

# なでなでプロジェクトV2



取り扱い説明書



# \* 目次 \*

## 1. Inputモジュール導入

- 1-1. スクリプトのコピー
- 1-2. カーソル画像のコピー
- 【補足】カーソルの定義方法

## 2. NADEv2モジュール導入

- 2-1. スクリプトのコピー
- 2-2. 必要画像のコピー
- 【おまけ】その他のモジュール

## 3. 立ち絵と領域設定

- 3-1. 立ち絵ファイルの保存
- 3-2. 領域設定ファイル作成

## 4. スクリプトの設定

- 4-1. 領域色を定義
- 4-2. マウスカーソルの定義
- 4-3. 細かい設定
- 4-4. 立ち絵とスイッチの設定

## 5. イベントの作成

- 5-1. 初期化处理
- 5-2. キャラクターの反応イベント作成

## 6. イベント内容の作成

- 6-1. 最低限の処理
- 6-2. キャラクターの反応を作りこみ

## 7. マウスジェスチャ

- 7-1. マウスジェスチャとは。
- 7-2. ジェスチャで出来ること。
- 7-3. イベントの実装その1
- 7-4. 条件分岐について



# 1. Inputモジュール導入

図1. Inputモジュール

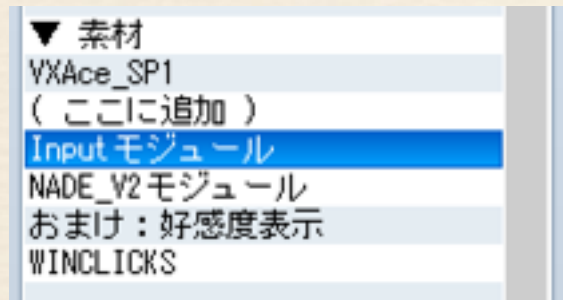


図2. コピーする画像

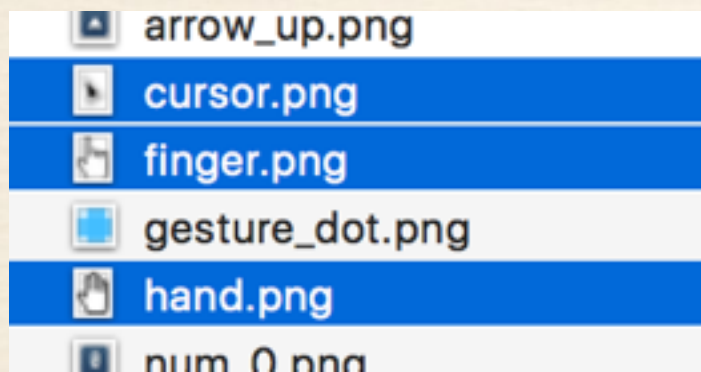


図3. 設定項目

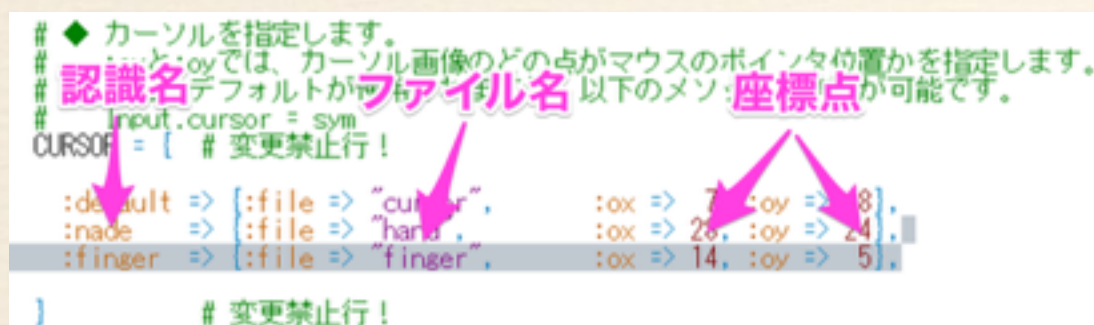
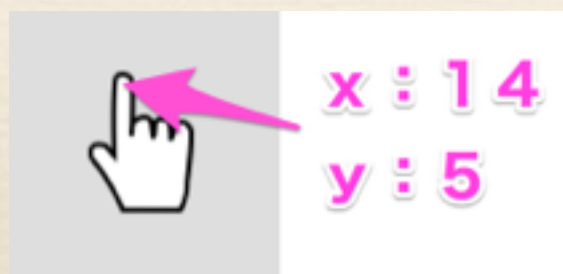


図4. 座標点の設定例



## 1-1. スクリプトのコピー

サンプルプロジェクトのスクリプトから「Inputモジュール」をコピーして、自分のプロジェクトへ貼り付けて下さい。

## 1-2. カーソル画像のコピー

サンプルプロジェクトの「Graphics/system」にあるカーソル画像を自分のプロジェクトにコピーして下さい。

デフォルトで登録してある画像は

「cursor」  
「hand」  
「finger」

の3つとなります。

触りの定義によって他にもカーソルが欲しい場合は別途、スクリプトの「設定項目」で新しいカーソルを定義して下さい。

## ※カーソルの定義方法

「:finger」の追加方法を例に説明します。

- I. 「:finger」を認識名として新しい行を追加。
- II. カーソル画像を「Graphics/system」に保存。
- III. ファイル名を設定項目の「:file」に記述。
- IV. マウスの座標と認識させる1点を「:ox」と「:oy」に記述。



# 2. NADEv2モジュール導入

図5. NADEv2モジュール

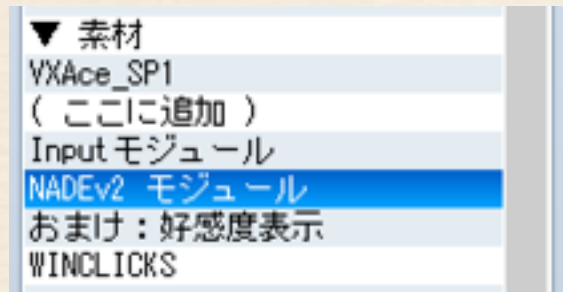


図6. コピーする画像

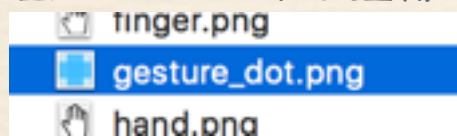


図7. 好感度ウィンドウ

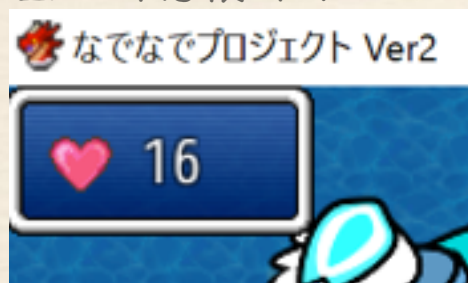


図8. ウィンドウのマウス操作



## 2-1. スクリプトのコピー

サンプルプロジェクトのスクリプトから「NADEv2モジュール」をコピーして、自分のプロジェクトへ貼り付けて下さい。  
※このモジュールは「Inputモジュール」よりも下に設置して下さい。

## 2-2. 必要画像のコピー

「Graphics/System」に保存されている画像「gesture\_dot.png」を自分のプロジェクトにコピーして下さい。

## ☆その他のモジュール

### ・「おまけ：好感度表示」

ゲーム変数に保存されている数値を1つだけウィンドウに表示する機能です。好感度表示等にお使い下さい。  
※不要であれば導入する必要はありません。

### ・「B:WINCLICKS」

ゲームのウィンドウ各種をマウス操作できるようにするためのモジュールです。タイトル画面、セーブ画面、メニュー画面、メッセージの入力待ちなどをカーソルやクリックで操作することができます。主にデフォースクリプトや拡張ウィンドウにしか対応しておりませんのでご了承下さい。  
※不良であれば導入する必要はありません。  
※なお、このモジュールはB版です。



# 3. 立ち絵と領域設定

図9. Inputモジュール






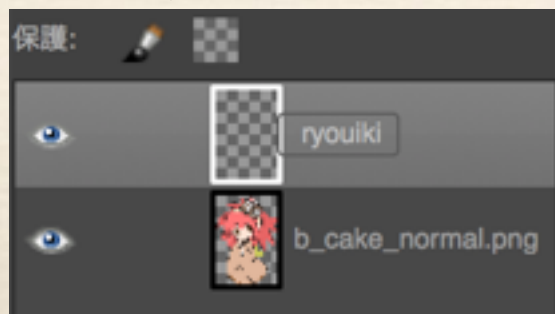
名前	変更日
 b_cake_normal.png	2016年1月15日 11:56
 b_cake_anger.png	2016年1月15日 12:04
 b_cake_happy.png	2016年1月15日 12:13
 b_cake_cry.png	2016年1月15日 12:17
 b_cake_anger_max.png	2016年1月15日 12:24

図10. 立ち絵ファイルを開く



図11. 新しいレイヤー作成



## 3-1. 立ち絵ファイルの保存

「Graphics/Pictures」フォルダに立ち絵に使用するピクチャを保存して下さい。

## 3-2. 領域設定ファイル作成

「なでなで」や「つんつん」等にマウスが反応する領域を設定するファイルを作成します。

領域設定ファイルの作成にはレイヤー機能と塗りの色をRGBそれぞれ「0~255」で設定できるお絵かきソフトが必要です。

※今回は「GIMP」を使用して説明します。

### 3-2-1. 立ち絵ファイルを開く

説明不要かもしれませんが、とりあえずお絵かきソフトで立ち絵を開いて下さい。

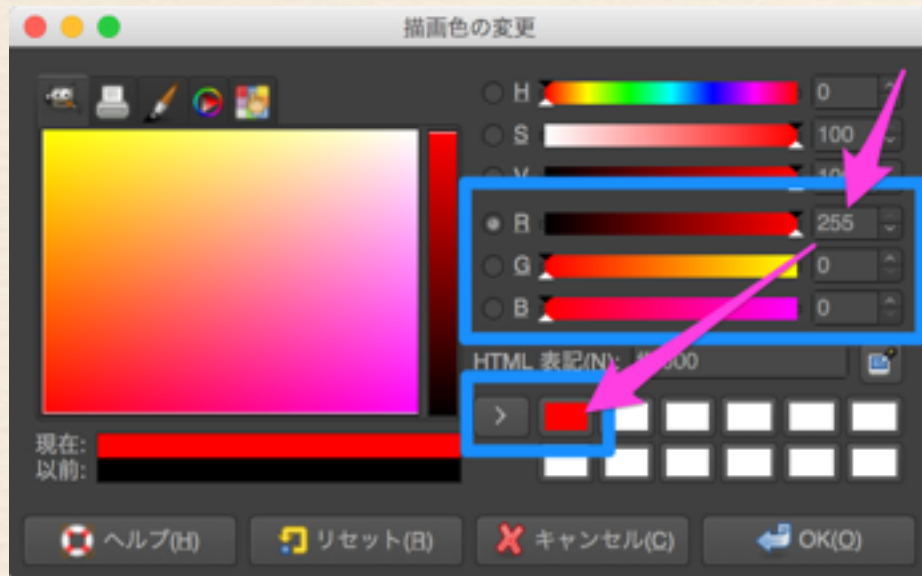
### 3-2-2. 新しいレイヤーを作成

立ち絵レイヤーの上に、同じサイズの透明レイヤーを1つ作成して下さい。



# 3. 立ち絵と領域設定

図12. 色の作成



## 3-2-3. 領域用の色を作成

領域設定用に色を作成します。RGBそれぞれ「0~255」で設定して下さい。今回はとりあえず「255, 0, 0」で作成します。

※色は作成したい領域の数だけ作成します。例えば、「頭、ほっぺ、口、鼻、髪の毛、首、手、足」のように設定したい場合は8色の作成が必要です。  
ただし、ほっぺや手、足のように左右で2箇所設定したい場合は、色をわけずに同じ色を使用します。

## 3-2-4. 反応して欲しいエリアを塗る

マウスに反応してほしい領域を塗ります。この塗った範囲にゲームでカーソルを合わせると、マウスカーソルの画像が変化して、そこを撫でたり出来ることが分かります。

図13. 反応して欲しい領域を塗る



図14. マウスカーソル画像の変化



## 【注意】

なでモジュールでは、RGBの値が正確に一致しないと領域を認識しません。

従って、ブラシ等のアンチエイリアスが効くツールでは境界がぼやけてしまってうまく動作しなくなります。鉛筆ツールや矩形ツール等の境界まで同色で塗れるツールを使用しての塗り分けをお願いします。



# 3. 立ち絵と領域設定

図15. 他の領域も塗る

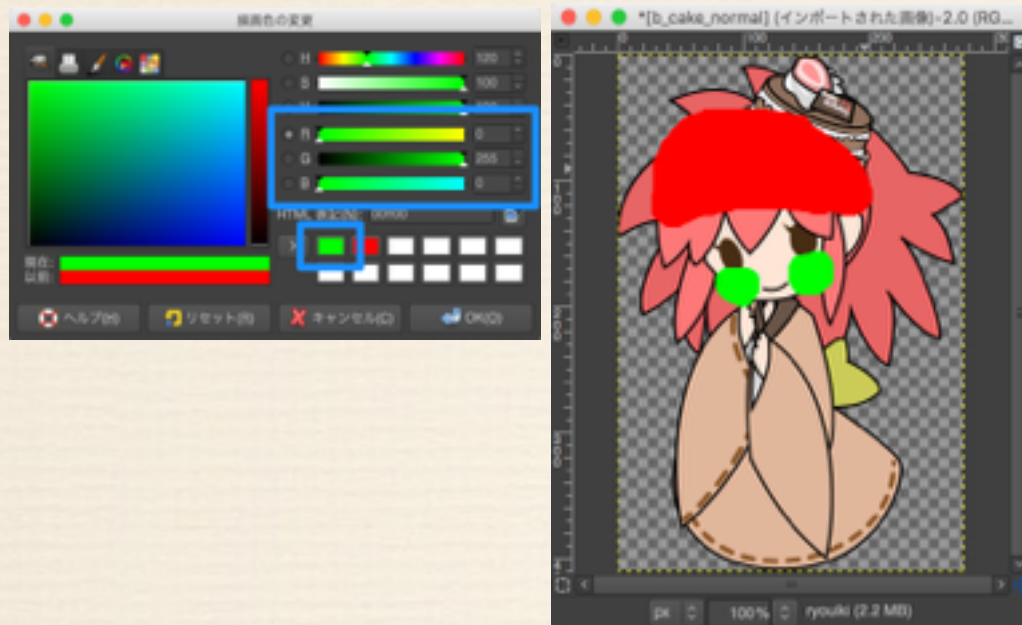
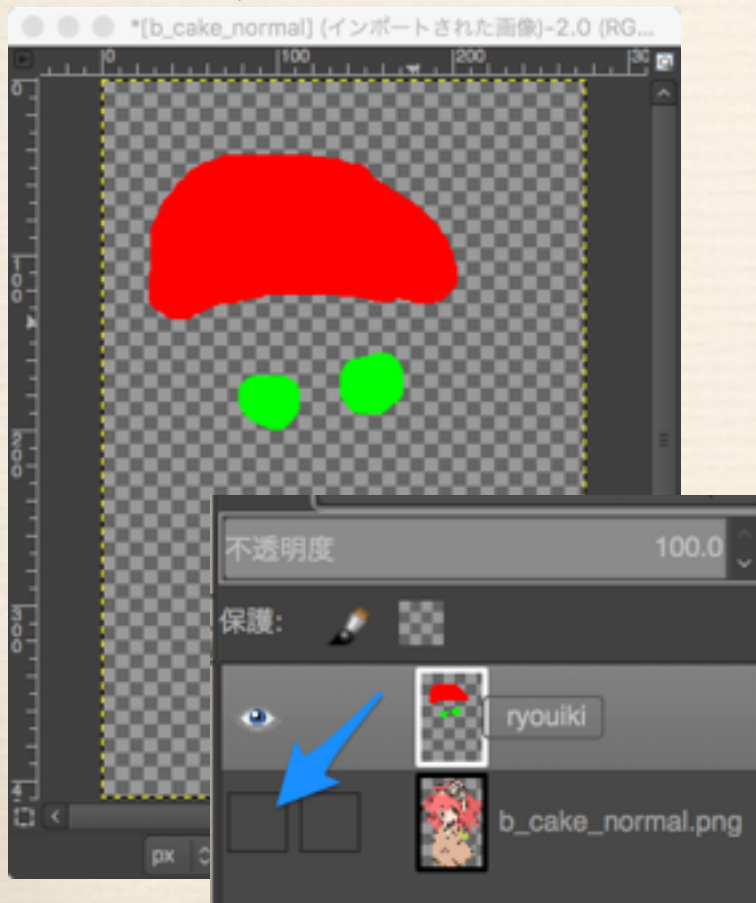


図16. 立ち絵レイヤーを非表示



## 3-2-5. 他の領域もどんどん作成

反応させたい領域分だけ色を作成し、どんどん塗っていきます。図15のほっぺのように同じ名前の領域が複数ある場合は、同じ色を使って塗って下さい。

今回は頭とほっぺの2箇所のみ作成します。

## 3-2-6. 領域レイヤーだけを保存

まず、立ち絵レイヤーを非表示にします。これで領域を塗り分けたレイヤーだけが表示されるはずです。

その状態で新しい名前で画像を保存して下さい。

※ファイルは「Graphics/Pictures」に保存して下さい。

※保存するファイルは立ち絵のファイル名に続けて「\_map」という名前で保存して下さい。サンプルで言えば

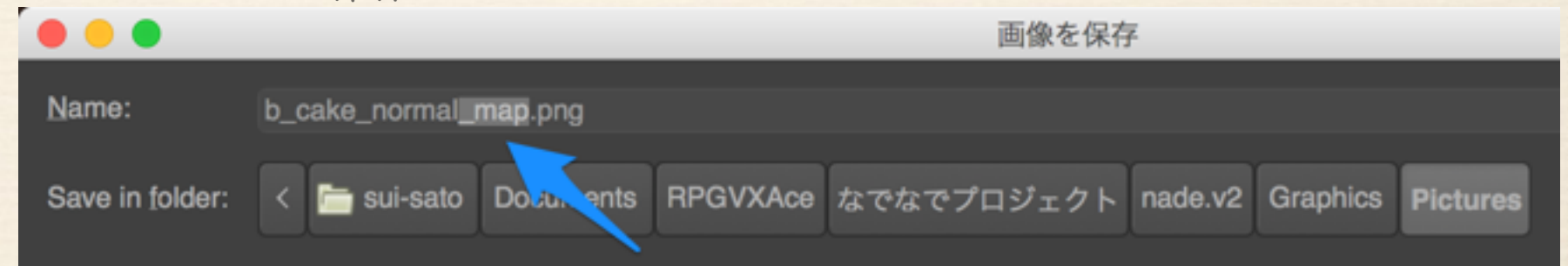
「b\_cake\_normal\_map.png」

となります。

## 【注意】

ファイルの種類はBMP形式やPNG形式で保存して下さい。GIF形式やJPEG形式では色情報が変わる場合があります。

図17. ファイルに保存





# 4. スクリプトの設定

図18. COLORSの設定

```
# 触り反応領域と色の設定を指定してください。  
# 赤・緑・青・アルファ値の順で 0~255 の値を設定します。  
# アルファ値については設定できないツールの場合は 255 として下さい。  
COLORS = [  
    #(RRR, GGG, BBB, AAA),  
    :head => Color.new(255, 0, 0, 255),  
    :cheek => Color.new(0, 255, 0, 255),  
]
```

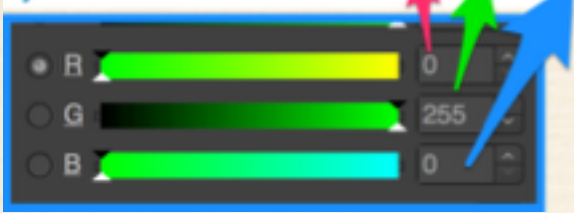


図19. CURSORの設定

```
# 触り反応領域で使用するマウスカーソルを指定してください。  
# ここで指定する値は、Inputの設定項目「CURSOR」で作成しておく必要があります。  
# 指定されていない領域では「:nade」が使用されます。  
CURSOR = [  
    :head => :nade,  
    :cheek => :finger,  
]
```



図20. 細かい設定

```
# なで反応で、どれだけの間隔で続ければ反応するか時間 (フレーム数)  
NADE_COUNT = 90  
  
# マウスが反応領域から外れてもカウントをリセットしない許容時間 (フレーム数)  
RESET_COUNT = 30  
  
# ダブルクリック判定のクリック感覚 (フレーム数)  
DC_SPAN = 60  
  
# マウスジェスチャの有効/無効設定 (trueで有効)  
GESTURE_ENABLE = true  
  
# マウスジェスチャの1パスあたりの移動量 (pixel数)  
GESTURE_MOVE = 15  
  
# マウスジェスチャの際マウスの軌道を画面上に描画するか設定 (trueで有効)  
GESTURE_ORBIT = true
```

## 4-1. 領域色を定義

領域設定ファイルで使った領域色を「COLORS」に定義します。書式は以下のとおり。

**:領域名 => Color.new(赤, 緑, 青, 透明度)**

今回は赤 (255, 0, 0,) と緑 (0, 255, 0) を設定しています。透明度については特に設定しなければ「255」にしておいて下さい。

※領域名は触りに反応してイベントが自動実行される際に条件分岐で使います。「head」や「cheek」等分かりやすいものを付けましょう。

## 4-2. マウスカーソルの定義

マウスカーソルが各領域の上に移動してきた時に使用するカーソル画像を「CURSOR」に定義します。書式は以下のとおり。

**:領域名 => :マウスカーソル認識名**

※領域名は 4-1 で定義した名前を使用します。

※マウスカーソル認識名は 1-2 で定義した名前を使用します。

※ここで定義しなかった領域名は、デフォルトで「:nade」カーソルが適用されます。

## 4-3. 細かい設定

メインの設定は次ページに譲って、先に細かい部分を 6 項目設定していきます。

スクリプト内の説明に従って、なでモジュールの細かい挙動を設定して下さい。

なお、マウスジェスチャについては別の項で説明します。



# 4. スクリプトの設定

図21. SWITCHESの設定

```
# 触り判定でONにするゲームスイッチIDを指定してください。
# この指定は画像のファイル名に対して指定します。
# このスイッチ番号でどのキャラが触られているかを判断してください。
# ※ このスイッチで並列処理or自動実行のイベントを作成するとOK。
#   ゲームスイッチ15番の自動実行イベントは、○○キャラのなでなで反応等。
#
# :nsw ... なでなで (マウスボタンを押さずにカーソルを一定以上動かす)
# :csw ... つんつん (ダブルクリックする)
# :lsw ... ぐりぐり (左ドラッグで一定以上動かす)
# :gsw ... マウスジェスチャ反応
SWITCHES = [
# ++++++ 1キャラ目 ++++++
# ++++++
"b_cake_normal" => [ :nsw => 15, :csw => 16, :lsw => 17, :gsw => 18 ],
"b_cake_anger"  => [ :nsw => 15, :csw => 16, :lsw => 17, :gsw => 18 ],
"b_cake_happy"  => [ :nsw => 15, :csw => 16, :lsw => 17, :gsw => 18 ],
"b_cake_cry"    => [ :nsw => 15, :csw => 16, :lsw => 17, :gsw => 18 ],
# ++++++ 2キャラ目 ++++++
# ++++++
"b_cake_normal2" => [ :nsw => 21, :csw => 22, :lsw => 23, :gsw => 24 ],
]
```

図22. 自動実行イベントの設定例

基本設定	名前:	トリガー:	条件スイッチ:
	キャラ1@なでなで	自動実行	0015:Chara_1 NadeNade ...

## 4-4. 立ち絵とスイッチの設定

なでなで等のマウス反応に使用する立ち絵を「SWITCHES」に定義します。書式は以下のとおり。

**"画像ファイル名" => {スイッチの設定}**

スイッチの設定は4項目を必要に応じて設定します。とある立ち絵ではこの機能は使いたくないという場合にはそれ以外の項目のみ設定して下さい。各項目の意味は以下のとおり。

:nsw...触り領域の中でカーソルを一定以上動かした場合に反応してスイッチをONにします。

:csw...触り領域をダブルクリックした場合に反応してスイッチをONにします。

:lsw...触り領域の中でカーソルを一定以上左ドラッグした場合に反応してスイッチをONにします。

:gsw...触り領域を始点にマウスジェスチャを実行した場合に反応してスイッチをONにします。

※マウスジェスチャについては別項で説明します。

画面に表示した立ち絵はここでの設定に応じてマウスの操作に反応してゲームスイッチを自動的にONにします。立ち絵の挙動（会話等）はONになったゲームスイッチで自動実行されるイベントを作成して作りこんで下さい。

また、複数のキャラクターを登場させたい場合はキャラクターごとにONにするスイッチを変えてあげることで、マウス反応の挙動を別の自動実行イベントに分けることができますのでうまく設定してみてください。（サンプルでは2キャラ目を1つだけ設定してあります。）



# 5. イベントの作成

図23. 初期化处理

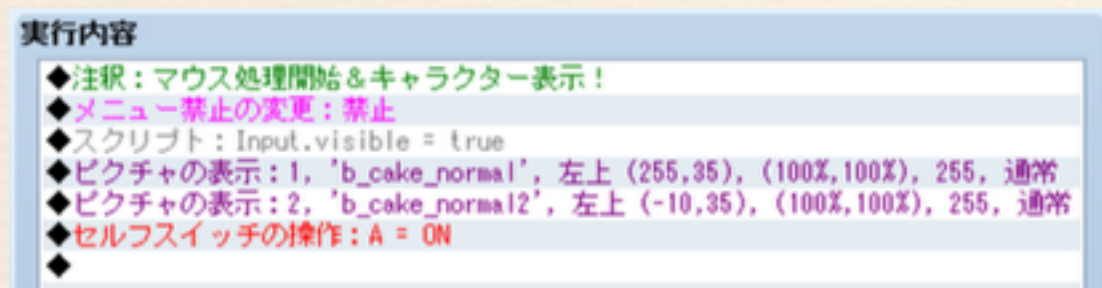


図24. イベントの作成

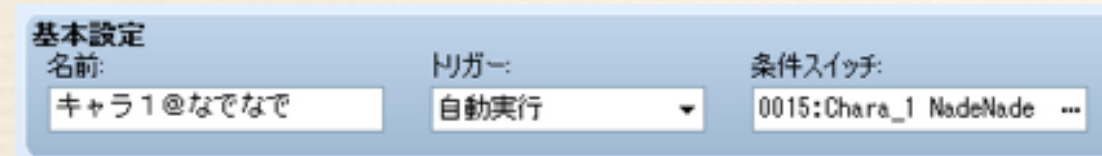


図25. 他の触り反応も作成



## 5-1. 初期化处理

まずはマウスに反応してもらうために準備をします。この処理は自動実行イベント等で1度だけ実行してください。

処理の内容は以下の2点。他の機能は必要に応じて初期化して下さい。

- ・マウスの表示
- ・立ち絵用ピクチャの表示

※ピクチャのズームには対応していません。100%で！

## 5-2. キャラクターの反応イベント作成

マウス操作に対するキャラクターの反応は自動実行イベントで作成します。図24のようにトリガーを「自動実行」に設定して下さい。

そして、出現条件のスイッチをスクリプトの「SWITCHES」で定義したIDに設定して下さい。

これで、それぞれのマウス操作に応じて各イベントが自動実行されるようになります。

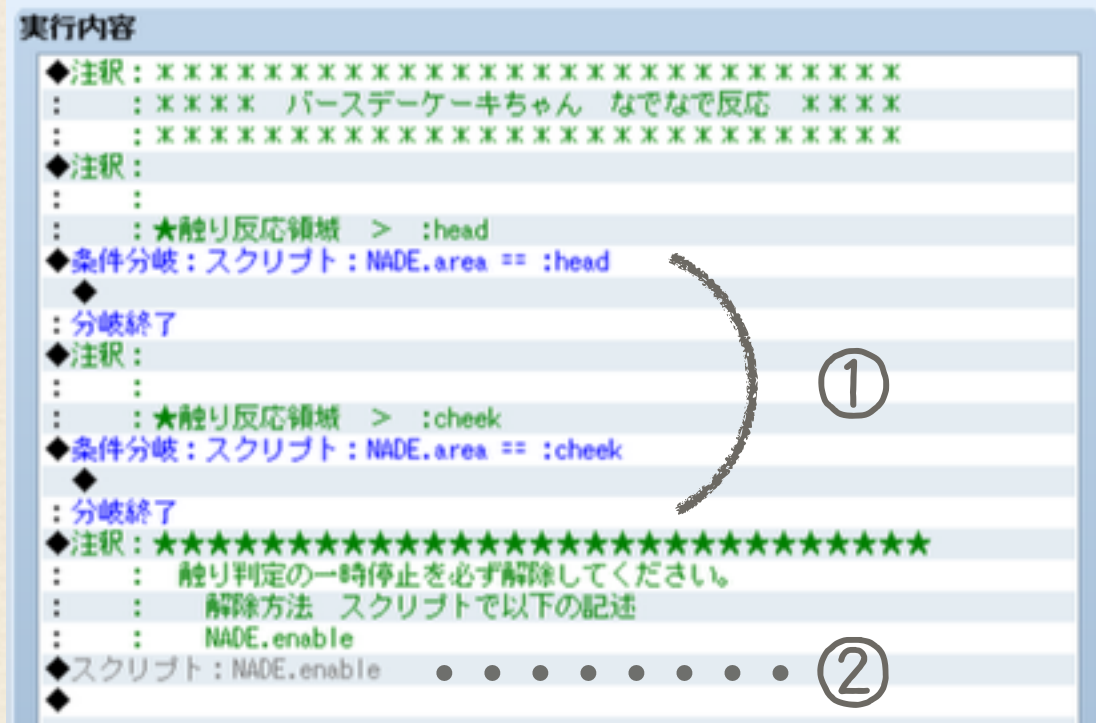
同じように他の触り反応や、複数のキャラクターを使う場合はそれぞれのキャラクターに必要なだけのイベントを作成しましょう。

イベントの内容作成については次のページで説明します。



## 6. イベント内容の作成

### 図26. 最低限の処理



## 図27. キャラクターの反応を作成



## 6-1. 最低限の処理

キャラクターの反応は図26の構成が最低限の処理になります。特に②については処理を忘れると次から立ち絵がマウスに反応しなくなりますので注意してください。この構成をテンプレートとしてコピーして使いまわすと間違いがないでしょう。

①条件分岐…立ち絵がマウスに反応すると「NADE.area」という変数に「COLORS」で設定した領域名の内、反応した名前を保存します。この値を「==」で比較して各部位固有のキャラクター反応を作成して下さい。

この条件分岐は「COLORS」で設定した分だけ作成して下さい。

②スクリプト…立ち絵がマウスに反応すると、イベント実行中に別のマウス操作が発生しないように一時的にマウス処理を無効化してあります。キャラクターの反応が終わったら最後にマウス処理を有効化するためのスクリプトを実行して下さい。

## 6-2. キャラクターの反応を作りこみ

6-1の①で作成した条件分岐の中に個々のキャラクター反応を作りこんでいきます。図27では立ち絵の変更と一言喋るだけの処理しかしていませんが、内部でさらに条件分岐を組み合わせて好感度やランダムな会話をつくり込んだり、メニューを起動したりと自由に作成してみてください。

サンプルプロジェクトのイベントではもう少し細かく作りこんでありますので、参考にしてみてください。



# 7. マウスジェスチャ

図28. マウスジェスチャ



## 7-1. マウスジェスチャとは。

「右ボタンを押しながらカーソルを上下左右に操作し、最後にボタンを離す」この一連の動作によって何かしらのイベントを起動する機能です。

起動されるイベントは、「右に動かした」「上に動かした」「下に動かした後、右に動かした」など、マウスカーソルを動かした方向の組み合わせによって変化します。

図28の例では「下→右」の順に動かした場合のジェスチャにヒントトークのイベントが起動するように設定あります。

## 7-2. ジェスチャで出来ること。

ジェスチャの機能を利用することで、手の動きを直接トレースしてゲームに伝えることが可能になります。

表1にも一例をあげましたが、例えば「肩から上に動かせば」突き飛ばす、逆に「肩から下に動かせば」抱きしめるといったアクションも作ることができるようになります。

また、表1では2パスまでしか定義していませんが「左→上→右→下→左」のように5パスで定義すれば「円」のように動かした時に反応します。「上→下→上→下」のように同じ上下でも4パス分動かせば「揺さぶる」として扱うこともできます。

あまり複雑な組み合わせはユーザーさんが混乱してしまいますが、ユーザーさんの手の動きに合わせてキャラクターの反応を演出したいときにご利用ください。

表1. サンプルプロジェクトの割当

動き	コマンド
左	手を近づける
右	ビンタ
上	強く押す
下	引き寄せる
上→下	乱暴に撫でる
下→上	めくる
下→右	ヒントトーク



## 7. マウスジェスチャ

## 図29. ①自動実行イベントの設定

**基本設定**

名前:	トリガー:	条件スイッチ:
キャラ1 @ ジェスチャ	自動実行 ▼	0018:Chara_1 Gesture ...

図30. ②ジェスチャの分岐

```
◆条件分岐：スクリプト：NADE.gesture == "L"
◆コモンイベント：[キャラ1@[L]]
◆
：分岐終了
◆条件分岐：スクリプト：NADE.gesture == "R"
◆コモンイベント：[キャラ1@[R]]
◆
：分岐終了
◆条件分岐：スクリプト：NADE.gesture == "UD"
◆コモンイベント：[キャラ1@[UD]]
◆
：分岐終了
◆条件分岐：スクリプト：NADE.gesture == "DU"
◆コモンイベント：[キャラ1@[DU]]
◆
：分岐終了
◆条件分岐：スクリプト：NADE.gesture == "DR"
◆コモンイベント：[キャラ1@[DR]]
◆
```

図31. ③呼び出し先でキャラクターを作りこみ

```

実行内容
◆注釈：*****
：      ：**   パースデーケーキちゃん   マウスジェスチャ反応   **
：      ：**   コマンド：ヒントトーク   **
：      ：*****
◆注釈：
：      ：
：      ：★触り反応領域   >   このコマンドでは触り判定領域は気にしない。
：      ：                        どこでやってもヒントをおしゃべりするよ。
◆変数の操作：[0006:乱数] = 乱数 ( 0..10 )
◆条件分岐：変数 [0006:乱数] == 1
◆ピクチャの表示：1, 'b_cake_anger', 左上 (255,35), (100%,100%), 255, 通常
◆文章：-, -, 通常, 下
：      ：触る場所によっては特定の反応しないところがあるよ。
：      ：なでなででは反応しないけど、つんつんは反応するみたいだね。
◆ラベルジャンプ：!!FINISH
◆
：分岐終了

```

### 7-3. イベントの実装その 1

- ①まず、スクリプトの「SWITCHES」の「:gsw」で設定したスイッチIDで起動する自動実行イベントを作成して下さい。
- ②イベントの実行内容では、条件分岐を使用してマウスの動きに合わせたコモンイベントを呼び出して下さい。
- ③呼び出したコモンイベント側でキャラクターの反応や、メニューの呼び出し、特別なコマンド等を組み込んで下さい。

## 7-4. 条件分岐について

スクリプトがジェスチャの操作を検出すると、カーソルの動きを記憶するようになっていきます。

その動きは「NADE.gesture」という変数に保存され、その値を条件分岐コマンドのスクリプトで比較することで、カーソルの動きに合わせたイベントを呼び出せるようになっています。

では、「NADE. gesture」にはどんな値が記憶されているかと言うと、「上下左右」の4方向がそれぞれ「UDLR」の文字として記憶されています。更に、「下に動かした後、右に動かした」といった2パス以上の動きは「DR」のように動かした数だけ文字が繋がって記憶されます。7-2の例で上げたような円の動きであれば「LURDL」のように判断すれば良いわけですね。