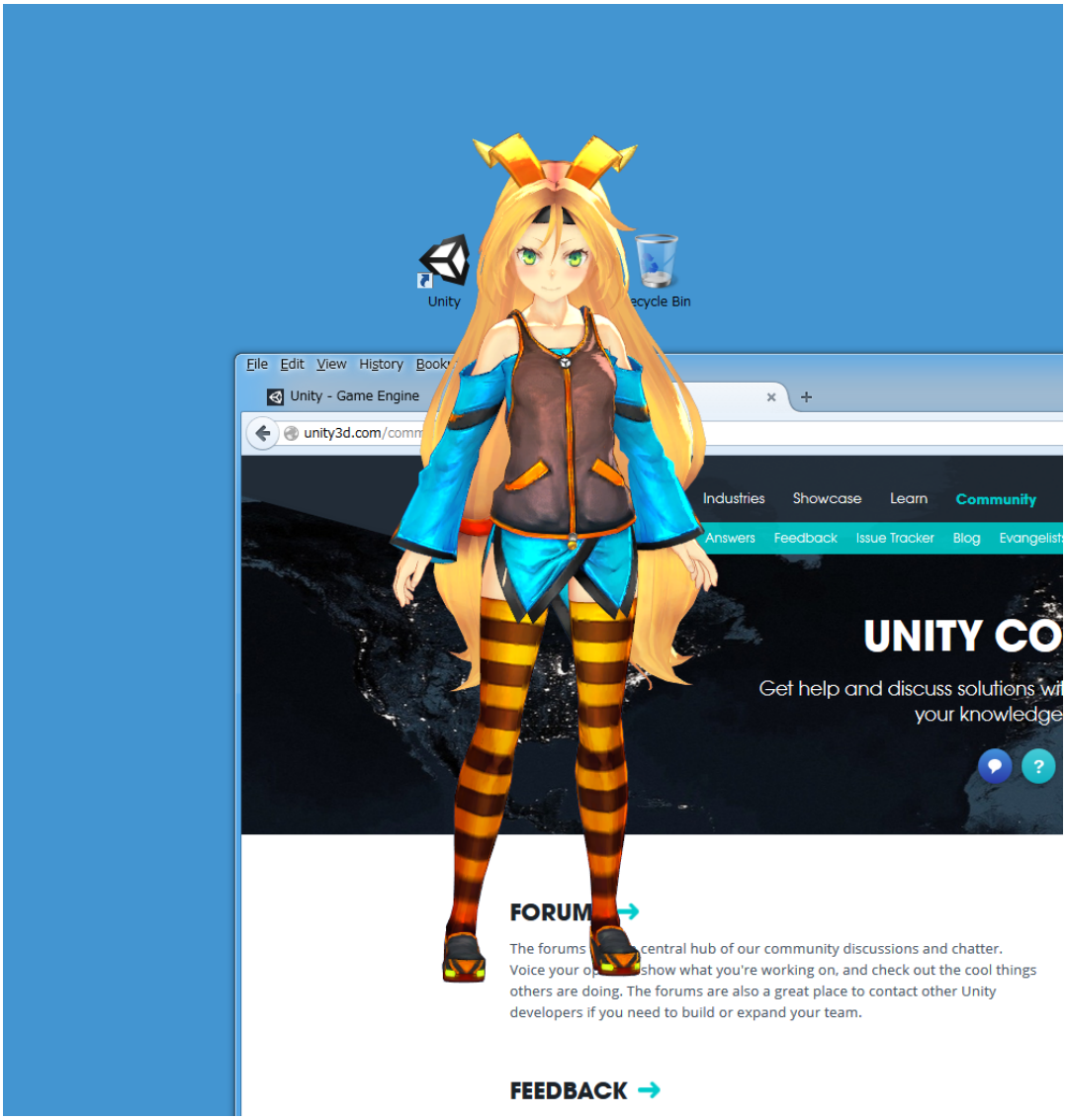
Desktop Mascot Maker for Unity5

リファレンスマニュアル

Version 1.7.0 2015/05/25 by PanzerSoft



Desktop Mascot Maker は Unity でデスクトップマスコットを作るツールです。

Unity のカメラに映るものなら 3D でも 2D でも何でも
簡単にデスクトップマスコットを作ることができます。

注意事項

このアセットを利用する前に以下の点にご注意下さい。

- **ビルドのターゲットとなるプラットフォームは Windows だけとなります**

アセットの特徴

- **Unity5 の場合、Personal Edtion、Professional Edition のどちらでも利用できます。どちらの Edition で利用してもアセットの性能に違いはありません**
- **デスクトップマスコットを Unity から自由自在に操ることができます**
- **マスコットキャラクターの透明度を変更できます**
- **Unity のメインウィンドウを消したり、半透明にすることができます**
- **uGUI の Image と Text が利用できます**
- **MascotMakerMulti コンポーネントを使えば、複数のデスクトップマスコットを表示できます NEW!**

Contacts

ご不明な点などありましたら以下へお問い合わせ下さい。

http://www.assetstore.unity3d.com/en/#!/publisher/9484

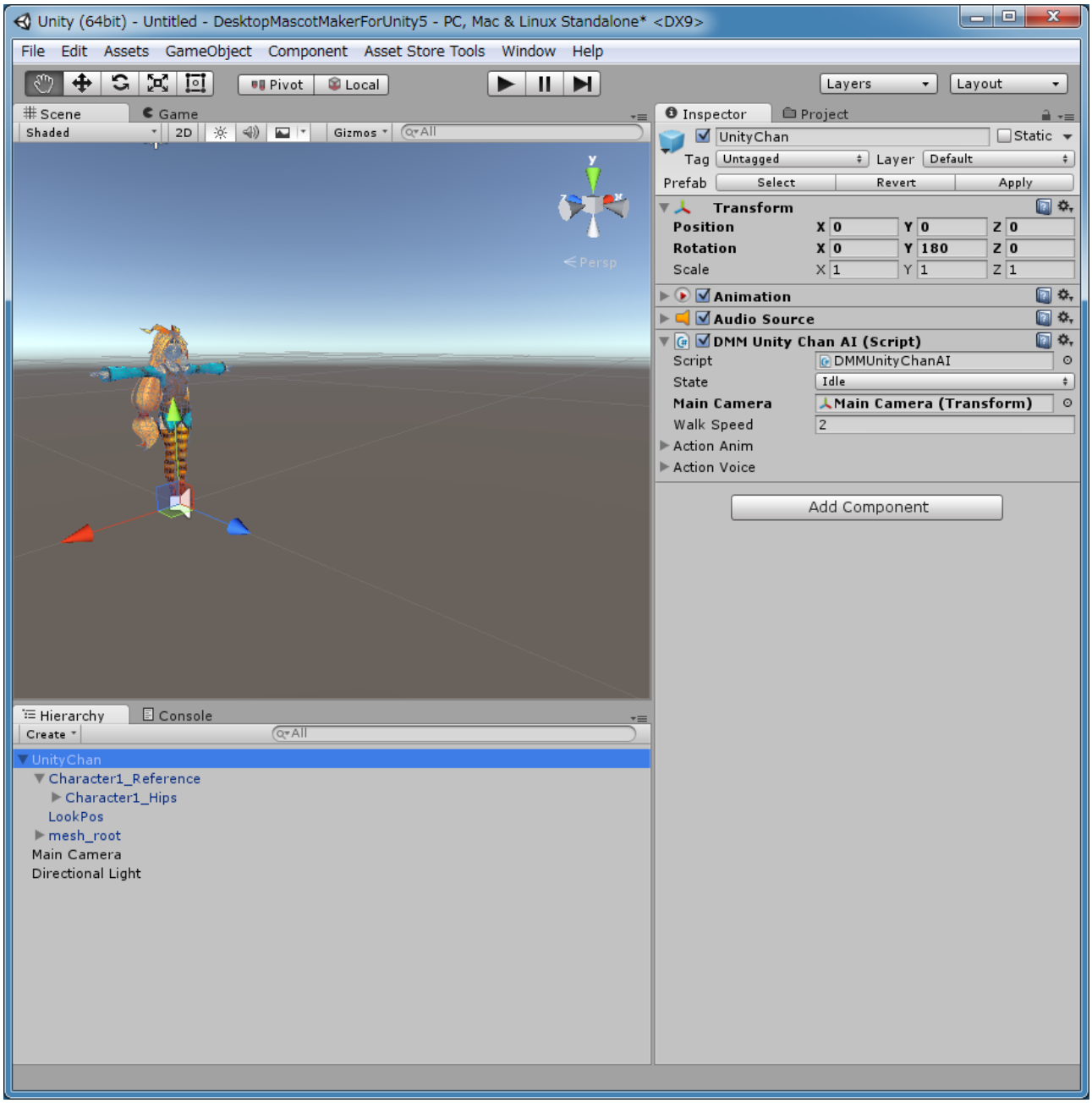
http://panzersoft.web.fc2.com/assetstore_jp.html

Table of Contents

クイックスタート

マスコットをシーンに置いて下さい

新しいシーンを作成し、そこにあなたのマスコットを置いて下さい。
ディレクショナルライトの明るさも調整しましょう。

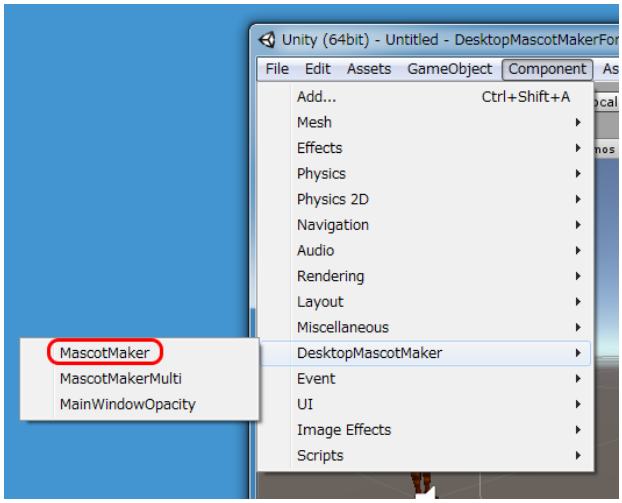


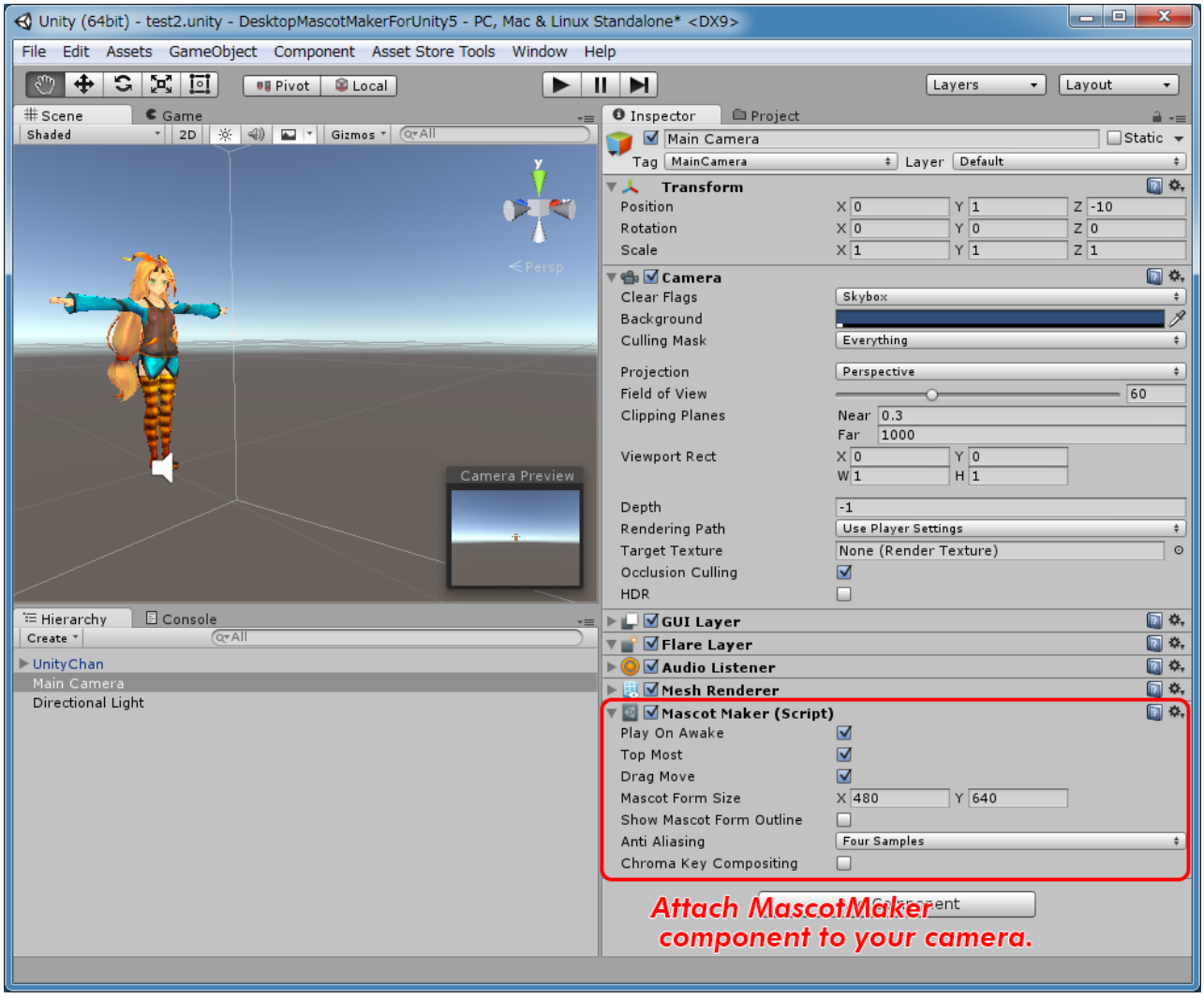
カメラに **MascotMaker** コンポーネントをアタッチして下さい

シーン内のメインカメラに MascotMaker コンポーネントをアタッチして下さい。ここではメインカメラにアタッチしていますが、カメラであればメインカメラでなくても構いません。

MascotMaker コンポーネントは、メニューの

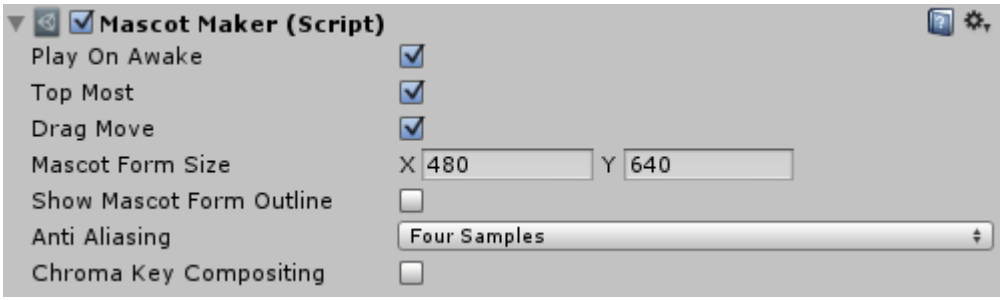
Menu > Component > DesktopMascotMaker > MascotMaker からアタッチできます。





MascotMaker component settings

MascotMaker コンポーネントの設定項目について説明します。



Play On Awake (bool value)

True に設定すると，マスコットは起動と同時に表示されます。

Top Most (bool value)

True に設定すると，マスコットは常に最前面に表示されます。

Drag Move (bool value)

True に設定すると、マスコットを左クリック＆ドラッグで移動できるようになります。

Mascot Form Size (Vector2 value)

Mascot Form のサイズを変更することができます。
（Mascot Form については後述します。MascotMaker クラスリファレンスを参照して下さい）
（Mascot Form のサイズは、プログラム実行中にも変更できます。ただし、重い処理なので Update 関数内でフレーム毎にサイズ変更するような処理はおすすめしません）

Show Mascot Form Outline (bool value)

このチェックボックスはデバッグ専用です。True に設定すると、Mascot Form のアウトラインが表示され、Mascot Form の大きさが確認できます。

Anti Aliasing (enum value)

アンチエイジングのレベルを指定します。サンプル数が多いほど画質が良くなりますが、処理が少し重くなります。

Chroma Key Compositing

True に設定すると、クロマキーモードで描画します（クロマキーモードについては後述します。Chroma Key Compositing のセクションを参照して下さい）

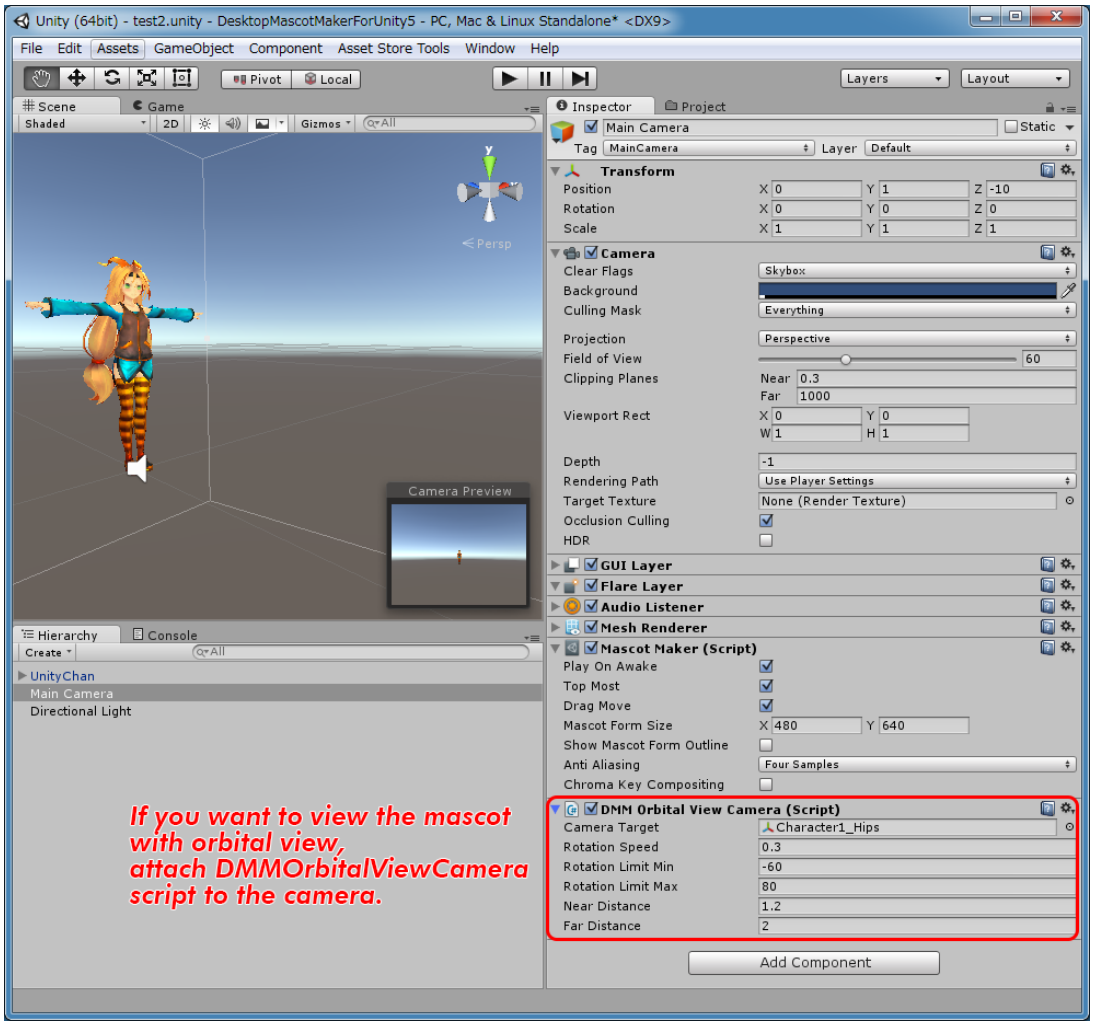
これらの設定項目は、スクリプトからの変更も可能です。

```
bool MascotMaker.Instance.PlayOnAwake           // bool value
bool MascotMaker.Instance.TopMost               // bool value
bool MascotMaker.Instance.DragMove              // bool value
Vector2 MascotMaker.Instance.MascotFormSize     // Vector2 value
bool MascotMaker.Instance.ShowMascotFormOutline  // bool value
MascotMaker.AntiAliasingType MascotMaker.Instance.AntiAliasing // enum value
bool MascotMaker.Instance.ChromaKeyCompositing   // bool value
```

カメラにカメラスクリプトをアタッチして下さい（オプション）

もし、マスコットの周囲ををまわるカメラでマスコットを見たい場合、DMMOrbitalViewCamera.cs スクリプトをカメラにアタッチして下さい。そして、スクリプトの Camera Target にゲームオブジェクトをアサインして下さい（この例ではユニティちゃんの Character1_Hips をアサインしています）。このようにすることで、カメラはユニティちゃんの周囲を回転することができるようになります。

この DMMOrbitalViewCamera.cs スクリプトはオプションです。このスクリプトがなくても Desktop Mascot Maker は利用できます。カメラをマスコットの周辺で回転させたい時だけ使って下さい。



DMM Orbital View Camera の設定項目

Camera Target : カメラが注視するターゲットを指定します。

Rotation Speed : カメラの回転速度を指定します。

Rotation Limit Min : 垂直方向の回転の最小値を指定します。

Rotation Limit Max : 垂直方向の回転の最大値を指定します。

Min Size : Orthographic カメラの場合、カメラの Size の最小値を指定します。

Max Size : Orthographic カメラの場合、カメラの Size の最大値を指定します。

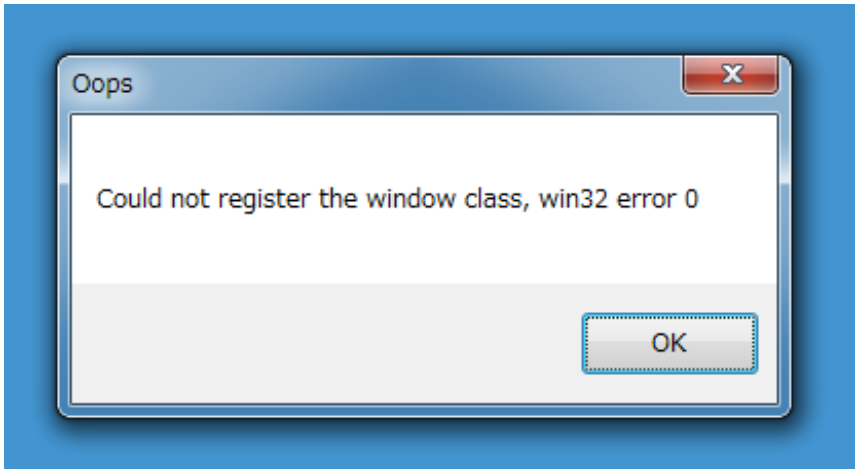
Near Distance : Perspective カメラの場合、カメラとターゲットの間の距離の最小値を指定します。

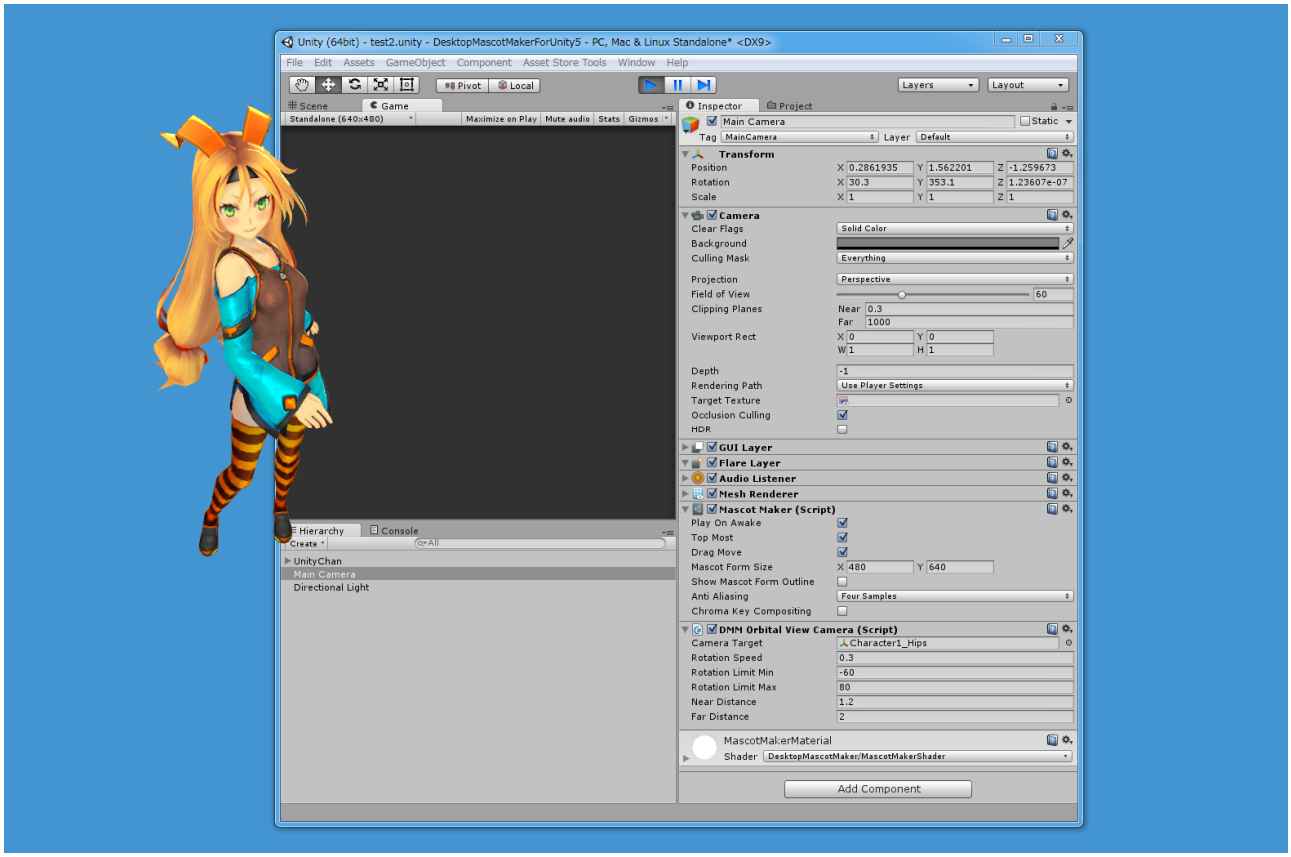
Far Distance : Persipective カメラの場合、カメラとターゲットの間の距離の最大値を指定します。

以上です !!

プレイボタンを押して、マスコットをチェックして下さい。

(注意: もし以下の様なメッセージボックスが出た場合 OK を押して下さい。このメッセージボックスがなぜ出るかは、巻末のよくある質問とその答えをご参照下さい)





更に高度な使い方について

Assets フォルダ内の Assets/DesktopMascotMaker/Examples フォルダを参照して下さい。デモシーンがあります。

マスコットとインタラクションする方法を知りたい場合、以下の３つのデモシーン Demo01_uGUI.unity、Demo02_Event.unity、Demo03_Physics.unity を参照して下さい。
Demo01_General.unity は Unity からマスコットとインタラクションする方法をデモします。
Demo02_Event.unity は Desktop Mascot Maker で利用できるイベント処理についてデモします。
Demo03_Physics.unity は Unity で発生した物理挙動をマスコットに反映する方法についてデモします。

uGUI の Image、Label を使いたい場合、デモシーン Demo04_uGUI.unity を参照して下さい。
Demo04_uGUI.unity は Desktop Mascot Maker での uGUI の使い方についてデモします。

もし Unity のメインウィンドウを非表示にしたい場合は、デモシーン Demo05_HideMainWindow.unity を参照して下さい。
Demo05_HideMainWindow.unity は Unity のメインウィンドウを消したり、半透明にしたりする方法についてデモします。

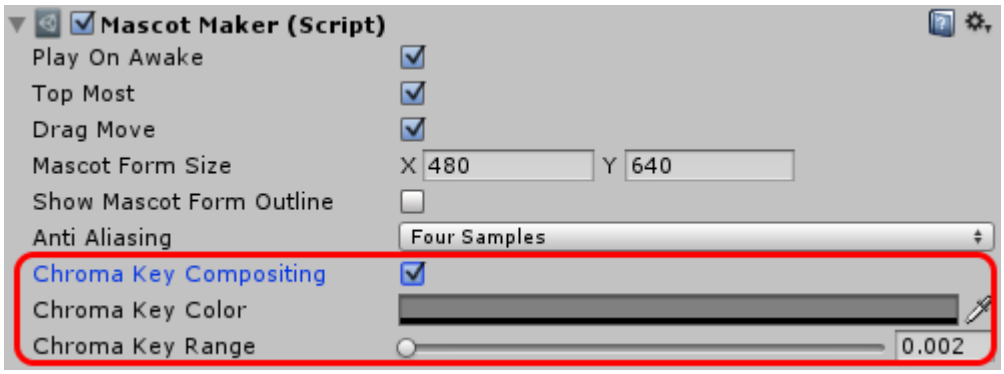
もし一度に複数のマスコットを表示したい場合、デモシーン Demo06_MultipleMascots.unity を参照して下さい。
Demo06_MultipleMascots.unity は MascotMakerMulti コンポーネントを使って、一度に複数のマスコットを表示する方法についてデモします。

Chroma Key Compositing（クロマキー合成）

マスコットに透過シェーダを利用してる場合、マスコットのレンダリングが乱れることがあります。
そのようなときは、Chroma Key Compositing をオンにして下さい。
透過シェーダを利用しない場合は、Chroma Key Compositing をオフにすることをおすすめします。

Chroma Key Compositing Off

Chroma Key Compositing On



Chroma Key Compositing Settings

Chroma Key Compositing (bool value)

このチェックボックスを True にすると, Chroma Key Compositing（クロマキー合成）モードがオンになります。

Chroma Key Color (Color value)

クロマキー合成の背景色です。この色が.透明色になります。

Chroma Key Range (float value)

クロマキー合成の強度です。 設定可能範囲は [0.002 - 0.5] です。

これらの設定項目は、スクリプトからの変更も可能です。

```
bool MascotMaker.Instance.ChromaKeyCompositing  // bool value
Color MascotMaker.Instance.ChromaKeyColor       // Color
float MascotMaker.Instance.ChromaKeyRange       // float value [0.002-0.5]
```

複数のデスクトップマスコットを表示する方法

Desktop Mascot Maker には２つの主要なコンポーネントがあります。１つは MascotMaker コンポーネントで、もう１つは MascotMakerMulti コンポーネントです。

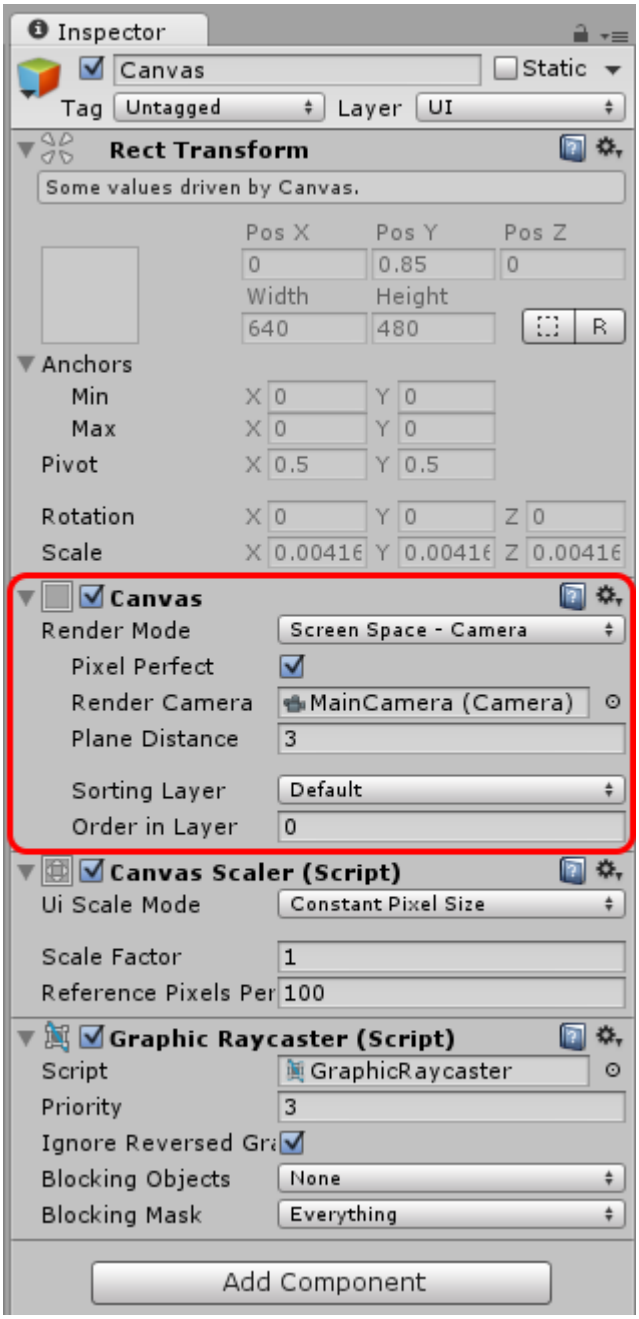
MascotMaker コンポーネントは、シングルトン デザイン パターンで実装されています。ですので、MascotMaker コンポーネントは、１つのシーンに１つまでしか置くことはできません。（２つ以上のコンポーネントをシーン上に置こうとすると、１つを除いて他のコンポーネントは削除されます）

一方で、MascotMakerMulti コンポーネントは、通常の MonoBehaviour コンポーネントとして実装されています。そのため、１つのシーンに複数の MascotMakerMulti コンポーネントを置くことができます。もし、あなたが１つのシーンで複数のマスコットを表示したい場合は、MascotMakerMulti コンポーネントを利用して下さい。

複数のマスコットを MascotMakerMulti で表示する詳しい方法については、MascotMakerMulti クラスリファレンスまたはデモシーン Demo06_MultipleMascots.unity を参照して下さい。

uGUI の利用方法

uGUI の Image/Text を Desktop Mascot Maker で利用するためには、uGUI の Canvas の RenderMode を「ScreenSpace - Camera」にして、Canvas の Render Camera にマスコットを表示するためのカメラを設定して下さい。



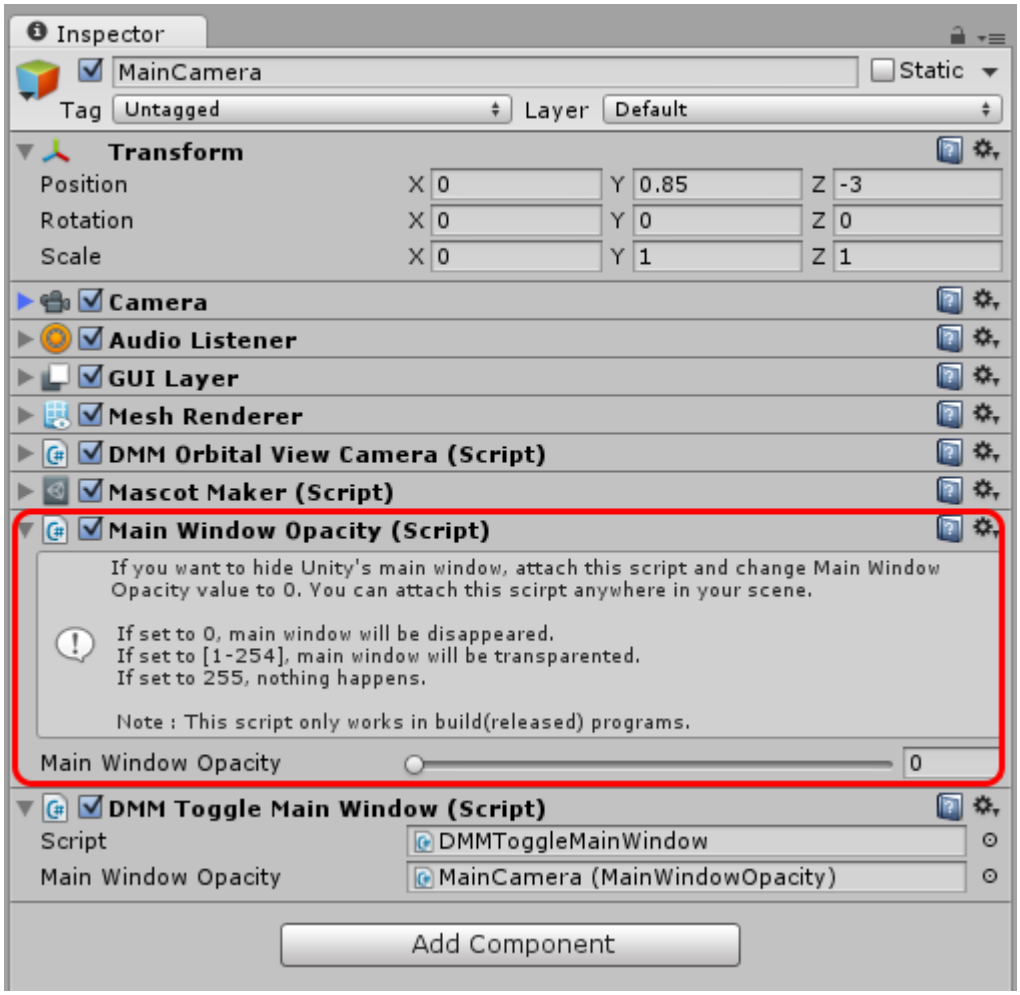
詳しくは Demo04_uGUI.unity のデモをご覧下さい。

Unity のメインウィンドウを消す方法

デスクトップマスコットキャラクターだけを表示して、Unity のメインウィンドウは非表示にしたい場合があります。そのような場合には、シーンのどこかに MainWindowOpacity.cs スクリプトをアタッチして、Main Window Opacity のスライダの値を 0 に設定して下さい。そうするとリリースビルド時には Unity のメインウィンドウは表示されなくなります。**（注意：Unity Editor 上では非表示になりません。必ずビルドして確認して下さい）**

MainWindowOpacity コンポーネントは、メニューの
Menu > Component > DesktopMascotMaker > MainWindowOpacity からも追加できます。

0 に設定すると、メインウィンドウは完全に非表示になります。
１～254 の間で設定すると、メインウィンドウは半透明になります。
255 に設定すると、メインウィンドウは通常の状態となり、何も起こりません。



詳しくは、デモシーン *Demo05_HideMainWindow.unity* をご参照下さい。

MascotMaker クラスリファレンス

シングルトンインスタンスを通して変数や関数にアクセスする

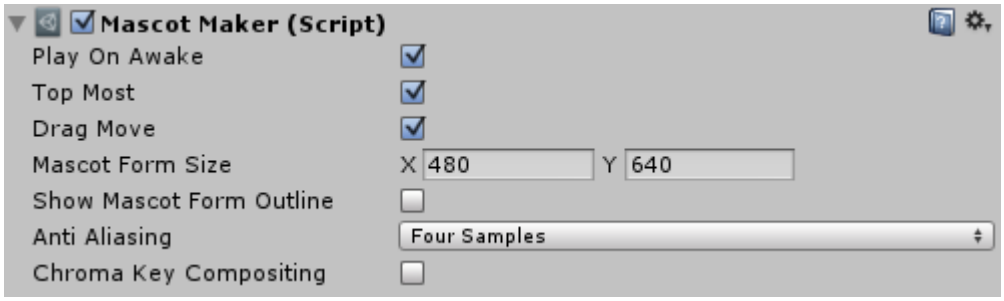
MascotMaker クラスはシングルトンデザインパターンで実装されています。そのため MascotMaker クラスの変数や関数にアクセスするためには、*MascotMaker.Instance* を利用してください。

```
using DesktopMascotMaker;  // use this namespace
...
MascotMaker.Instance.xxx = ...  // like this
```

MascotMaker コンポーネントをカメラにアタッチすると、カメラの表示内容が MascotForm に転送され、表示されます。

MascotMaker コンポーネントはシングルトン デザイン パターンで実装されているため、1 つのシーンに 1 つまでしか存在を許されていません。もし 1 つのシーンに、複数のマスコットを表示したい場合は、MascotMakerMulti コンポーネントを利用して下さい。MascotMakerMulti コンポーネントは通常の MonoBehaviour コンポーネントとして実装されているため、1 つのシーンに複数置くことができます。MascotMakerMulti について詳しくは、このドキュメントの MascotMakerMulti クラスリファレンスまたはデモシーン Demo06_MultipleMascots.unity をご参照下さい。

MascotMaker コンポーネントの設定値



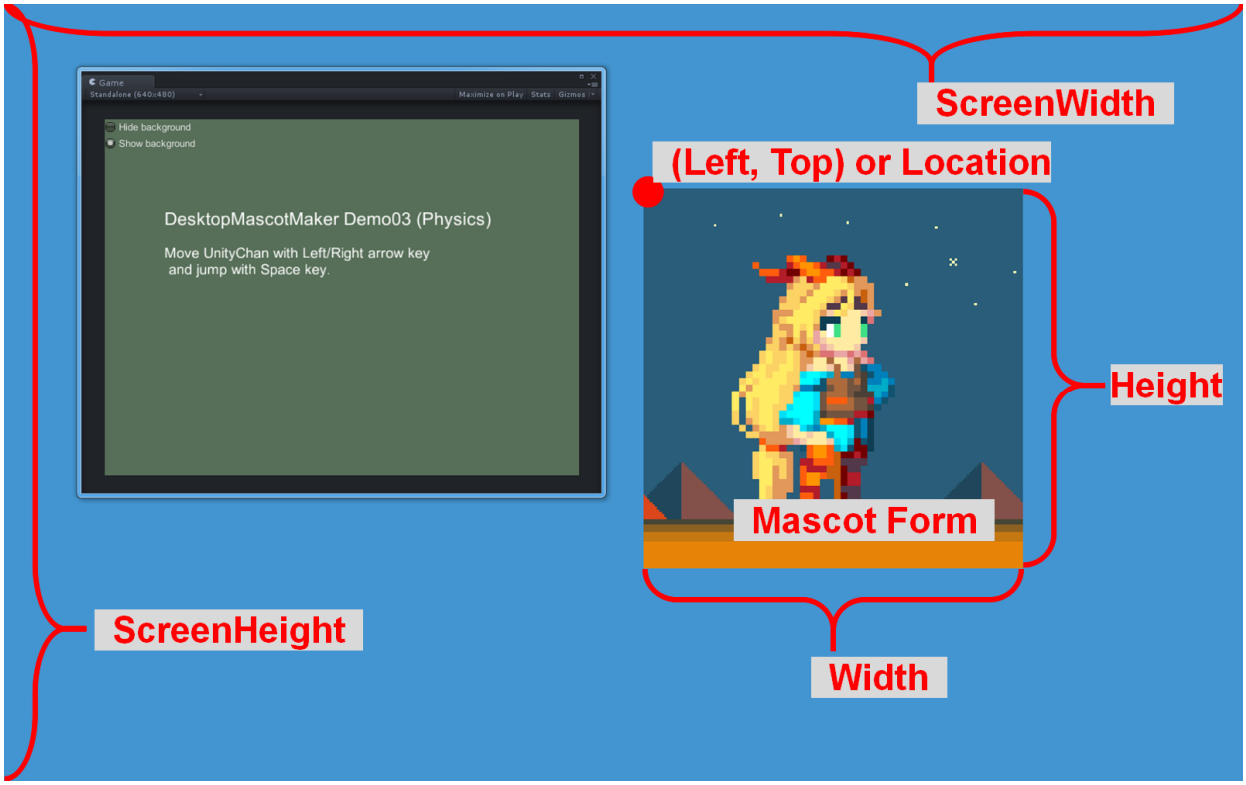
MascotMaker コンポーネントの設定値は、以下の変数を介して、スクリプトからも変更できます。

```
bool MascotMaker.Instance.PlayOnAwake           // bool value
bool MascotMaker.Instance.TopMost               // bool value
bool MascotMaker.Instance.DragMove              // bool value
Vector2 MascotMaker.Instance.MascotFormSize     // Vector2 value
bool MascotMaker.Instance.ShowMascotFormOutline  // bool value
MascotMaker.AntiAliasingType MascotMaker.Instance.AntiAliasing // enum value
bool MascotMaker.Instance.ChromaKeyCompositing   // bool value
```

Mascot Form の位置

Mascot Form とは、マスコットを表示するための矩形の領域のことです（下図参照）。
Mascot Form の位置は、*Left, Top* (または *Location*) で操作できます。

```
int MascotMaker.Instance.Left
int MascotMaker.Instance.Top
System.Drawing.Point MascotMaker.Instance.Location
int MascotMaker.Instance.Width
int MascotMaker.Instance.Height
int MascotMaker.Instance.ScreenWidth  // read only
int MascotMaker.Instance.ScreenHeight  // read only
```



例えば、スクリプトで以下の様に書くと Mascot Form の左上が、スクリーンの左上の位置に合わさります。

```
MascotMaker.Instance.Left = 0;
MascotMaker.Instaince.Top = 0;
```

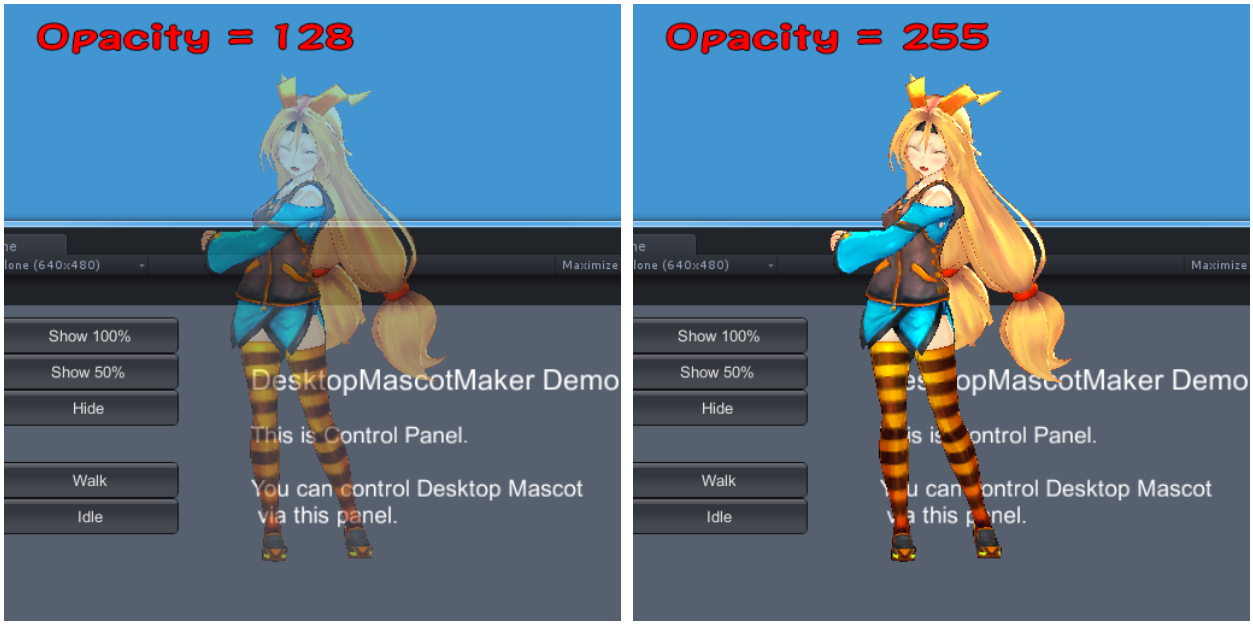
もし、Mascot Form をランタイムに動かしたい場合は、*MascotMaker.Instance.Left/Top* の値を以下の様に Unity の Update 関数の中で変更すれば可能です。

```
void Update()
{
	MascotMaker.Instance.Left += Time.deltaTime * speed;
}
```

マスコットの不透明度

マスコットの不透明度、表示、非表示は以下のようにして変更します。

```
MascotMaker.Instance.Opacity = 128;  // マスコットの不透明度 [0-255]
MascotMaker.Instance.Hide();  // マスコットを完全に非表示にする
MascotMaker.Instance.Show();  // マスコットを表示する
```



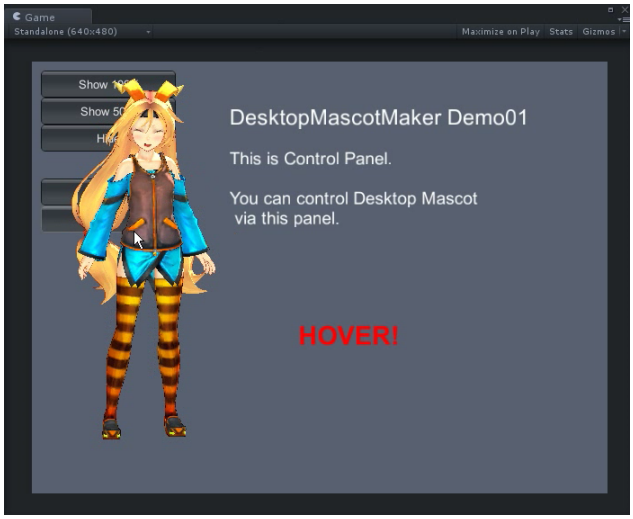
Opacity の値を [1-254]の範囲で設定すれば、マスコットは半透明になります。

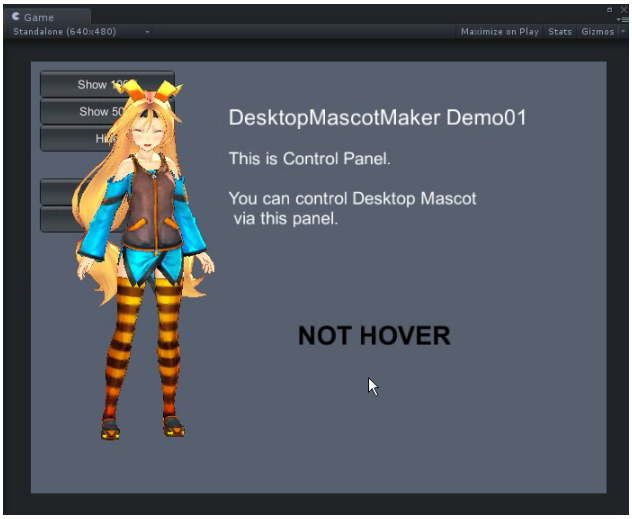
もしマスコットを非表示にしたい場合は、Hide()関数を利用して下さい。`MascotMaker.Instance.Hide()` はマスコットを完全に非表示にし、内部的な描画処理が発生しません。一方で、`MascotMaker.Instance.Opacity = 0` でマスコットを完全に非表示にした場合、内部的には描画処理が発生したままとなり、相対的に処理パフォーマンスが低くなります。

マウスがマスコットに重なっているか？

マウスカーソルがマスコットの上に重なっているかを検知します。もし、マウスカーソルがマスコットの上に重なっている場合、*MascotMaker.Instance.IsMouseHover* は True になります。重なっていない場合、False になります。

```
bool mouseHover = MascotMaker.Instance.IsMouseHover;  // read only
```





詳しくはデモシーン Demo01_General.unity をご参照下さい。

Event Settings イベント関数の設定

Desktop Mascot Maker では、マウスイベント、マスコットのアクティベート／ディアクティベートイベント、キーイベント、ドラッグアンドドロップイベント、ムーブイベントが利用できます。
以下の様に特定の関数をイベントハンドラに登録して利用します。

```
// MascotMaker のEventHandler に関数を登録します
MascotMaker.Instance.OnLeftMouseDown += MouseDown;
...
void MouseDown(object sender, MouseEventArgs e)
{
        // マスコット上でマウスクリックするとこの部分のコードが実行されます
        Debug.Log("Clicked!");
}
```

MascotMaker で利用できるイベントハンドラのリストです。

Event	Event Handler Type	
OnLeftMouseDown	MouseEventHandler	マウス左ボタンが押されたときに発生
OnLeftMouseUp	MouseEventHandler	マウス左ボタンが放されたときに発生
OnLeftDoubleClick	MouseEventHandler	マウス左ボタンがダブルクリックされたときに発生
OnRightMouseDown	MouseEventHandler	マウス右ボタンが押されたときに発生.
OnRightMouseUp	MouseEventHandler	マウス右ボタンが放されたときに発生
OnRightDoubleClick	MouseEventHandler	マウス右ボタンがダブルクリックされたときに発生
OnMiddleMouseDown	MouseEventHandler	マウス中ボタンが押されたときに発生
OnMiddleMouseUp	MouseEventHandler	マウス中ボタンが放されたときに発生
OnMiddleDoubleClick	MouseEventHandler	マウス中ボタンがダブルクリックされたときに発生
OnMouseWheel	MouseEventHandler	マウスホイールが回転したときに発生
OnActivated	EventHandler	マスコットの Form がアクティベートされたときに発生
OnDeactivate	EventHandler	マスコットの Form がディアクティベートされたときに発生
OnKeyDown	KeyEventHandler	マスコットの Form でキーが押されたときに発生
OnKeyUp	KeyEventHandler	マスコットの Form でキーが放されたときに発生
OnDragDrop	DragEventHandler	ドラッグアンドドロップ動作が完了したときに発生
OnDragEnter	DragEventHandler	マスコットの Form 内にドラッグで入ったときに発生
OnDragLeave	EventHandler	マスコットの Form 内からドラッグで出たときに発生
OnDragOver	DragEventHandler	マスコットの Form 上をドラッグしているときに発生
OnMove	EventHandler	マスコットの Form がマウス操作で動いているときに発生

詳しくはデモシーン *Demo02_Event.unity* をご参照下さい。

MascotMakerMulti クラスリファレンス

もし複数のマスコットを同時に表示したい場合、MascotMakerMulti コンポーネントをカメラにアタッチして利用して下さい。MascotMakerMulti コンポーネントは、メニューの
Menu > Component > DesktopMascotMaker > MascotMakerMulti からアタッチできます。

MascotMaker コンポーネントは、シングルトン デザイン パターンで実装されています。ですので、MascotMaker コンポーネントは、１つのシーンに１つまでしか置くことはできません（２つ以上のコンポーネントをシーン上に置こうとすると、１つを除いて他のコンポーネントは削除されます）。一方で、MascotMakerMulti コンポーネントは、通常の MonoBehaviour コンポーネントとして実装されています。そのため、１つのシーンに複数の MascotMakerMulti コンポーネントを置くことができます

MascotMaker コンポーネントと MascotMakerMulti コンポーネントは、変数へのアクセス法が違います。MascotMaker コンポーネントを使う場合、以下の様に変数にアクセスします。

```
MascotMaker.Instance.Left =  100;
```

一方で、MascotMakerMulti コンポーネントを使う場合、はじめに（インスペクタ上での登録や、GetComponent 関数などの）何らかの方法で、MascotMakerMulti のインスタンスを取得しなければなりません。スクリプトでは例えば以下のようになります。

```
// 変数を宣言する
public MascotMakerMulti mascotMakerMulti;

void Start ()
{
        // GetComponent 関数でMascotMakerMulti のインスタンスを取得する
        mascotMakerMulti = GetComponent<MascotMakerMulti>();
        // MascotMakerMulti のインスタンスを介して変数にアクセスする
        mascotMakerMulti.Left = 100;
}
```

全ての MascotMakerMulti の変数は MascotMaker の変数と機能的に全く同一です。例えば、
`mascotMakerMulti.Left` の変数の機能的な意味は `MascotMaker.Instance.Left` と同一です。
`mascotMakerMulti.Show()`の関数の機能的な意味は`MascotMaker.Instance.Show()`と同一です。
`mascotMakerMulti.OnLeftMouseDown` のイベントハンドラの機能的な意味は
`MascotMaker.Instance.OnLeftMouseDown` と同一です。

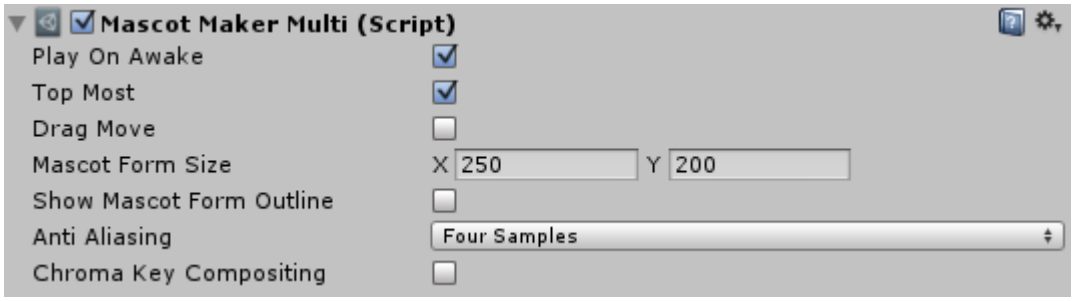
要するに、もし MascotMaker の代わりに MascotMakerMulti を利用したい場合は、*MascotMaker.Instance* を *mascotMakerMulti*(ここで *mascotMakerMulti* は *MascotMakerMulti* のインスタンスです)に変更すれば利用できます。

詳しくは、デモシーン Demo06_MultipleMascots.unity をご参照下さい。
特に、スクリプト DMMUnityChan2DAIForMulti.cs に MascotMakerMulti の使い方が書いてあります。

補足: 1 つの MascotMaker コンポーネントと複数の（あるいは単数の）MascotMakerMulti コンポーネントは同時に使うことができます。

MascotMakerMulti コンポーネントの設定値

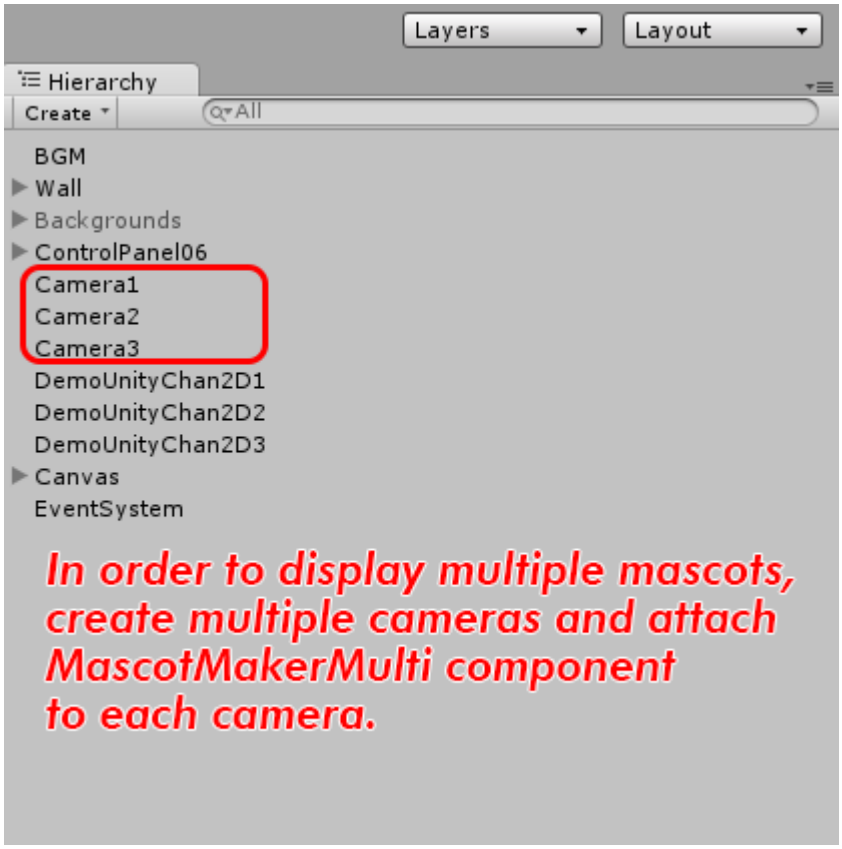
MascotMakerMulti コンポーネントの設定項目の意味は MascotMaker コンポーネントのものと同一です。各項目の意味については、クイックスタートをご参照下さい。



MascotMaker コンポーネントの設定値は、以下の変数を介して、スクリプトからも変更できます。

```
// mascotMakerMulti は MascotMakerMulti コンポーネントのインスタンスです
 bool mascotMakerMulti.PlayOnAwake           // bool value
 bool mascotMakerMulti.TopMost               // bool value
 bool mascotMakerMulti.DragMove              // bool value
 Vector2 mascotMakerMulti.MascotFormSize     // Vector2 value
 bool mascotMakerMulti.ShowMascotFormOutline  // bool value
 MascotMaker.AntiAliasingType mascotMakerMulti.AntiAliasing // enum value
 bool mascotMakerMulti.ChromaKeyCompositing   // bool value
```

MascotMakerMulti コンポーネントは、複数のカメラのそれぞれにアタッチされて利用されます。例えば、もし３つの独立したマスコットを表示したい場合は、３つのカメラを作成し、そのカメラそれぞれに MascotMakerMulti コンポーネントをアタッチします。

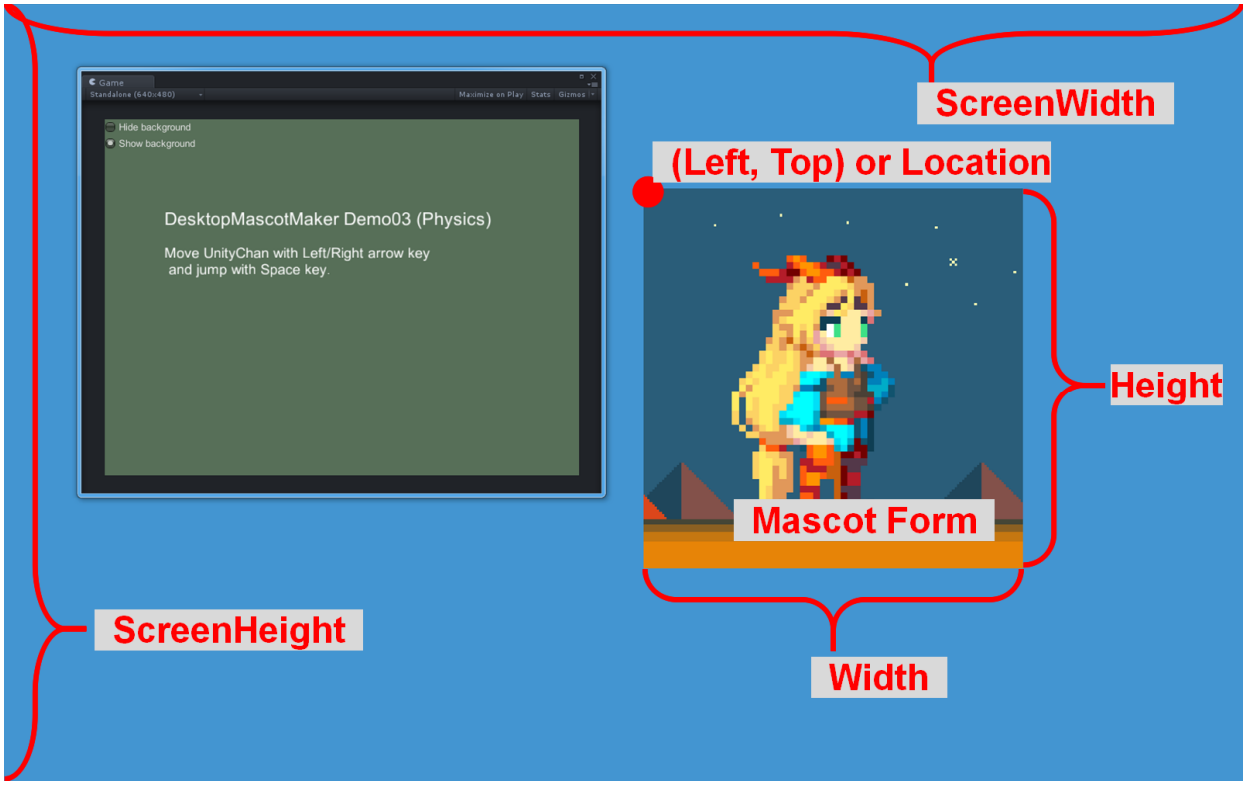


詳しくは、デモシーン Demo06_MultipleMascots.unity をご覧ください。

Mascot Form の位置

Mascot Form とは、マスコットを表示するための矩形の領域のことです（下図参照）。
Mascot Form の位置は、*Left, Top* (または *Location*) で操作できます。

```
// mascotMakerMulti はMascotMakerMulti コンポーネントのインスタンスです
int mascotMakerMulti.Left
int mascotMakerMulti.Top
System.Drawing.Point mascotMakerMulti.Location
int mascotMakerMulti.Width
int mascotMakerMulti.Height
int mascotMakerMulti.ScreenWidth  // read only
int mascotMakerMulti.ScreenHeight  // read only
```



例えば、スクリプトで以下の様に書くと Mascot Form の左上が、スクリーンの左上の位置に合わさります。

```
// mascotMakerMulti はMascotMakerMulti コンポーネントのインスタンスです
mascotMakerMulti.Left = 0;
mascotMaker.Instaince.Top = 0;
```

もし、Mascot Form をランタイムに動かしたい場合は、*mascotMakerMulti.Left/Top* の値を以下の様に Unity の Update 関数の中で変更すれば可能です。

```
void Update()
{
        // mascotMakerMulti はMascotMakerMulti コンポーネントのインスタンスです
        mascotMakerMulti.Left += Time.deltaTime * speed;
}
```

マスコットの不透明度

マスコットの不透明度、表示、非表示は以下のようにして変更します。

```
// mascotMakerMulti はMascotMakerMulti コンポーネントのインスタンスです
mascotMakerMulti.Opacity = 128;  // マスコットの不透明度[0-255]
mascotMakerMulti.Hide();  // マスコットを完全に非表示にする
mascotMakerMulti.Show();  // マスコットを表示する
```

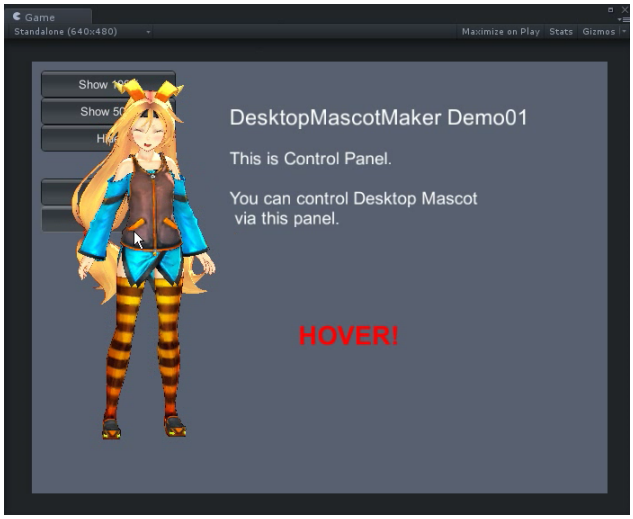
Opacity の値を [1-254]の範囲で設定すれば、マスコットは半透明になります。

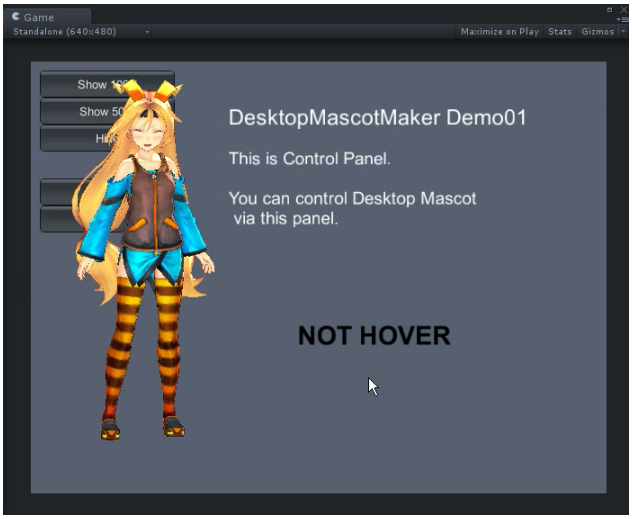
もしマスコットを非表示にしたい場合は、Hide()関数を利用して下さい。*MascotMaker.Instance.Hide()* はマスコットを完全に非表示にし、内部的な描画処理が発生しません。一方で、*MascotMaker.Instance.Opacity = 0* でマスコットを完全に非表示にした場合、内部的には描画処理が発生したままとなり、相対的に処理パフォーマンスが低くなります。

マウスがマスコットに重なっているか？

マウスカーソルがマスコットの上に重なっているかを検知します。もし、マウスカーソルがマスコットの上に重なっている場合、*mascotMakerMulti.IsMouseHover* は True になります。重なっていない場合、False になります。

```
// mascotMakerMulti はMascotMakerMulti コンポーネントのインスタンスです
bool mouseHover = mascotMakerMulti.IsMouseHover;  // read only
```





Event Settings イベント関数の設定

Desktop Mascot Maker では、マウスイベント、マスコットのアクティベート／ディアクティベートイベント、キーイベント、ドラッグアンドドロップイベント、ムーブイベントが利用できます。
以下の様に特定の関数をイベントハンドラに登録して利用します。

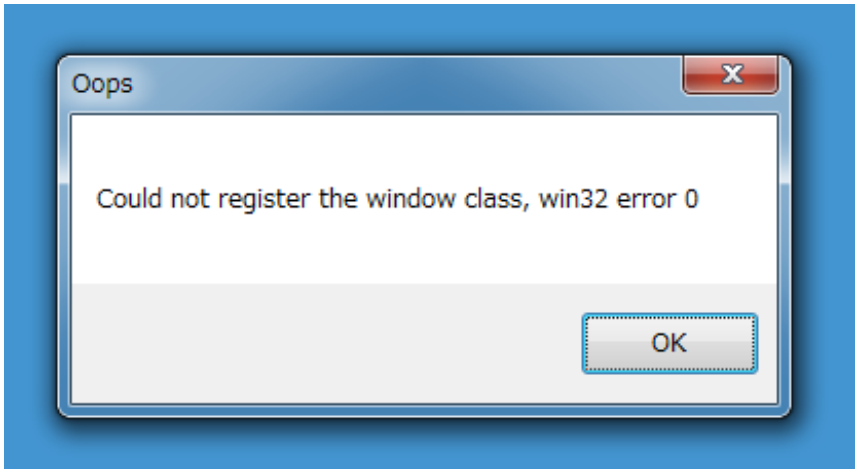
```
// MascotMakerMulti のEventHandler に関数を登録します
// mascotMakerMulti はMascotMakerMulti コンポーネントのインスタンスです
mascotMakerMulti.OnLeftMouseDown += MouseDown;
...
void MouseDown(object sender, MouseEventArgs e)
{
        // マスコット上でマウスクリックするとこの部分のコードが実行されます
        Debug.Log("Clicked!");
}
```

MascotMakerMulti で利用できるイベントハンドラのリストです。 (MascotMaker と全く同一です)

Event	Event Handler Type	
OnLeftMouseDown	MouseEventHandler	マウス左ボタンが押されたときに発生
OnLeftMouseUp	MouseEventHandler	マウス左ボタンが放されたときに発生
OnLeftDoubleClick	MouseEventHandler	マウス左ボタンがダブルクリックされたときに発生
OnRightMouseDown	MouseEventHandler	マウス右ボタンが押されたときに発生.
OnRightMouseUp	MouseEventHandler	マウス右ボタンが放されたときに発生
OnRightDoubleClick	MouseEventHandler	マウス右ボタンがダブルクリックされたときに発生
OnMiddleMouseDown	MouseEventHandler	マウス中ボタンが押されたときに発生
OnMiddleMouseUp	MouseEventHandler	マウス中ボタンが放されたときに発生
OnMiddleDoubleClick	MouseEventHandler	マウス中ボタンがダブルクリックされたときに発生
OnMouseWheel	MouseEventHandler	マウスホイールが回転したときに発生
OnActivated	EventHandler	マスコットの Form がアクティベートされたときに発生
OnDeactivate	EventHandler	マスコットの Form がディアクティベートされたときに発生
OnKeyDown	KeyEventHandler	マスコットの Form でキーが押されたときに発生
OnKeyUp	KeyEventHandler	マスコットの Form でキーが放されたときに発生
OnDragDrop	DragEventHandler	ドラッグアンドドロップ動作が完了したときに発生
OnDragEnter	DragEventHandler	マスコットの Form 内にドラッグで入ったときに発生
OnDragLeave	EventHandler	マスコットの Form 内からドラッグで出たときに発生
OnDragOver	DragEventHandler	マスコットの Form 上をドラッグしているときに発生
OnMove	EventHandler	マスコットの Form がマウス操作で動いているときに発生

よくある質問とその答え

Q： このメッセージボックスは何ですか？



A： すみません、これはMonoの内部的なメッセージです。しかし、このメッセージはあなたのプロジェクトに影響しません。無視して下さい

ご不便をおかけいたしまして申し訳ございません。
このメッセージウィンドウは、Unity に含まれる mono の dll 内で生成されています。
(...¥Unity¥Editor¥Data¥Mono¥lib¥mono¥2.0¥System.Windows.Forms.dll)

こちらのコードの 894 行目が呼び出されることによって発生します。
https://github.com/mono/mono/blob/mono-2-10/mcs/class/Managed.Windows.Forms/System.Windows.Forms/XplatUIWin32.cs

このメッセージウィンドウは、Unity のエディタ環境内で Win32RegisterClass が false を返してしまうために表示されます。このメッセージウィンドウは、Unity のエディタ環境内でのみ表示されます。**あなたが最終的にビルドして作るプログラムではこちらのメッセージウィンドウは表示されません**のでご安心下さい。

Q： パフォーマンスチューニングはどのようにしたら良いですか？

A： インスペクタ上で Mascot Form のサイズを可能な限り小さくして下さい

はじめに、Desktop Mascot Maker の処理パフォーマンスは Mascot Form のサイズの大きさに依存します。ですので、インスペクタ上から設定できる Mascot Form のサイズを可能な限り小さいものにして下さい。Mascot Form のサイズはプログラム実行時にも変更が可能です。

次に、もしマスコットを非表示にする場合は、Hide()関数を利用して下さい。Opacity = 0 にするよりも処理パフォーマンスが少し向上します。

Q： プレイモードにしてみたのですが、ユニティちゃんが見当たりません

A： プロジェクトを再起動してみてください