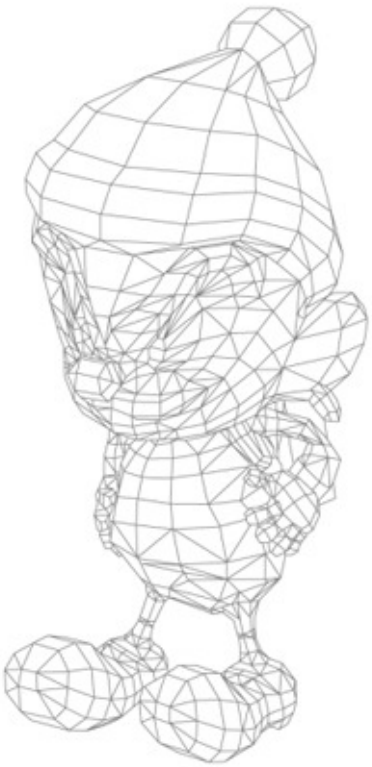
MASCOT CAPSULE®
REAL-TIME 3DRENDERING ENGINE

BAC File Ver.6.0
Data Format Specification

MascotCapsuleV3

日本語版



Ver.2.0

HI CORP.

－更新履歴－

2003 年 10 月 8 日
・Ver.1.0　新規作成

2005 年 6 月 15 日
・Ver.1.0

更新内容
- 文章の一部およびレイアウトが修正

2007 年 3 月 20 日
・Ver.2.0

更新内容
- 文章およびレイアウトが修正
- サンプルコードを追加

― 目 次 ―

1. はじめに

本稿は **MascotCapsuleV2/V3**（以下、**V2/V3** と記す）の中間ファイルである **BAC ファイル**（以下、**BAC** と記す）について解説しています。
BAC ファイルとは、**V2/V3** で利用するモデル情報を持つデータファイルのことで、バージョン5.0/6.0（以下、**BAC5/BAC6** と記す）があります。**BAC5** は、**V2** で利用するモデル情報を持つデータファイルです。それに対して **BAC6** は、**V3** で利用するモデル情報を持つデータファイルになります。本稿では、この **BAC6** のデータフォーマットについて詳細を説明します。

1.1. BAC6 とは

BAC6 とはテキストフォーマット形式（BAC5 はバイナリフォーマット形式のファイルです）のファイルで、**V3** で利用するモデルデータに適用されているマテリアルの属性や、ボーン情報などを文字列で記述していきます。
アニメーション情報に関しては **TRA4** ファイルに定義します。別紙「TRA File Ver.4.0 Data Format Specification」を参照して下さい。**BAC6** と **TRA4** は **V3** で利用する対となる中間ファイルです。
BAC6 のファイル拡張子は「**.bac**」です。テキスト形式で扱う文字規約は「ASCII コード」の範囲でサポートしています。

2. ファイル構造

ここでは **BAC6** の基本となる構造と規約について説明します。**BAC6** の基本構造を理解する上で**「2.1 ～ 2.2」**を参照して下さい。

2.1. マクロ定義

ここでは **BAC6** に記述されるパラメータのマクロ定義について説明します。**「2.2 基本コード規約」**を参照する上で、ベーシックタイプとしてパラメータ部を以下のマクロで定義しています。

1） [STRING]
ヌル終端処理された文字列を意味しています。
文字列はダブルクォーテーション(“”)で囲い、255 文字(byte)以内に収めてください。

2） [A | B]
A あるいは B を意味しています。
A 又は B はヌル終端文字列で 255 文字(byte)以内に収めてください。

3） [INT]
整数値を表す文字列を意味します。
文字列(数値)の範囲は int 型のビット範囲に収めて下さい。

4） [FLOAT]
浮動小数値を表す文字列を意味します。
文字列(数値)の範囲は float 型のビット範囲に収めて下さい。

5） **...**
任意の繰り返しを意味します。

6） **;** (セミコロン)
セミコロンから行末まではコメント文字列を意味します。

2.2. 基本コード規約

BAC6 の基本となるコード規約として、ファイル識別情報として先頭に「**;BAC**」と定義し、ヘッダーを意味する**(Head)**チャンクとモデル情報を明記する**(Figure)**チャンクをそれぞれ定義する必要があります。
また、これら定義の順序に変更は認められておらず、定義する順序として必ず「;BAC」で始まり、次にヘッダーの(Head)チャンク、最後にモデル情報を明記する(Figure)チャンクを定義する構造になっています。(Head)と(Figure)チャンクはそれぞれに属する各チャンクを持ち、その各チャンクは固有のコンポーネントを更に定義していきます。
これらを踏まえて「【図】基本フォーマット」を参照して下さい。

【図】基本フォーマット

```
;BAC
( Head
     ( bacVersion [FLOAT] )
)
( Figure
     ( name [STRING] )

     ( Textures
              ( i2 [INT] [INT] )
              …
     )

     ( Colors
              ( f3 [FLOAT] [FLOAT] [FLOAT] )
              …
     )

     ( Materials
              ( material
                           ( blendMode       [normal | add | sub | half] )
                           ( doubleFace      [true | false] )
                           ( transparent     [true | false] )
                           ( lighting        [true | false] )
                           ( textureIndex    [INT] )
                           ( colorIndex      [INT] )
                           ( specular        [FLOAT] )
                           ( alpha           [FLOAT] )
                       ( shininess       [FLOAT] )
              )
     )

     ( Vertices
              ( coords
                           ( pnt [FLOAT] [FLOAT] [FLOAT] )
                           …
              )
              ( normals
                           ( vct [FLOAT] [FLOAT] [FLOAT] )
                           …
              )
     )
```

```
    ( Bones
          ( bone
                        ( name            [STRING] )
                        ( hasChild        [true | false] )
                        ( hasBrother      [true | false] )
                        ( translate       [FLOAT] [FLOAT] [FLOAT] )
                        ( rotate          [FLOAT] [FLOAT] [FLOAT] )
                        ( handle          [FLOAT] [FLOAT] [FLOAT] )
                        ( vertexIndices
                                [INT] [INT] …
                        )
           )
          …
    )

    ( TextureCoords
           ( f2 [FLOAT] [FLOAT] )
           …
    )

    ( Polygons
           ( face [INT] ( i3 [INT] [INT] [INT] )( i3 [INT] [INT] [INT] ))
           ( face [INT] ( i4 [INT] [INT] [INT] [INT] )( i4 [INT] [INT] [INT] [INT] ))
           …
    )
    ( DynamicPolygons
        ( group
           ( name [STRING] )
           ( face [INT] ( i3 [INT] [INT] [INT] )( i3 [INT] [INT] [INT] ))
           ( face [INT] ( i4 [INT] [INT] [INT] [INT] )( i4 [INT] [INT] [INT] [INT] ))
           …
        )
        …
    )
)
```

3. 各チャンクについて

ここでは、BAC6 に記述する各チャンクの詳細を説明します。

3.1. (Head)チャンク

(bacVersion [FLOAT]) チャンクには BAC のバージョンを指定し、ファイルヘッダーを定義します。BAC6 はバージョン 6.0 なので[FLOAT]には **6.0** と文字列で定義します。

```
( Head
        ( bacVersion 6.0 )
)
```

3.2. (Figure)チャンク

(Figure)チャンクには、モデルデータ(ポリゴンメッシュ)の情報を記述します。(Figure)チャンク内の各チャンクに詳細が記述されます。それらの各チャンクに関する詳細は **3.2.1〜3.2.9** を参照して下さい。

```
( Figure
        ( name … )
        ( Textures … )
        ( Materials … )
        ( Vertices … )
        ( Bones … )
        ( Polygons … )
        ( DynamicPolygons … )
)
```

(name)チャンクにはモデルデータの名前を記述します。
(Textures)チャンクには任意の数だけテクスチャ情報を記述します。
(Materials)チャンクには任意の数だけマテリアル情報を記述します。
(Vertices)チャンクには任意の数だけモデルデータの頂点情報を記述します。
(Bones)チャンクには任意の数だけボーン情報を記述します。但し、最低でも１つはボーン情報を記述しなければいけません。
(Polygons)チャンクには任意の数だけモデルデータのポリゴン情報を記述します。
(DynamicPolygons)チャンクには任意の数だけ動的ポリゴン情報を記述します。

3.2.1. (name)チャンク

(Figure)チャンクの中の(name [STRING])チャンクには、モデルデータの名前を指定します。
モデルの名前を必ずしも定義する必要はありませんが、定義する場合は(name "")のようにダブルクォーテーションで囲む必要があります。

```
( name "figure" )
```

3.2.2. (Textures)チャンク

(Textures) チャンクは、(i2 [INT] [INT])チャンクを使用してモデルデータに使用されているテクスチャ画像のピクセルサイズ(縦横)を列挙します。テクスチャが必要ないモデルデータ（ポリゴンメッシュ）には、(Figure)チャンク内にこのチャンクを登録する必要はありません。

3.2.2.1 (i2 [INT] [INT])チャンク

テクスチャ画像のピクセルサイズ(幅、高さ)を列挙します。このチャンクには、それぞれ上から順に 0 から始まるインデックス ID が自動的に割り当てられ、インデックス ID は (Materials)チャンク内の (textureIndex [INT])チャンクによって参照されます。

```
( Textures
        ( i2 256 256 ) ; index 0
        ( i2 128 128 ) ; index 1
        …
)
```

3.2.3. (Colors)チャンク

(Colors)チャンクは、(f3 [FLOA] [FLOATA] [FLOATA])チャンクを使用して、モデルデータに使用されているカラーポリゴン用のカラーテーブルを登録します。
モデルデータにカラーポリゴンが必要でない場合は、このチャンクを(Figure)チャンクに登録する必要はありません。

3.2.3.1 (f3 [FLOAT] [FLOAT] [FLOAT])チャンク

カラーポリゴン用のカラーテーブルを列挙します。このチャンクにはそれぞれ上から順に 0 から始まるインデックス ID が自動的に割り当てられ、(Materials)チャンク内の(colorIndex [INT])チャンクによって参照されます。カラーの有効範囲は RGB(0.0, .0.0, 0.0)を黒色として、RGB(1.0, 1.0, 1.0)を白色とした範囲でカラー情報を記述します。

```
( Colors
        ( f3 1.000 0.000 0.000 ) ; index 0 【赤】
        ( f3 0.000 0.000 1.000 ) ; index 1 【青】
        …
)
```

3.2.4. (Materials)チャンク

モデルデータのポリゴンに設定される任意の表示属性タイプを(material)チャンクを使用して登録します。このチャンクは必ず(Figure)チャンク内に登録する必要があります。

```
( Materials
        ( material
                ( blendMode   normal )
                ( doubleFace  true )
                ( transparent false )
                ( lighting    true )
                ( textureIndex -1 )
                ( colorIndex    0 )
                ( specular      0.000 )
                ( alpha         0.000 )
                ( shininess     0.000 )
        )
        …
)
```

3.2.4.1 (material)チャンク

ポリゴンに設定するマテリアルの任意の表示属性タイプを、専用の各チャンク(3.2.4.2～3.2.4.10)を使用して列挙します。このチャンクはそれぞれ上から順に 0 から始まるインデックス ID が自動的に割り当てられ、インデックス ID は(Polygon)チャンク内の(face)チャンクによって参照されます。
このチャンク内で列挙される専門の各チャンクの記述順序に規定はありません。
(Materials)チャンク内にこのチャンクは必ず１つは登録しなければいけません。

3.2.4.2 (blendMode [normal | add | sub | half])チャンク

(blendMode [normal | add | sub | half]) チャンクにはポリゴンが表示されるフラグメント処理を以下の属性からいずれか１つ指定します。

※ d_c = デスティネーションカラー、 s_c = ソースカラー

normal	標準のフラグメント処理をします(置換)	$d_c = s_c$
add	ポリゴンのカラーを塗り重ねます(加算)	$d_c = d_c + s_c$
sub	ポリゴンのカラー値を減算します(減算)	$d_c = d_c - s_c$
half	ポリゴンカラーの半分の値で重ねます(半透過)	$d_c = 0.5\ d_c + 0.5\ s_c$

3.2.4.3 (doubleFace [true | false])チャンク

(doubleFace [true | false])チャンクはポリゴンの両面を描画するかどうかを指定します。

true	ポリゴンの両面描画を有効にします
false	ポリゴンの表面描画を有効にし、裏面は描画しません

3.2.4.4 (transparent [true | false])チャンク

(transparent [true | false])チャンクはテクスチャの BMP カラーパレット 0 番の色を透過させるかどうかを指定します。

true	透過効果を有効にします
false	透過効果を無効にします

3.2.4.5 (lighting [true | false])チャンク

(lighting [true | false])チャンクはライティング処理の有無を指定します。

true	ライティング効果を有効にします
false	ライティング効果を無効にします

3.2.4.6 (textureIndex [INT])チャンク

(textureIndex [INT])は(Textures)チャンクで自動的に割り当てられたインデックス ID を指定します。テクスチャポリゴンとして処理しない場合は **-1** を指定します。この場合、(Textures)チャンクにインデックス ID が登録されている場合においても、テクスチャポリゴンの設定は無効となります。またテクスチャポリゴン設定の(textureIndex [INT])チャンクとカラーポリゴン設定の(colorIndex [INT])チャンクは排他関係にあります。

3.2.4.7 (colorIndex [INT])チャンク

(colorIndex [INT])チャンクは(Colors)チャンクで登録したカラーID を指定します。
登録 ID は必ず(Colors)チャンクに明記されている ID でなければいけません。
カラーポリゴンとして処理しない場合は **-1** を指定します。
またカラーポリゴン設定の(colorIndex [INT])チャンクとテクスチャポリゴン設定の(textureIndex [INT])チャンクは排他関係にあります。

3.2.4.8 (specular [FLOAT])チャンク

(specular [FLOAT])チャンクは環境マッピングを設定する鏡面反射フラグ値です。値の有効範囲は 0.00～1.00 で、値は **0.25** 以上で環境マッピングによる鏡面反射処理を有効にします。

3.2.4.9 (alpha [FLOAT])チャンク

(alpha [FLOAT])チャンクはテクスチャの半透過アルファ値を指定します。値の有効範囲は
0.0 ～ 1.0 になります。
この機能は V3 ではサポートされていません。従って必要のない場合は(material)チャンク内にこのチャンクを定義する必要はありません。

3.2.4.10 (shininess [FLOAT])チャンク

(shininess [FLOAT])チャンクは環境マッピングに設定する鏡面反射係数を指定します。値の有効範囲は 0.0 ～ 1.0 になります。
この機能は V3 ではサポートされていません。従って必要のない場合は(material)チャンク内にこのチャンクを定義する必要はありません。

3.2.5. (Vertices)チャンク

(Vertices)チャンクは、モデルデータを構成する頂点の位置情報並びにその頂点に対する法線ベクトルを列挙します。頂点座標を列挙する(coords)チャンクと、法線情報を列挙する(normals)チャンクによって頂点情報を定義します。

```
( Vertices
    ( coords
        ( pnt -1.000 -1.000 0.000 ) ; index 0
        ( pnt 1.000 -1.000 0.000 )  ; index 1
        ( pnt 1.000 1.000 0.000 )   ; index 2
        ( pnt -1.000 0.000 0.000 )  ; index 3
        …
    )
    ( normals
        ( vct 0.000 0.000 1.000 )  ; index 0
        ( vct 0.000 0.000 1.000 )  ; index 1
        ( vct 0.000 0.000 1.000 )  ; index 2
        ( vct 0.000 0.000 1.000 )  ; index 3
        …
    )
)
```

(pnt)チャンク 0 番インデックスが定義する頂点座標は、(vct)チャンクのインデックス 0 番が定義する法線ベクトル値と対の関係性を保持して定義しなくてはいけません。このように定義する頂点座標と法線ベクトル値は、頂点数だけ関係性を維持する必要があります。

3.2.5.1 (coords)チャンク

1 頂点の三次元座標(XYZ)をコンポーネント化して(pnt [FLOAT] [FLOAT] [FLOAT])チャンクに頂点座標を定義します。このチャンクはポリゴンを形成する頂点情報をリストで定義するもので必ず(Vertices)チャンク内に明記する必要があります。

(pnt [FLOAT] [FLOAT] [FLOAT])チャンク

(pnt)チャンクはポリゴンを形成する三次元頂点位置(XYZ)を列挙します。このチャンクは上から順に 0 から始まるインデックス ID が自動的に割り当てられます。このインデックス ID は(normals)チャンク内で列挙される(vct)チャンクのインデックス ID と一致していなければなりません。
頂点座標は制御するボーンと同一の座標系で定義します。

3.2.5.2 (normals)チャンク

1 頂点の三次元法線ベクトルをコンポーネント化して(vct [FLOST] [FLOAT] [FLOAT])チャンクに法線ベクトルを定義します。このチャンクはポリゴンを形成する頂点の法線ベクトル情報をリストで定義するもので、必ず(Vertices)チャンク内に明記する必要があります。

(vct [FLOAT] [FLOAT] [FLOAT])チャンク

(vct)チャンクはポリゴンを形成する頂点の法線ベクトル情報(XYZ)を列挙します。このチャンクは上から順に 0 から始まるインデックス ID が自動的に割り当てられます。このインデックス ID は(coords)チャンク内に列挙される(pnt)チャンクのインデックス ID と一致していなければなりません。
法線ベクトルは制御ボーンと同一の座標系で定義されます。

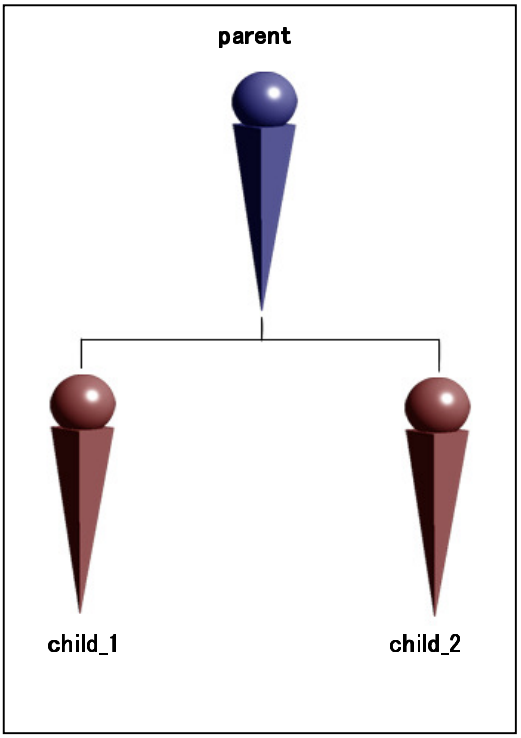
3.2.6. (Bones)チャンク

モデルデータに割り当てられているボーンの階層構造を定義します。モデルデータを構成する上で最低でも１つのボーンが必要です。(Bones)チャンク内で最初に記述されている(bone)チャンクは、モデルデータにおける最上位ボーン（ルート）を表し、必ず１つにする必要があります。最上位に位置するボーンの複数定義は対応していません。
「【図】ボーン構造定義」と「【図】ボーン階層構造」の関係性を参照して下さい。

【図】ボーン構造定義

```
( Bones
     ( bone
        ( name            "parent" )
        ( hasChild        true )
        ( hasBrother      false )
        ( translate       0.000 2.000 0.000 )
        ( rotate          0.000 2.000 1.000 )
        ( handle          0.000 3.000 0.000 )
        ( vertexIndices
           0 1 2 3
        )
     )
     ( bone
        ( name            "child_1" )
        ( hasChild        false )
        ( hasBrother      true )
        ( translate       2.000 0.500 0.000 )
        ( rotate          2.000 0.500 1.000 )
        ( handle          2.000 2.000 0.000 )
        ( vertexIndices
           4 5 6 7
        )
     )
     ( bone
        ( name            "child_2" )
        ( hasChild        false )
        ( hasBrother      false )
        ( translate       1.000 0.500 0.000 )
        ( rotate          1.000 0.500 1.000 )
        ( handle          1.500 1.500 0.000 )
        ( vertexIndices
           8 9 10 11
        )
     )
)
```

【図】ボーン階層構造



3.2.6.1 (bone)チャンク

(bone)チャンクは個々のボーン情報を入れるコンポーネントの親です。モデルデータに２つのボーンが割り当てられている場合、(bone)チャンクを２つ記述する必要があります。このチャンクは１モデルデータに対して必ず１つは(Bones)チャンク内に定義しなくてはいけません。
また、(bone)チャンクは自身と他のボーンとの階層構造等の関係性を定義する子チャンクを持ちます。これらについては以下、(1)～(9)を参照して下さい。

(1) (name [STRING])チャンク

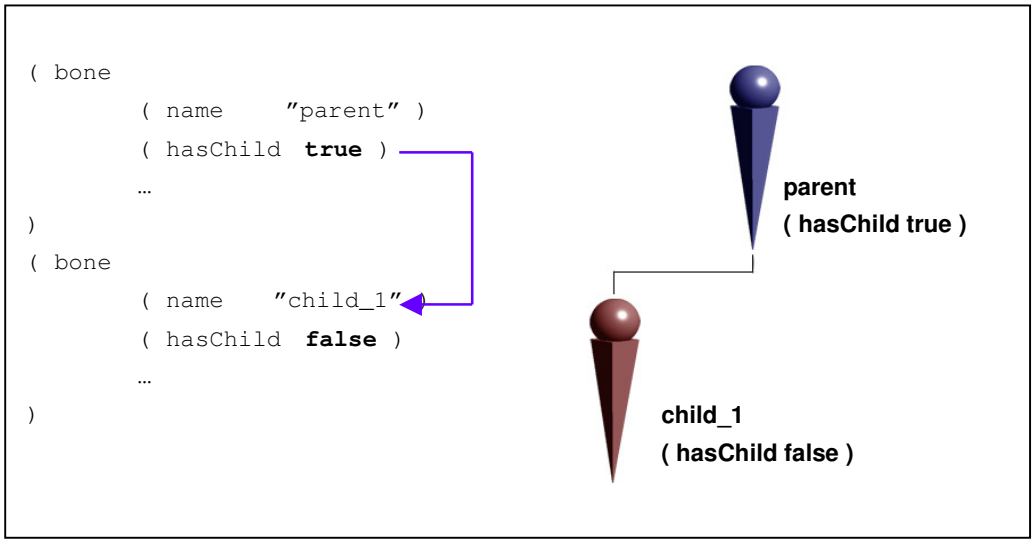
(name)チャンクには、対象となるボーンの名前(文字列は 255 文字(byte)以内で収めてください)を定義します。ボーン名は必ずしも定義する必要はありませんが、定義する場合は(name " ")のように必ずダブルクォーテーションで囲う必要があります。

(2) (hasChild [true | false])チャンク

このチャンクはボーン階層を定義するもので、(bone)チャンク内の(name [STRING])チャンクが示すボーンに、子となるボーンが存在するかどうか定義します。

true	このボーンに子ボーンは存在する
false	このボーンに子ボーンは存在しない

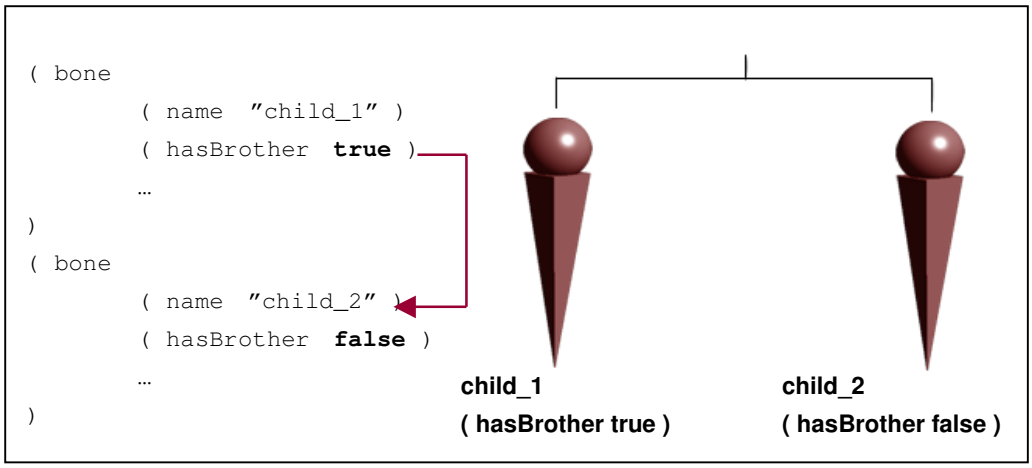
※ true になっていれば次に記述されているボーンはそのボーンの子として定義します。



(3) (hasBrother [true | false])チャンク

(hasBrother [true | false])チャンクは、(bone)チャンク内の(name [STRING])チャンクが示すボーンに、兄弟ボーン(同一階層)が存在するかどうか定義します。

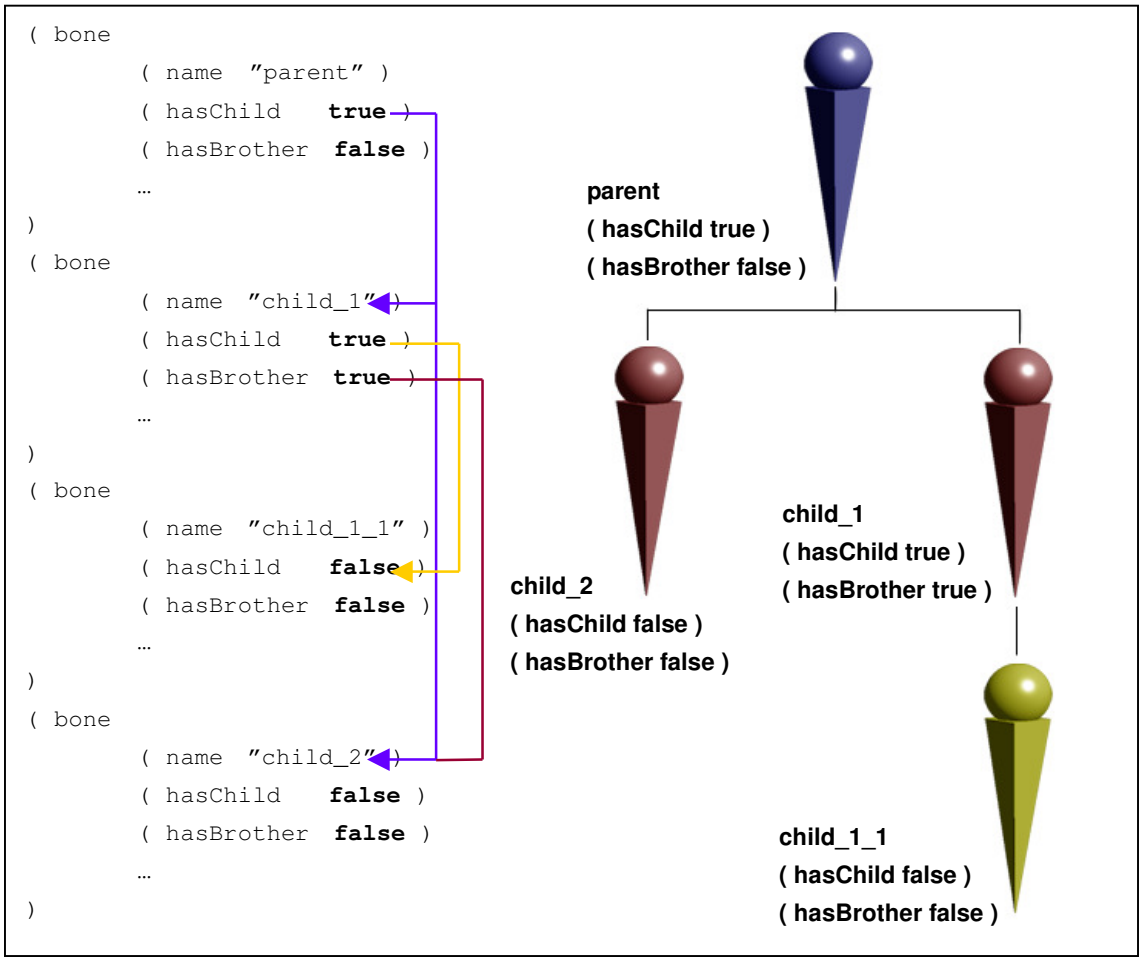
true	このボーンに同階層ボーンは存在する
false	このボーンに同階層ボーンは存在しない



※ 兄弟関係にあり、最後に定義するボーンは FALSE に定義します。
(ボーン「child_2」がこれに準じます。)
これにより同一階層の終端であると認識し、再帰的なボーン階層を定義します。

(4) ボーン構造の再帰処理

(hasChild)チャンクと(hasBrother)チャンクの両方が true に定義されている場合、次のボーンは子として定義され再帰的に処理します。全ての子ボーンを処理が終了した後、改めてその後のボーンを同列の弟として定義します。



(5) (translate [FLOAT] [FLOAT] [FLOAT])チャンク

ボーンの基本姿勢の位置情報は、三次元座標位置 XYZ を原点として定義します。座標は対象ボーンの親ボーン座標系に基づいて、対象となるボーンのグローバル座標(絶対値)の原点を設定します。ボーンはこのチャンクで設定される値を原点として、回転、拡大縮小の中心点とします。

※ 標準姿勢は (translate 0.0 0.0 0.0) です。

(6) (rotate [FLOAT] [FLOAT] [FLOAT])チャンク

このチャンクでは対象ボーンの親ボーンから、+Z 軸方向に伸びる相対ベクトル(方向ベクトル)を定義します。rotate に定義するベクトルは対象となるボーンの translate を基準とした仮想的な先端を意味し、このボーンに続く子ボーンへの距離と同意です。また、rotate は handle と直交関係にあります。

※ 標準姿勢は +Z 軸方向に伸びるベクトルの為、(rotate 0.0 0.0 1.0)となります。

(7) (handle [FLOAT] [FLOAT] [FLOAT])チャンク

このチャンクでは対象ボーンの親ボーンから +Y 軸（translate を基準にした）方向に伸びる相対ベクトル(方向ベクトル)を定義します。また、handle は rotate と直交関係にあります。

※ 標準姿勢の状態では +Y 軸方向に伸びる為、(handle 0.000 1.000 0.000)となります。

(8) ボーン基本姿勢計算式

対象となるボーンの変換行列(4 行 4 列)を次のように定義します。

この時、ローカル変換行列(アフィン変換)を”**L**”と定義し、グローバル変換行列(アフィン変換)を”**W**”と定義します。

$$L=\begin{bmatrix} xx & xy & xz & 0 \\ yx & yy & yz & 0 \\ zx & zy & zz & 0 \\ tx & ty & tz & 1 \end{bmatrix} \quad W=\begin{bmatrix} XX & XY & XZ & 0 \\ YX & YY & YZ & 0 \\ ZX & ZY & ZZ & 0 \\ TX & TY & TZ & 1 \end{bmatrix}$$

この時、対象となるボーンの基本姿勢を決定する各要素は次のように表します。

$$translate=(TX,TY,TZ)$$
$$rotate=(zx,zy,zz)+translate=(zx+TX,zy+TY,zz+TZ)$$
$$handle=(yx,yy,yz)+translate=(yx+TX,yy+TY,yz+TZ)$$

rotate と handle は直交関係にある為、どの回転軸(向き)でも問題ありません。標準姿勢の rotate は +Z 軸に方向ベクトルは伸び handle は+Y 軸に方向ベクトルが伸びています。これらは個々に translate をボーンの原点として伸びる方向ベクトルです。

(9) (vertexIndices [INT] [INT] … …)チャンク

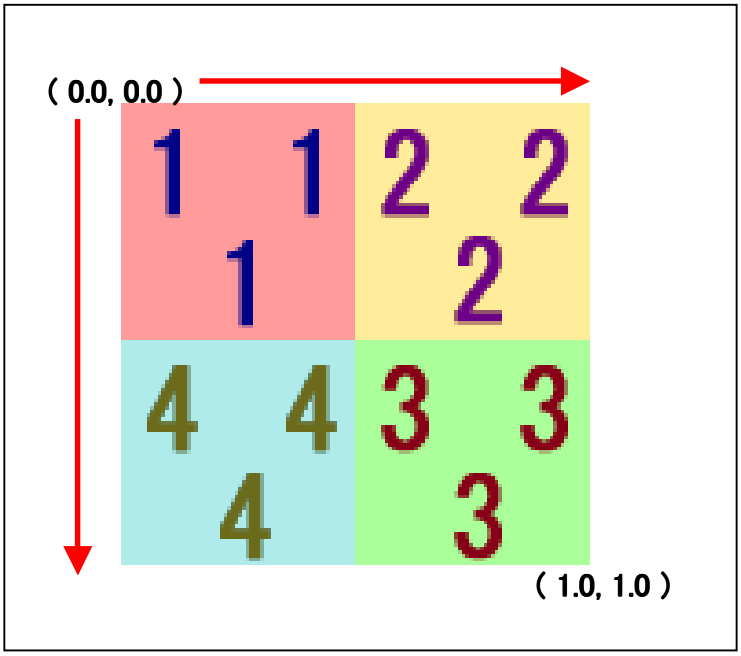
対象ボーンに属している頂点 ID を記述します。この頂点 ID とは(Vertices)チャンクの(coords)チャンクに記述されたインデックス番号を明記します。

属している頂点が無ければ記述の必要はありませんが、どのボーンにも属していない頂点を含むポリゴンは表示されません。

3.2.7. (TextureCoords)チャンク

モデルデータに設定されているテクスチャのUV座標値を任意の数だけ(f2 [FLOAT] [FLOAT])チャンクに記述します。BAC6 へ定義する基本となるテクスチャ座標系は以下の図の通りです。

【MascotCapsuleV3 UV 座標系】



3.2.7.1 (f2 [FLOAT] [FLOAT])チャンク

テクスチャの UV 座標値を 1 つのコンポーネントとして列挙します。このチャンクは上から順に 0 から始まるインデックス ID が自動的に割り当てられます。これにより記述の順序は、(Vertices)チャンクの (coords)チャンクに記述した頂点インデックスと必ず一致するように UV 座標値を列挙しなければいけません。

このチャンクは、(Polygons)チャンクの(face)チャンクによって記述順にインデックスとして参照されます。

```
( TextureCoords
      ( f2 0.000 0.000 ) ; index 0
      ( f2 0.000 1.000 ) ; index 1
      ( f2 1.000 0.000 ) ; index 2
      ( f2 1.000 1.000 ) ; index 3
)
```

3.2.8. (Polygons)チャンク

(Polygons)チャンクはモデルデータを形成する各ポリゴン情報を、(face)チャンクを内部に属して定義をします。

3.2.8.1 (face)チャンクのインデックスセット

(face)チャンクの 1 番目のインデックスはポリゴン面が参照するマテリアルの ID です。マテリアル ID は (Material)チャンクで記述された(material)チャンクの記述順序をインデックス 0 番目から指しており、マテリアル ID とポリゴンを関連付けます。

```
( Polygons
     ( face  0  ( i3 0 1 2 ) ( i3 -1 -1 -1 ) )        ; インデックス 0 番目のマテリアルを適用
     ( face  1  ( i4 3 4 5 6 ) ( i4 3 4 5 6 ) )       ; インデックス 1 番目のマテリアルを適用
)
```

ポリゴンを形成する為の結線情報として、頂点インデックスを 3 つもしくは 4 つ定義します。(i3)チャンクを用いた場合 3 頂点で形成するポリゴンを意味し、(i4)チャンクを用いた場合は 4 頂点で形成するポリゴンの定義となります。※ 5 頂点以上のポリゴンはサポート外です。

```
( Polygons
     ( face 0 ( i3  0 1 2 )  ( i3  -1 -1 -1 ) )       ; 三角形ポリゴン
     ( face 1 ( i4  3 4 5 6 )( i4  3 4 5 6  ) )       ; 四角形ポリゴン
)
```

(i3)又は(i4)に続くインデックスセットには、ポリゴンを形成する頂点インデックスと、その頂点に属するテクスチャ UV の UV 座標インデックスを定義します。
最初のインデックスセットには頂点座標のインデックスを定義します。
それぞれが対応する(Vertices)チャンク内の(coords)チャンク・インデックス ID を定義します。
この定義により各頂点インデックスからポリゴンの結線情報を定義したことになります。

```
( Polygons
     ( face 0 ( i3  0 1 2  ) ( i3  -1 -1 -1 ) )       ; 三角形ポリゴンの結線情報
     ( face 1 ( i4  3 4 5 6 ) ( i4  3 4 5 6  ) )      ; 四角形ポリゴンの結線情報
)
```

2 番目のインデックスセットにはテクスチャの UV 座標を定義します。
このインデックスセットにはそれぞれが対応する(TextureCoords)チャンク内の(f2)チャンク・インデックス ID を定義します。このインデックス ID は 1 番目の頂点インデックスセットと対になっていなければいけません。各インデックスが対になっていないと UV 座標に不具合が生じます。
モデルデータのどのポリゴンにもテクスチャが使用されていない場合や、ある特定のポリゴンにテクスチャが適用されていない場合などは、**–1** をインデックス ID に割り当てます。

```
( Polygons
     ( face 0 ( i3 0 1 2  ) ( i3  -1 -1 -1 ) )       ; テクスチャなしポリゴン
     ( face 1 ( i4 3 4 5 6 ) ( i4  3 4 5 6  ) )       ; テクスチャ適用ポリゴン
)
```

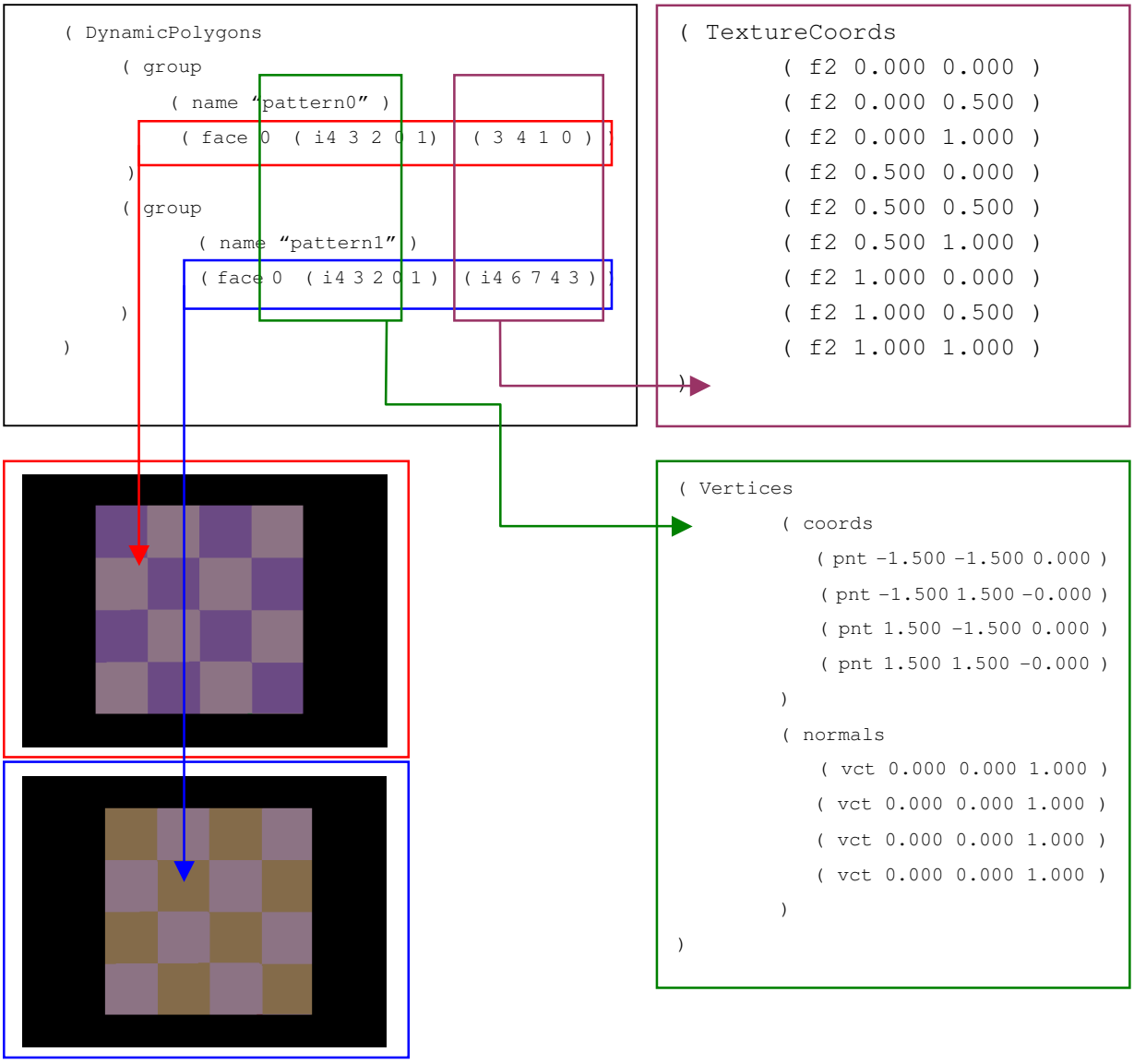
3.2.9. (DynamicPolygons)チャンク

(DynamicPolygons)チャンクではダイナミックポリゴンのパターン定義を行います。
同一頂点を使用するポリゴンにおいて、それらのポリゴン情報は(Polygon)チャンクと(DynamicPolygon)チャンクの両方に定義しないでください。
(DynamicPolygons)チャンクの定義には必ず(TextureCoords)チャンクの定義が必要です。これは、後に解説する(group)チャンクに属する(face)チャンクがテクスチャ座標のインデックスを参照するからです。
ダイナミックポリゴンを使用しないモデルデータの場合は定義する必要はありません。

3.2.9.1 (group)チャンク

(group)チャンクはダイナミックポリゴンのパターンを定義します。(group)チャンクには上から順番に、インデックス 0 から始まる ID が定義されます。これにより(group)チャンクの定義数だけパターンが定義されたことになります。このチャンクに属する(name)チャンクには任意でパターン名をつけることができます。文字列は 255 文字(byte)以内で収めてください。また、このチャンクに属するポリゴンの定義には(Polygons)チャンクと同様の手続きで定義を行います。(face)チャンクの詳細については「**3.2.8. (Polygons)チャンク**」を参照してください。

【図】(DynamicPolygons)チャンクの関係



4. サンプルコード

4.1. BAC6 サンプル 1

ここではテクスチャが適用されているモデルデータを出力したコードを記述しています。

【Sample01.bac】

```
;BAC

( Head
	( bacVersion 6.0 )
)
( Figure
	( name "TexturePolygonSample" )
	( Textures
		( i2 256 256 )
	)
	( Colors
		( f3 0.500 0.500 0.500 )
	)
	( Materials
		( material
			( blendMode   normal )
			( doubleFace  true )
			( transparent false )
			( lighting    true )
			( textureIndex 0 )
			( colorIndex  -1 )
			( specular    1.000 )
		)
	)
	( Vertices
		( coords
			( pnt -1.500 0.000 0.000 )
			( pnt -1.500 3.000 0.000 )
			( pnt 1.500 0.000 0.000 )
			( pnt 1.500 3.000 0.000 )
		)
		( normals
			( vct 0.000 0.000 1.000 )
			( vct 0.000 0.000 1.000 )
			( vct 0.000 0.000 1.000 )
			( vct 0.000 0.000 1.000 )
		)
	)
```

```
        ( Bones
                ( bone
                        ( name          "bone" )
                        ( hasChild      false )
                        ( hasBrother    alse )
                        ( translate     0.000 0.000 0.000 )
                        ( rotate        0.000 0.000 1.000 )
                        ( handle        0.000 1.000 0.000 )
                        ( vertexIndices
                              0 1 2 3
                        )
                )
        )
        ( TextureCoords
                ( f2 0.000 0.000 )
                ( f2 0.000 1.000 )
                ( f2 1.000 0.000 )
                ( f2 1.000 1.000 )
        )
         ( Polygons
                ( face 0 ( i4 3 2 0 1 ) ( i4 2 3 1 0 ) )
        )
)
```

4.2. BAC6 サンプル 2

ここではダイナミックポリゴンが適用されているモデルを出力したコードを記述しています。

【Sample02.bac】

```
;BAC

( Head
	( bacVersion 6.0 )
)
( Figure
	( name "DynamicPolygonSample" )
	( Textures
		( i2 128 128 )
	)
	( Colors
		 ( f3 1.000 0.500 0.500 )
	)
	( Materials
		( material
			( blendMode   normal )
			( doubleFace  true )
			( transparent false )
			( lighting    true )
			( textureIndex 0 )
			( colorIndex  -1 )
			( specular    1.000 )
		)
		( material
			( blendMode    normal )
			( doubleFace   true )
			( transparent  false )
			 ( lighting    true )
			( textureIndex -1 )
			( colorIndex   0 )
			( specular     1.000 )
		)
	)
	( Vertices
		( coords
			( pnt -400.000 -200.000 0.000 )
			( pnt -400.000 200.000 -0.000 )
			( pnt 0.000 -200.000 0.000 )
			( pnt 0.000 200.000 -0.000 )
			( pnt 400.000 -200.000 0.000 )
			( pnt 400.000 200.000 -0.000 )
		)
```

```
		( normals
			( vct 0.000 0.000 1.000 )
			( vct 0.000 0.000 1.000 )
			( vct 0.000 0.000 1.000 )
			( vct 0.000 0.000 1.000 )
			( vct 0.000 0.000 1.000 )
			( vct 0.000 0.000 1.000 )
		)
	)
	( Bones
		( bone
			( name		"bone" )
			( hasChild	false )
			( hasBrother	false )
			( translate	0.000 0.000 0.000 )
			( rotate	0.000 0.000 100.000 )
			( handle	0.000 100.000 0.000 )
			( vertexIndices
				0 1 2 3 4 5
			)
		)
	)
	( TextureCoords
		( f2 0.000 0.000 )
		( f2 0.000 0.500 )
		( f2 0.000 1.000 )
		( f2 0.500 0.000 )
		( f2 0.500 0.500 )
		( f2 0.500 1.000 )
		( f2 1.000 0.000 )
		( f2 1.000 0.500 )
		( f2 1.000 1.000 )
	)
	( Polygons
		( face 1 ( i4 3 2 0 1 ) ( i4 -1 -1 -1 -1 ) )
		( face 1 ( i4 3 2 0 1 ) ( i4 -1 -1 -1 -1 ) )
	)
	( DynamicPolygons
		( group
			( name "pattern0" )
			( face 0 ( i4 5 4 2 3 ) ( i4 3 4 1 0 ) )
		)
		( group
			( name "pattern1" )
			( face 0 ( i4 5 4 2 3 ) ( i4 6 7 4 3 ) )
		)
	)
)
```

BAC File Ver.6.0 Data Format Specification

MascotCapsuleV3

日本語版

バージョン 2.0
発行日 2003 年 10 月 8 日
発行者 株式会社 エイチアイ
〒153-0043 東京都目黒区東山 1-4-4 目黒東山ビル 5F
TEL.03-3710-2843 FAX.03-5773-8660
http://www.hicorp.co.jp/