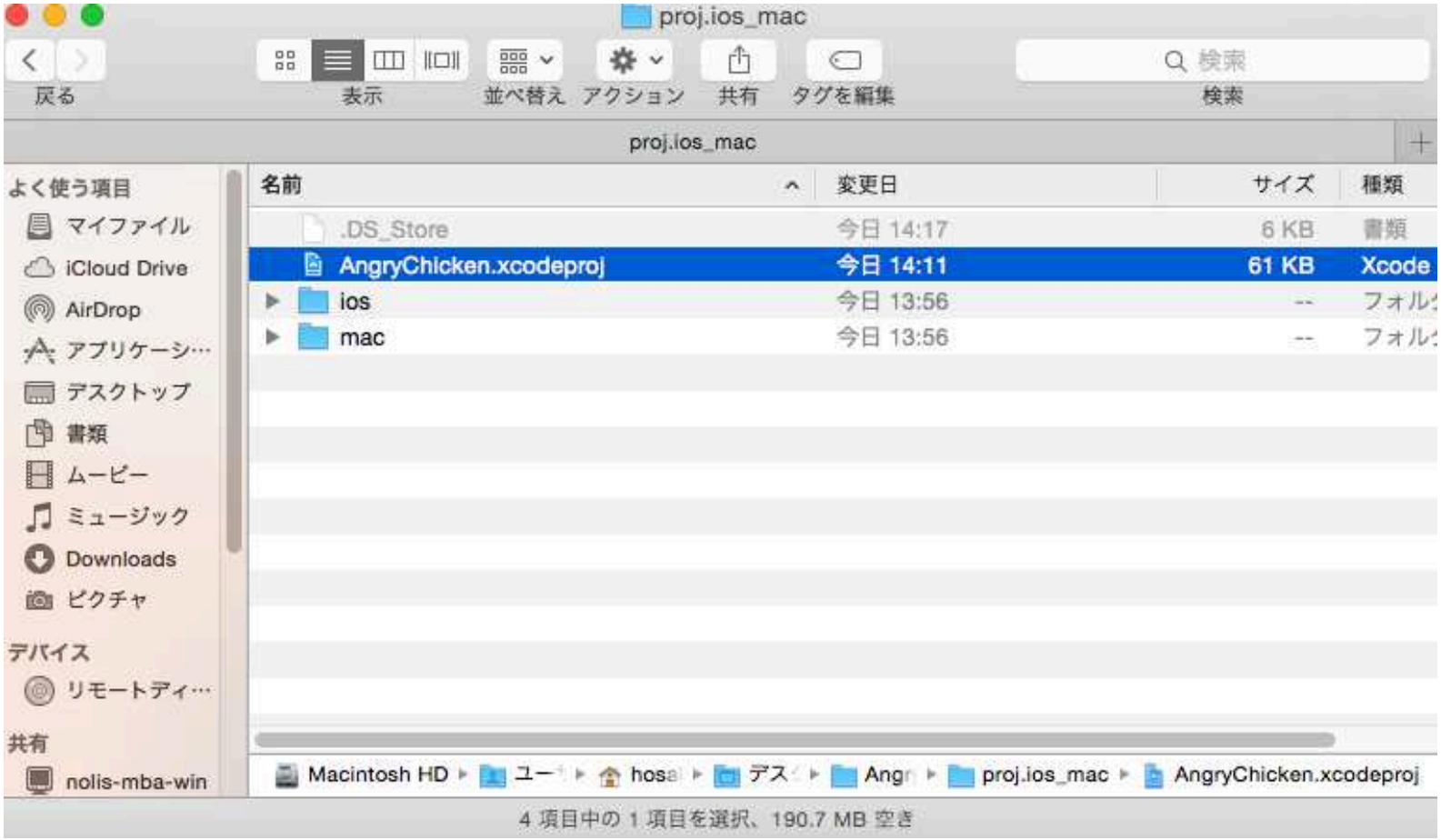
ここまでのソースプログラムはここ

http://monolizm.com/sab/src/AngryChicken.zip

ＧＥＴだぜ！

まずは起動しよう

AngryChickenをxcodeで実行。



結論から言うと・・・

思った以上に簡単で、数行書くだけで終わり。
スクロールだけなら、なんと１行！！

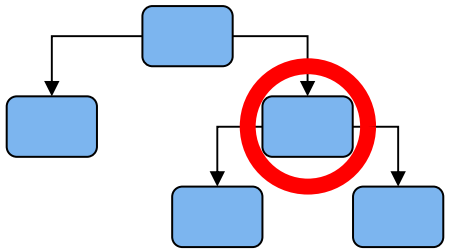
①チキンを追従したスクロール設定

HelloWorld::initメソッドに以下のコードを追加

```
// 登録するリスナー情報を生成
this->runAction(Follow::create(character,Rect(0,0,winSize.width*2.5,winSize.height)));
```

Followクラスは特定のNodeに対して、画面が付いていくというActionです。指定したNodeを、カメラがずっと追尾してくるようなものをイメージしてくれればＯＫ。

＊Nodeとは？
直訳すると節目とか。つまりツリー構造を表した時の節にあたる。
ちなみに一番上のNodeをルートといったりするのだ。



cocos2d-xは描画対象をツリー上で管理する。１画面を構成するクラスをSceneである。Sceneがルートnodeとなる。つまりSceneにaddChildするとSceneの子となったNodeは描画対象となる。

ここまでで実行してみよう！

おお！できた！・・・が、ちょっとこのままではバグがあります。
飛ばしてスクロールされるまでは良いが、その後にチキンをタップして引っ張って飛ばそうとしても、引っ張り線が出てこなく飛ばすことができないのだ！

まず不具合の原因を考えてみる。これは経験値が高ければだいたい予測がつくはず。前回の回で引っ張り線が出る部分処理として、

「タップした位置がチキンだったら、そこから引っ張り線をだす。それ以外のタップだったら何もしない」

という処理だった。チキンをタップした判定が正常に動作していない点が一番可能性が高い。・・・言ってしまうのその通りなのである。

見た目上はチキンをタップしているはずが、プログラム内ではチキンをタップしてないことになっているということである。

スクロール処理を入れる前は、チキンがどうあれ引っ張って飛ばせた。
しかし、処理を入れた後、スクロールしてしまうと、チキンをタップして、引っ張っても出てこないのだ！！

ここで出てくるのは座標系の話

冷静に考察

つまりスクロールしているから、画面上にあるわけで、画面をスクロールさせなかった場合は、画面外に出てしまっていることになる。当たり前だが。つまりチキンは、iPhoneの画面の外側にあるのである。以下のような状態だ。

スクロールしないと画面外

スクロールすると常に画面内

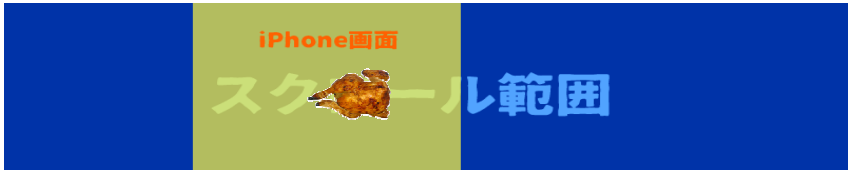
タッチ座標とチキン座標が
どうなっているか考えてみよう

まずはチキン座標を考える

考えやすくするために。まず「スクロールしない方」で考えてみる。

チキンは画面外に出ているのでかなりｘ位置は大きい値になっているはずだ。
このiPhone画面をiPhoen5sとして考える（1136）ここはx=1400位置ぐらいとしておこう。

お次は「スクロールする方」で画面で考えてみる。



スクロールする方で考えてみても、チキンの位置は画面で見えているだけで、実はx=1400の位置にある。
つまりカメラが追従しているというイメージだ。

まだピンとこない人は現実世界で置き換えてみよう。カメラを横切るようにボールを投げる。当然カメラを動かさなければ、ボールが出現して、数秒後画面内から消えるだろう。しかしカメラがボールをとらえるように動かしたらボールは映ったままだ。当たり前のことである。

そしてここからが重要。その場合ボールの位置はカメラを動かそうが、動かなかろうが世界の位置（言ってしまえば緯度経度？）は変わらないのである。

つまりチキンはスクロールして画面に映っていようが、スクロールなしで画面外に行こうがゲーム内の世界では、x=1400なのである。

次はタッチ座標を考える

まず「スクロールしない方」で考えてみる。

画面中心をタップしたことにしよう。iPhone5sはサイズは640x1153だから、x=320，y=570くらいか？
馬鹿にしているかといわれそうだが、当然画面には、チキンはいないので引っ張れるわけがない。

スクロールする方で画面で考えてみる。

チキンは一見タップしているように見える。が、前述したようにチキンをタップしたという判定にならない。
画面中心をタップしている。結論を言おう、タッチ位置はx=320，y=570なのである。

さてここでチキンの座標を思い出してほしい。チキンはx=1400であり、タッチはx=320だ。
いかがだろう？全然違うのである。判定がおかしくなるのは無理もない。

何が起こっているのか？

タッチ座標とチキン座標は
属する世界（原点の位置）
が違うのである。

結論を言うとこうなる

ゲーム世界。つまりチキンが住む世界の原点（つまりx=0，y=0）

iPhone画面世界。タッチ位置を求める世界の原点（つまりx=0,y=0）

じゃぁどうすればよいか？

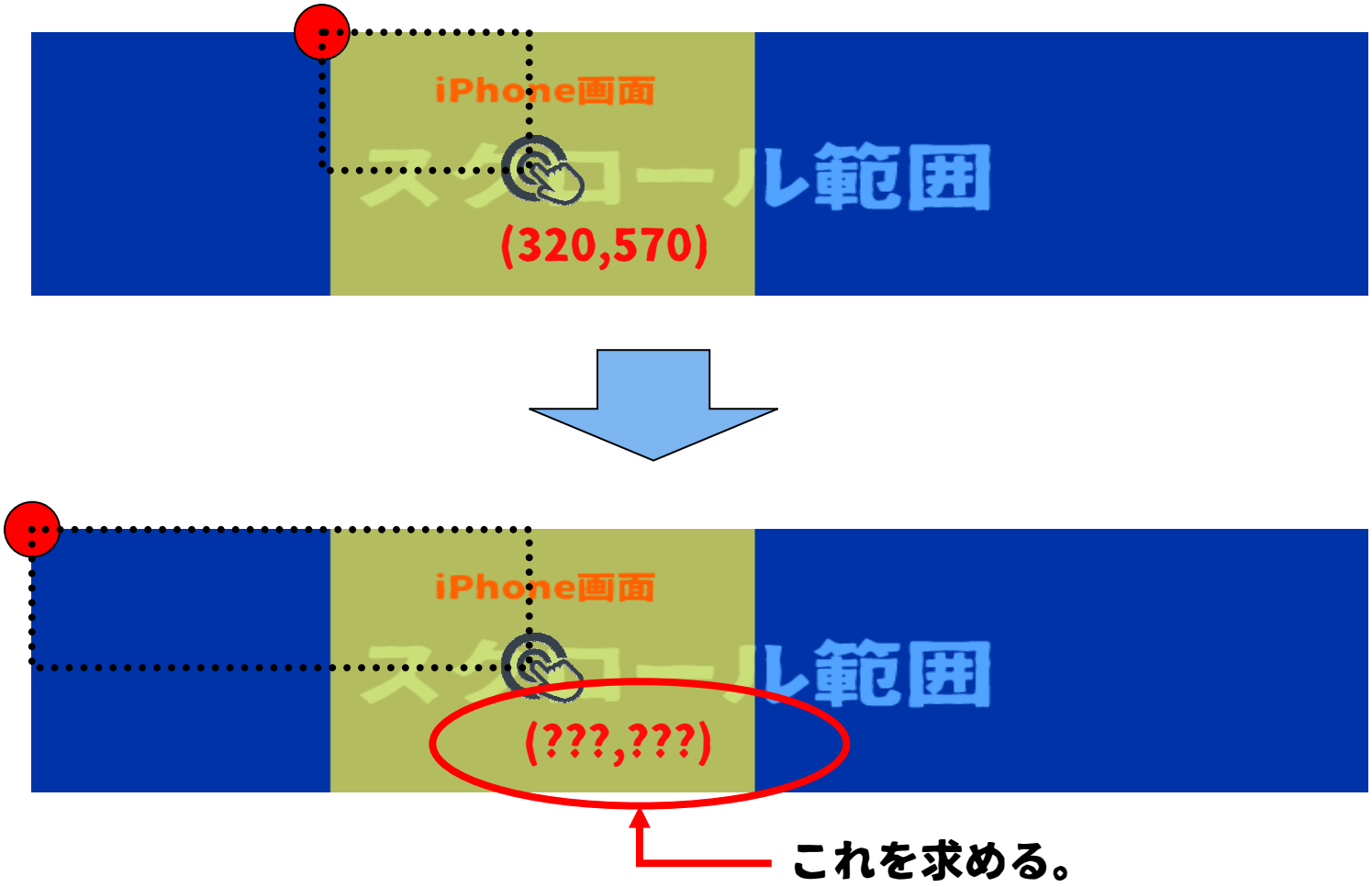
同じ世界にしてやればよい。

つまりどちらか一方の座標をもう片方の世界に引っ越しさせればよい。

以下の２通りの考え方がある。

１．タッチ座標をチキン座標系と同じ世界の座標に変換する
２、チキン座標をタッチ座標系と同じ世界の座標に変換する

今回は「１．」。つまりタッチ座標をチキンのいる座標系に変換するのだ。
（座標変換といいます）



HelloWorld::_onTouchBeganを以下のように赤文字部分を追加

```
bool HelloWorld::_onTouchBegan(Touch* touch, Event* event)
{
    // キャラクターとタッチ位置のあたり判定
    auto* charSprite = (Sprite*)this->getChildByTag(CHAR_OBJTAG);
    auto charSpriteRect = COMMON_HELPER::getRect(charSprite);
    auto tPos = touch->getLocation();
    tPos = this->convertToNodeSpace(tPos);

    log("pos x=%f, y=%f", tPos.x, tPos.y);
    if ( charSpriteRect.containsPoint(tPos) == true )
    {// チキンをタッチした場合は、ひっぱりラインを生成

    ～～～以下、省略～～
```

HelloWorld::_onTouchMovedを以下のように赤文字部分を追加

```
// タッチイベントのドラッグ
void HelloWorld::_onTouchMoved(Touch* touch, Event* event)
{
    // 更新用に退避
    _currentTouchPoint = touch->getLocation();
    _currentTouchPoint = this->convertToNodeSpace(_currentTouchPoint);
}
```

HelloWorld::__onTouchEndedを以下のように赤文字部分を追加

```
// タッチイベントの終了　08/27
void HelloWorld::_onTouchEnded(Touch* touch, Event* event)
{
    auto* charSprite = (Sprite*)this->getChildByTag(CHAR_OBJTAG);
    auto charPos = charSprite->getPosition();

    //画像位置とタッチ終了から、ベクトルの強さを求める。
    auto endPos = touch->getLocation();

    endPos = this->convertToNodeSpace(endPos);

    auto force = Vect(charPos.x - endPos.x, charPos.y - endPos.y) * 4;

    // キャラクターに力を加える。
    charSprite->getPhysicsBody()->applyImpulse(force);

    // 加えたら線を削除
    this->removeChildByTag(PULL_LINE_OBJTAG);
}
```

実行してみよう！

チキンに引っ張って飛ばすと・・・

チキンが中央でホールドされて、画面全体がスクロールされる。ちゃんと飛んでる最中でも引っ張って飛ばせるぞ！

次回は物理演算Chipmunk
ひとまずここで体裁を整えよう編

ここまでのソースプログラムはここ

http://monolizm.com/sab/src/AngryChicken9.zip

ご清聴ありがとうございました。