PEX プロトコル仕様書

1991 年　11 月　第 1.0 版 (Version 5.0P 対応)
1992 年　4 月　第 2.0 版 (Version 5.1C 対応)
1992 年　5 月　第 2.1 版 (Version 5.1C2 対応)
1992 年　6 月　第 2.2 版 (Version 5.1P 対応)

富士通株式会社

Copyright (C) 1988,1989,1990,1991,1992 Massachusetts Institute of Technology
All Right reserved.

Permission to use, copy, modify, and distibute this document for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice are retained, and that the name of M.I.T. not be used in advertising or publicity pertaining to this document without specific written prior permission. M.I.T. makes no representations about the suitability of this document or the protocol defined in this document for any purpose. It is provided "as is" without express or implied warranty.

このドキュメントをいかなる目的に対しても，無料で使用，複写，変更，配付することをここに許可する．その場合，上記の著作権表示をすべての複写物に表示すること，著作権表示とこの許可表示をともにそのままに保つこと，このドキュメントについて公表したり広告したりする際に，事前許可なく M.I.T. の名前を使用しないことを守らなければならない．M.I.T. はいかなる目的のためにも，このドキュメントの適合性やプロトコルについて説明することはしない．これは明示的あるいは暗黙的な保証のない，無保証なものとして提供される．

All Rights Reserved, Copyright (C) 富士通株式会社 1992

目次

1 まえがき **6**

1.1 和訳まえがき . . . 6

1.1.1 著作権および責任 . . . 6

1.1.2 原文書 . . . 6

1.1.3 翻訳者 . . . 6

1.1.4 **V5.0P** から **V5.1C** への主な変更点 . . . 7

1.1.5 **V5.1C** から **V5.1C2** への主な変更点 . . . 8

1.1.6 **V5.1C2** から **V5.1P** への主な変更点 . . . 9

1.2 原文書まえがき（謝辞） . . . 10

2 PEX プロトコル仕様 **11**

2.1 リクエスト処理 . . . 11

2.2 プロトコル形式 . . . 11

2.2.1 リクエスト形式 . . . 11

2.2.2 リプライ形式 . . . 11

2.2.3 エラー形式 . . . 11

2.2.4 事象形式 . . . 11

2.3 構文 (Syntax) . . . 12

2.4 名前付けの慣例 . . . 12

2.5 共通型 (common type) . . . 13

2.5.1 LISTofFOO . . . 13

2.5.2 BITMASK と LISTofVALUE . . . 13

2.5.3 浮動小数形式− FLOAT . . . 13

2.5.4 色 (color) . . . 13

2.5.5 要素型 (element type) . . . 13

2.5.6 型 (type) . . . 14

2.5.7 エラー (error) . . . 19

2.6 事象 (event) . . . 20

2.7 パディング (padding) . . . 20

2.8 拡張情報 . . . 21

2.8.1 拡張情報を得る . . . 21

2.8.2 列挙型情報を得る . . . 22

2.8.3 インプリメンテーション依存の定数を得る . . . 32

2.8.4 レンダラ対象のマッチング . . . 35

2.8.5 エスケープ . . . 35

2.8.6 リプライをともなうエスケープ . . . 36

3 出力コマンド **37**

3.1 データ形式 . . . 37

3.2 エラー . . . 37

3.2.1 浮動小数点データ・エラー . . . 37

3.2.2 色型エラー . . . 38

3.2.3 データ長エラー . . . 38

3.2.4 出力コマンド・エラー . . . 38

3.3 出力コマンド概説 . . . 39

4 ルックアップ・テーブル **65**

4.1 ルックアップ・テーブル・リソースの管理 . . . 71

4.1.1 ルックアップ・テーブルの作成 . . . 71

4.1.2 ルックアップ・テーブルの複写 . . . 71

4.1.3 ルックアップ・テーブルの解放 . . . 72
4.2 ルックアップ・テーブルへの問い合わせ . . . 73
4.2.1 ルックアップ・テーブル情報を得る . . . 73
4.2.2 事前定義されたエントリを得る . . . 73
4.2.3 設定されている指標を得る . . . 74
4.2.4 テーブル・エントリを得る . . . 74
4.2.5 複数のテーブル・エントリを得る . . . 75
4.3 ルックアップ・テーブルの更新 . . . 76
4.3.1 テーブル・エントリの代入 . . . 76
4.3.2 テーブル・エントリの削除 . . . 76

5 パイプライン・コンテキスト 77
5.1 パイプライン・コンテキスト・リソースの管理 . . . 86
5.1.1 パイプライン・コンテキストの生成 . . . 86
5.1.2 パイプライン・コンテキストの複写 . . . 86
5.1.3 パイプライン・コンテキストの解放 . . . 86
5.2 パイプライン・コンテキスト・リソースへの問い合わせ . . . 87
5.2.1 パイプライン・コンテキスト属性を得る . . . 87
5.3 パイプライン・コンテキストの変更 . . . 87
5.3.1 パイプライン・コンテキスト属性の変更 . . . 87

6 レンダラ 88
6.1 レンダラ・リソースの管理 . . . 93
6.1.1 レンダラ・リソースの生成 . . . 93
6.1.2 レンダラ・リソースの解放 . . . 93
6.2 レンダラ・リソースの変更 . . . 94
6.2.1 レンダラ・リソースの変更 . . . 94
6.3 レンダラ・リソース属性の問い合わせ . . . 95
6.3.1 レンダラ属性を得る . . . 95
6.3.2 レンダラ属性ダイナミックスを得る . . . 96
6.4 クライアント側探訪 . . . 97
6.4.1 レンダリングの開始 . . . 97
6.4.2 レンダリングの終了 . . . 97
6.4.3 パイプライン属性の退避 . . . 98
6.4.4 パイプライン属性の復元 . . . 98
6.5 出力コマンド実行 . . . 99
6.5.1 出力コマンドによる表示 . . . 99
6.5.2 要素の表示 . . . 99
6.5.3 累積状態 . . . 100
6.5.4 ネットワークの表示 . . . 100

7 レンダラ・ピッキング 101
7.1 単一ピック . . . 103
7.1.1 単一ピックの開始 . . . 103
7.1.2 単一ピックの終了 . . . 104
7.1.3 １つピックする . . . 105
7.2 全ピック . . . 106
7.2.1 全ピックの開始 . . . 106
7.2.2 全ピックの終了 . . . 107
7.2.3 全ピックする . . . 108

8 ストラクチャ **109**
8.1 ストラクチャ・リソース管理 . . . 110
8.1.1 ストラクチャの作成 . . . 110
8.1.2 ストラクチャの複写 . . . 110
8.1.3 ストラクチャの削除 . . . 111
8.2 ストラクチャの問い合わせ . . . 112
8.2.1 ストラクチャ情報を得る . . . 112
8.2.2 ストラクチャ要素情報を得る . . . 113
8.2.3 ネットワーク内のストラクチャの通知 . . . 113
8.2.4 ネットワーク内の先祖ストラクチャの検索 . . . 114
8.2.5 ネットワーク内の子孫ストラクチャの検索 . . . 115
8.2.6 ストラクチャ要素を得る (Fetch) . . . 116
8.3 ストラクチャ・リソース属性の変更 . . . 117
8.3.1 編集モードの設定 . . . 117
8.3.2 要素ポインタの設定 . . . 117
8.3.3 要素ポインタのラベルへの移動 . . . 118
8.3.4 要素の検索 . . . 119
8.4 ストラクチャの編集 . . . 120
8.4.1 要素の退避 . . . 120
8.4.2 ストラクチャ要素の削除 . . . 121
8.4.3 ラベルまでのストラクチャ要素の削除 . . . 121
8.4.4 2ラベル間のストラクチャ要素の削除 . . . 122
8.4.5 ストラクチャ要素の複写 . . . 123
8.4.6 ストラクチャ参照の変更 . . . 124

9 名前セット **125**
9.1 名前セット・リソース管理 . . . 125
9.1.1 名前セットの生成 . . . 125
9.1.2 名前セットの複写 . . . 125
9.1.3 名前セットの解放 . . . 125
9.2 名前セット問い合わせ . . . 126
9.2.1 名前セットを得る . . . 126
9.3 名前セットの更新 . . . 126
9.3.1 名前セットの変更 . . . 126

10 サーチ・コンテキスト **127**
10.1 サーチ・コンテキスト・リソース管理 . . . 129
10.1.1 サーチ・コンテキストの生成 . . . 129
10.1.2 サーチ・コンテキストの複写 . . . 129
10.1.3 サーチ・コンテキストの解放 . . . 129
10.2 サーチ・コンテキストの問い合わせ . . . 130
10.2.1 サーチ・コンテキストを得る . . . 130
10.3 サーチ・コンテキストの更新 . . . 130
10.3.1 サーチ・コンテキストの変更 . . . 130
10.4 ストラクチャ・ネットワーク検索 . . . 131
10.4.1 ネットワーク検索 . . . 131

11 PHIGS ワークステーション **132**
11.1 PHIGS ワークステーション・リソース管理 . . . 133
11.1.1 PHIGS ワークステーションの生成 . . . 133
11.1.2 PHIGS ワークステーションの解放 . . . 134
11.2 PHIGS ワークステーションの問い合わせ . . . 135

11.2.1 PHIGS ワークステーション情報を得る . . . 135
11.2.2 Dynamic を得る . . . 139
11.2.3 視野表現を得る . . . 141
11.3 PHIGS ワークステーションの操作 . . . 142
11.3.1 すべてのストラクチャの再表示 . . . 142
11.3.2 ワークステーションの更新 . . . 142
11.3.3 クリップ領域の再表示 . . . 143
11.3.4 実行遅延動作 . . . 143
11.3.5 視野優先順位の設定 . . . 143
11.3.6 ディスプレイ更新モードの設定 . . . 144
11.3.7 DC から WC への写像 . . . 144
11.3.8 WC から DC への写像 . . . 145
11.4 PHIGS ワークステーションの更新 . . . 146
11.4.1 視野表現の設定 . . . 146
11.4.2 ワークステーション・ウィンドウの設定 . . . 146
11.4.3 ワークステーション・ビューポートの設定 . . . 147
11.4.4 HLHSR モードの設定 . . . 147
11.4.5 バッファ・モードの設定 . . . 148
11.5 ストラクチャのポスト / アンポスト . . . 149
11.5.1 ストラクチャのポスト . . . 149
11.5.2 ストラクチャのアンポスト . . . 149
11.5.3 すべてのストラクチャのアンポスト . . . 149
11.5.4 ポストしている PHIGS ワークステーションを得る . . . 150

12 ワークステーション・ピッキング 151
12.1 ピック装置記述子 . . . 151
12.1.1 ピック装置の記述子を得る . . . 153
12.1.2 ピック装置記述子の変更 . . . 153
12.2 ピックメジャ . . . 154
12.2.1 ピックメジャの生成 . . . 155
12.2.2 ピックメジャの解放 . . . 155
12.2.3 ピックメジャ属性を得る . . . 155
12.2.4 ピックメジャの更新 . . . 156

13 PEX フォント 157
13.1 PEX フォント・リソース管理 . . . 157
13.1.1 PEX フォントのオープン . . . 157
13.1.2 PEX フォントのクローズ . . . 158
13.2 PEX フォントの問い合わせ . . . 159
13.2.1 PEX フォントの問い合わせ . . . 159
13.2.2 PEX フォントのリスト . . . 160
13.2.3 PEX フォント情報のリスト . . . 160
13.2.4 PEX 文字列範囲の問い合わせ . . . 161

A 標準 PEX サブセットの定義 163

B PHIGS/PHIGS-PLUS のための最小サポート 167

C PEX エラーの定義 169

D テーブルの省略値の定義 179

E 登録されている PEX エスケープ **182**
E.1 エコー色の設定 182

F 和訳用語対照表 **183**

1 まえがき

1.1 和訳まえがき

1.1.1 著作権および責任

本仕様書は **PEX Protocol Specification** を富士通株式会社および株式会社富士通研究所が翻訳したものである．本翻訳の著作権は富士通株式会社に属するが，これを無料で使用，複写，変更，配付することは妨げない．その場合，原文書の M.I.T の著作権表示と許可表示，および富士通株式会社の著作権表示をそのままの形で，すべての複写物に表示することを条件とする．

富士通株式会社は本仕様書の内容について，いっさいの責任を放棄する．

1.1.2 原文書

本ドキュメントは以下の英文資料を全訳したものである．また，原文書にはない各バージョン間の主な差異点，和英用語対照表，索引などは独自に作成した．

PEX Protocol Specification

Version 5.0P - X PUBLIC REVIEW DRAFT (14-September-1990)

Version 5.1C - X CONSORTIUM REVIEW DRAFT (4-Feburary-1992)

Version 5.1C2 - SECOND X CONSORTIUM REVIEW DRAFT (20-April-1992)

Version 5.1P - X PUBLIC REVIEW DRAFT (11-May-1992)

1.1.3 翻訳者

翻訳は以下のメンバーでおこなった．

翻訳

富士通株式会社	三橋 博恭
株式会社富士通研究所	駒崎 弘，大倉 和博，原田 裕明

編集・校正

株式会社富士通研究所	原田 裕明

監修

株式会社ＰＦＵ	長谷川 清
株式会社富士通静岡エンジニアリング	増田 雅之
株式会社富士通研究所	青山 達朗
PEX Interoperability Committee Japan	

1.1.4 V5.0P から V5.1C への主な変更点

- **2.1** リクエストの処理 を追加.
- プロトコル型として以下を追加.
  PICK_ALL_STATE
  PICK_DATA
  PICK_ONE_TYPE
  PICK_RECORD
  PICK_ALL_STATE
  MATCH_DRAW_TYPE
  RENDERER_TARGET
  VISUAL_ID
- **2.6** 事象 において，事象 **MAX_HITS_REACHED** の説明を追加.
- 列挙型 *HLHSRMode* に， 6 ZBufferID を追加.
- 列挙型 *Escape* を追加.
- インプリメンテーション依存定数として MaxHitsEventSupported (CARD32) を追加.
- 拡張情報のリクエストについて， **PEXMatchRendererTargets** と **PEXEscape** を追加.
- 出力コマンドとして， **Noop** を追加.
- レンダラ・リソースの構成要素として，以下を追加.
  pick_inclusion
  pick_exclusion
  pick_start_path
  background_color
  clear_I
  clear_Z
  echo_mode
- リクエスト **PEXGetRendererAttributes** で使用するビットマスクに， *PickNameset*， *EchoMode* を追加.
- リクエスト **PEXBeginRendering** のエラーに RendererState を追加.
- リクエスト **PEXRenderElements**， **PEXAccumulateState** を追加.
- リクエスト **PEXGetDynamics** のエラーに Match を追加.
- リクエスト **PEXSetWKSViewport** のエラーに Alloc を追加.
- **12 Renderer Picking** の章を追加.
- 付録 **A** 標準サブセット に「ストラクチャ・レンダリング」のサブセットを追加.
- 付録 **E** 登録されている **PEX** エスケープ に **PEXSetEchoColor** を登録.

Version 5.1 C1 で変更になった箇所には *c1 の印を付けてある.

1.1.5 V5.1C から V5.1C2 への主な変更点

- **Renderer Picking** の章を **12** から **7** へ移動. 説明を大幅に書換え.
- 型として以下のものを追加.
  BITMASK_SHORT
  PICK_ALL_METHOD
  PICK_ONE_METHOD
  PICK_STATUS
- 型 "ENUM_TYPE" に以下を追加.
  Escape
  PickOneMethod
  PickAllMethod
- 列挙型として以下を追加.
  PickOneMethod
  PickAllMethod
- エスケープについてリクエスト **PEXEscapeWithReply** 追加.
- 次のリクエストにエラーとして Match を追加.
  PEXGetEnumeratedTypeInfo
  PEXGetImpDepConstants
  PEXCreateLookupTable
  PEXGetTableInfo
  PEXGetPredefinedEntries
- リクエスト **PEXEndRendering** にエラー RendererState を追加.
- 次のリクエストについて, RendererState エラーを「レンダラが *Rendering* または *Picking* 状態である」に変更.
  PEXBeginRendering
  PEXRenderNetwork
  PEXBeginPickOne
  PEXPickOne
  PEXBeginPickAll
  PEXPickAll

Version 5.1 C2 で変更になった箇所には *c2 の印を付けてある.

1.1.6 V5.1C2 から V5.1P への主な変更点

- **2.5.7** エラー に説明を追加.
- リクエスト **PEXCreatePipelineContext**, **PEXChangePipelineContext** に NameSet エラーを追加.
- リクエスト **PEXRendererDynamics** の説明を変更.

Version 5.1 P で変更になった箇所には *p の印を付けてある.

1.2 原文書まえがき（謝辞）

M.I.T. の X コンソーシアムを代表して，われわれは PEX の設計，決定，インプリメンテーションに関わったすべての方々に感謝したいと思う．

この作業は 1987 年 6 月に MIT で X3D の最初の会議が催された時から始まった．ここで Digital Equipment Corporation と Sun Microsystems, Inc. が，DEC が作成した最初の草案にもとづいてプロトコルを決めることを共同提案した．

この草案にはプロフェッショナルな人たちからの意見を反映させた．われわれは特に最初のドキュメントの著者を覚えている：Randi J. Rost, Jeffrey Friedberg, Jeffrey S. Saltz, Pete Nishimoto, William H. Clifford, Jr（以上すべて DEC 社）．彼らはこの重要な企画に対して先頭を切ってくれた．

われわれは 1987 年 6 月から 1991 年 8 月まで，課題の討議のためにしばしば集まった PEX アーキテクチャ・チームに対して大きな借りを負っている．この PEX アーキテクチャ・チームのメンバーは次の方たちである：Jeffrey Friedberg, John McConnell, Pete Nishimoto, Randi J. Rost, Jeffrey S. Saltz（以上 DEC 社），Dave Gorgen, Tom Gross, Jeff Stevenson（以上 Hewlett-Packard 社），David Plunkett（Solbourne Computer 社），Jan Hardenbergh（Stardent Computer, Inc.），Marty Hess, Jim Van Loo（以上 Sun Microsystem 社）． *c1

われわれの真摯な感謝を Bertram Herzog（Michigan 大）にも捧げる．彼は X3D 組織の議長を務め，PEX プロジェクトが成功するように図られ，公正で時を得た進行は海外からも信頼された．

X3D グループの何人かのメンバーは PEX アーキテクチャ・チームに貴重な意見を提供してくれた．われわれは特に次の方々の貢献を記憶しておく：James Michener（Hewlett-Packard Company），Todd Newmann, Raymond Drewry（以上 DEC 社）．彼らはアーキテクチャ・チームの会議に出席して，PEX の改良にとって必要不可欠であった，批評的な意見を提供してくれた．他に技術討論に積極的に参加してくれた方々もおられる：Sally C. Barry, Andy Vesper, Ray Shapiro（以上 DEC 社）．

われわれはまた初期の PEX のインプリメンタの方たちにも感謝する．彼らの中には PEX プロトコル・ドキュメントの効果的なチェックをしてくれた PEX サンプル・インプリメンテーション・チーム（Sun Microsystems）も含まれる． Marty Hess, Lisa Chabot, Cheryl Huntington, Erwin Hom, Nash Aragam, Tom Gaskins, John Recker, Chris Nicholas らである．

M.I.T. における X コンソーシアムの指導者 Robert Scheifler は，われわれの努力が X ウィンドウ・システムの理念と定義に適合できるように，きわめて貴重な批評とアドバイスをしてくれた． X コンソーシアムの Jim Fulton も討議や技術的討論に貢献してくれた．

最初の PEX ドキュメント編集者であった Randi Rost へは議論のすべてを文書の形にしてくれたことに対する感謝を捧げる． Sally Barry と Jeff Friedberg へは，初期の PEX ドキュメントの多くの中間的文書のレヴューに骨を折ってくれたことへの感謝を捧げる． PEX 5.0 のドキュメント編集者である Sally Barry は PEX 5.0 の仕様を作る間の Jeff Friedberg, Jan Hardenbergh, Randi Rost らの貴重な助力と励ましに感謝する． PEX 5.1 のドキュメント編集者である Paula Womack はプロトコル仕様の完璧なレヴューをしてくれた Ken Garnett（Shographics）に感謝する． *c1

付け加えて PEX のインプリメンテーションのために働いている多くの会社の PEX チームがおこなってくれた注意深いレビューは，本ドキュメントの正確さに役立った． *c1

PEX のいくつかの記述が負っている X11, PHIGS, PHIGS-PLUS 仕様のドキュメントに関して働いている方にも感謝する．

最後にわれわれは名前を明示しなかった方々の貢献に感謝する．とてもたくさんの方の貢献が，PEX が X ウィンドウ・システムに対する有益で長命な拡張となることを助けている．

2 PEX プロトコル仕様

2.1 リクエスト処理

*c1

出力コマンド (output command) を処理するような Workstation の再生成 (regeneration)とリクエスト (request) については，同一の出力ドローアブル (destination drawable) へ並行レンダリングすることについて，アトミックに対応することは保証されない．しかし，各出力コマンドの要素に対しては，アトミックな実行を保証する．（ただし Execute Structure， GSE， GDP 3D， GDP 2D を除く．）またリクエストと出力コマンドは与えられた一つのコネクション (connection)について，順番に実行される．

2.2 プロトコル形式

すべての PEX [1] プロトコル (protocol)形式は X Window システムのプロトコル Version 11 [2] に準拠する．以後，本書ではこれを X11 とよぶことにする． PEX プロトコルは X11 の思想と要求を支持する．

2.2.1 リクエスト形式

各 X11 リクエスト (request)は 4 バイトのヘッダ (header)（8 ビットの主要オペコード (major opcode)， 8 ビットのデータフィールド (data field)， 16 ビットのデータ長フィールド (length field)からなる）を含む．このヘッダに続いて 0 バイト以上のデータがあり，その長さはデータ長フィールドに記述されている． PEX は X11 の独自の拡張であるため， 8 ビットの主要オペコードは PEX 拡張のオペコード (opcode)を含んでいる．データ長フィールドには 4 バイト単位のリクエスト長が含まれる（ヘッダを含む）．データ（または副オペコード (minor opcode)）フィールドに，このリクエストに対する PEX オペコードが含まれる．

2.2.2 リプライ形式

各リプライ (reply)は少なくとも 32 バイトからなる．ヘッダはこの 32 バイトに含まれる．一つのリプライのヘッダは 32 ビットのデータ長フィールド， 16 ビットの連続番号フィールド (sequence number field)， 8 ビットのタイプ・フィールド (type field)からなる． 0 バイト以上のデータ（その長さはデータ長フィールドに記述されている）がこれに続く．リプライの中の不使用部分のバイトは 0 になる保証はない．連続番号フィールドは対応するリクエストの連続番号のうち，下位の重要な 16 ビットからなる．型フィールドは生成されたリプライの型 (type)を定義する．

2.2.3 エラー形式

エラー (error)の報告は 32 バイトの長さである．各エラーは 8 ビットのエラーコード (error code)を含む．このエラーコードは発生した PEX エラーを特定するために使用される．各エラーリプライは主要オペコード（エラーを報告する拡張は主要オペコードによって特定される），副オペコード（エラーを起こした拡張オペコード），失敗したリクエストの連続番号の下位 16 ビットなどを含む．またパケット (packet)がエラーパケットであることを示す 8 ビットの型フィールドが含まれる．一つのエラー中の不使用バイトは 0 である保証はない．

2.2.4 事象形式

事象 (event)は 32 バイトの長さである．各事象は 8 ビットの型コード (type code)を含む．もし事象が **SendEvent** リクエストから発生したならば，このフィールド内の最上位ビットがセットされる．事象コード (event code) 64–127 は拡張のためのリザーブである． core X11 のプロトコルは，拡張によって生成された事象中の事柄を表現するメカニズムは規定していない．

[1] X3D-PEX および PEX は MIT の商標である．

[2] X Window System は MIT の商標である．

2.3 構文 (Syntax)

{... } は変数の集合を示す. [...] は構成要素 (structure component)のリストである. 強調すべき場合, リクエスト名はボールド体で示す (たとえば **PEXCreateStructure**). リクエスト・パラメータはアンダースコア (_) を区切りとして用い, イタリック体で示す (たとえば *item_mask*). 定義された定数, 登録された型ニューモニック (type mnemonic), 頭が大文字の変数, などは区切りのために大文字を使い, イタリック体で示される (たとえば *RGB_Float*). 定義された型はすべて大文字で, アンダースコアを区切りとして, 標準体で示される (たとえば COORD_3D).

リクエストは下記のような形式で記述される.

名前:

PEXSampleRequest

リクエスト:

arg1: type1
argN: typeN

リプライ:

result1: type1
resultM: typeM

エラー:

kind1,.....,kindK

説明:

機能の記述

もしリプライの記述がないときは, リクエストがリプライを持たない (非同期) が, エラーは報告されるかもしれない.

2.4 名前付けの慣例

PEX リクエストは一定の名前付け慣例を用いる. リクエスト名として一般的に使われる名詞は以下に示される.

Create	リソース (resource)のインスタンス (instance)を生成する.
Free	リソースをクライアント (client)からアクセスできないようにし, 他のリソースから参照されていなければ, そのシステム・リソース (メモリなど) を解放する.
Copy	一つのリソースから他の同じ型のリソースに属性 (attribute)をコピーする.
Get	サーバ (server)からクライアントにリソースの属性を返す.
Change	リソースの属性を変更する.
Set	リソースの選択された属性を変更する.
Destroy	リソースの, あるインスタンスとそれに対するすべての参照を消去し, メモリの割当を解除する.
Delete	リソースの中のいくつかの部分を削除する.
Fetch	サーバからクライアントへストラクチャ要素 (structure element)を返す.
Store	クライアントからサーバ内のあるストラクチャ・リソースへストラクチャ要素を送る.
Begin	ある種の初期化を実行する.
End	ある種の停止処理を実行する.

2.5 共通型 (common type)

この章では PEX プロトコル仕様で使用する共通型を定義する.

2.5.1 LISTofFOO

LISTofFOO 形式の型名は型 FOO の要素のリストを意味する. すなわちデータ長フィールドのサイズは変化しうる (FOO と同じサイズである必要はない). リスト内の項目数が簡単に計算できる場合には, 項目数は与えられないかもしれない. PEX プロトコルにおける他のすべての場合 (LISTofVALUE を除く) には, データ長フィールドは明確に与えられる.

2.5.2 BITMASK と LISTofVALUE

BITMASK 型と LISTofVALUE 型はいくらか特殊である. 異なるリクエストが引数に含まれる.

item_mask: BITMASK
item_list: LISTofVALUE

これはクライアントに対して, 多種類の「引数」の部分集合を設定することを許す. *item_mask* はどの引数が供給されるかを指定する. 各引数は特別なビット位置に割り当てられる. BITMASK 表現は定義された引数よりも多くのビットを含むかもしれないが, *item_mask* 内の不使用ビットは 0 でなければならない. さもなければ拡張は Value エラーを起こす. *item_list* はマスクの各ビットに対応した項目からなる. これはマスクの最下位ビットから最上位ビットの順に並ぶ.

2.5.3 浮動小数形式 – FLOAT

PEX プロトコルは異なる浮動小数形式 (floating-point format)を通過させる. すべての浮動小数の引数は FLOAT として設定される. FLOAT はリクエストに付随したフォーマット・ワード (format word)に含まれる浮動小数型として定義されている. さらに MATRIX, VECTOR, COORD などの項目はリクエストに付随したフォーマット・ワードによって設定される浮動小数形式となる.

2.5.4 色 (color)

PEX では, 色は色型 (color type)とその値として扱う. 色型はその色が指標値 (index value)なのか, ある型の直値 (direct value)なのか, を設定する. PEX サーバは指標色 (index color)と, 少なくとも一つの直値型を扱わなければならない. 指標色は色ルックアップ・テーブル (color lookup table)から色を得られるような指標を用いて定義される. 直値色は RGB, HSV, HLS, CIELUV色値などを直接定義する. 登録された直値色形式のリストは「拡張情報」の章に示す. PEX サーバはインプリメンテーション依存の形で直値色を記憶するのは自由である. しかしクライアントから要求されたときに, それらの値をオリジナルな色型に変換できなければならない.

2.5.5 要素型 (element type)

第 3 章は PEX のインプリメントによって認識される出力コマンドの集合を記述している. これらの出力コマンドは 16 ビットの ELEMENT_TYPE によって区別される. この値は出力コマンドの実際の型を定義する 16 ビット符号なし short からなる. 要素型の高位ビットは, 出力コマンドが標準の PEX なのか (高位ビットが 0), 独自拡張コマンドなのかを示すために使われる. サーバは非標準の PEX 出力コマンドを含むストラクチャ要素を生成できることを求められる. しかしそのような出力コマンドの実行は非実行 (no-op) にできる. これら出力コマンドの内容が未知であるため, 浮動小数変換やバイト交換などは, サーバによってサポートされていない非標準出力コマンドのうえでは実行されない. PHIGS スタイルの GSE や GDP の出力コマンドを使うのと異なり, これらの拡張メカニズムは標準 PEX 拡張に対して優雅な形ですべての拡張機能を追加すること, および必要でない出力コマンドを無視するようなインプリメンテーションを許している.

2.5.6 型 (type)

PEX のプロトコル型を以下に示す.

ASF_ATTRIBUTE	:{*MarkerTypeASF, MarkerScaleASF, MarkerColorASF, TextFontIndexASF, TextPrecASF, CharExpansionASF, CharSpacingASF, TextColorASF, LineTypeASF, LineWidthASF, LineColorASF, CurveApproxASF, PolylineInterpASF, InteriorStyleASF, InteriorStyleIndexASF, SurfaceColorASF, SurfaceInterpASF, ReflectionModelASF, ReflectionAttrASF, BFInteriorStyleASF, BFInteriorStyleIndexASF, BFSurfaceColorASF, BFSurfaceInterpASF, BFReflectionModelASF, BFReflectionAttrASF, SurfaceApproxASF, SurfaceEdgesASF, SurfaceEdgeTypeASF, SurfaceEdgeWidthASF, SurfaceEdgeColorASF*}	
ASF_SPECIFIER	:[enables, asfs: BITMASK]	
ASF_VALUE	:{*Bundled, Individual*}	
ATEXT_STYLE	:ENUM_TYPE_INDEX （*ATextStyle* とともに使用する)	
BITMASK	:CARD32	
BITMASK_SHORT	:CARD16	*c2
BOOLEAN	:{*False, True*}	
BUFFER_MODE	:{*Single, Double*}	
CARD8	: 符号なし 8 ビット整数	
CARD16	: 符号なし 16 ビット整数	
CARD32	: 符号なし 32 ビット整数	
CHARACTER	:{CARD8, CARD16, CARD32}	
COLOR	:{TABLE_INDEX, DIRECT_COLOR [3]}	
CLIP_INDICATOR	:{*Clip, NoClip*}	
COLOR_APPROX_MODEL	:ENUM_TYPE_INDEX （*ColorApproxModel* とともに使用する）	
COLOR_APPROX_TYPE	:ENUM_TYPE_INDEX （*ColorApproxType* とともに使用する）	
COLOR_MODEL	:ENUM_TYPE_INDEX （*RenderingColorModel* とともに使用する）	
COLOR_SPECIFIER	:[color_type: COLOR_TYPE, color_value: COLOR]	
COLOR_TYPE	:ENUM_TYPE_INDEX （*ColorType* とともに使用する）	
COMPOSITION	:{*PreConcatenate, PostConcatenate, Replace*}	
CONSTANT_NAME	:CARD16	
CONTOUR	:{*Disjoint, Nested, Intersecting, Unknown*}	
COORD	:{COORD_2D, COORD_3D, COORD_4D}	
COORD_2D	:[x, y: FLOAT]	
COORD_3D	:[x, y, z: FLOAT]	

COORD_4D	:[wx, wy, wz, w: FLOAT]	*c1
COORD_TYPE	:{*Rational, NonRatinal*}	
CULL_MODE	:{*None, BackFaces, FrontFaces*}	
CURVE_APPROX	:[approx_method: CURVE_APPROX_METHOD, tolerance: FLOAT]	
CURVE_APPROX_METHOD	:ENUM_TYPE_INDEX （*CurveApproxMethod* とともに使用する）	
DC_HIT_BOX_DATA	:[pick_position: DEVICE_COORD_2D, pick_distance: FLOAT]	*c1
DEVICE_COORD	:[x, y: INT16, z: FLOAT]	
DEVICE_COORD_2D	:[x, y: INT16]	
DEVICE_RECT	:[xmin, ymin, xmax, ymax: INT16]	
DIRECT_COLOR	:direct color value[3]	
DISPLAY_STATE	:{*NotEmpty, Empty*}	
DISPLAY_UPDATE	:ENUM_TYPE_INDEX （*DisplayUpdateMode* とともに使用する）	
DRAWABLE_ID	:{WINDOW_ID, PIXMAP_ID}	
DYNAMIC_TYPE	:{*IMM, IRG, CBS*}	
ECHO_MODE	:{*NoEcho, Echo, Unecho*}	*c1
EDGE	:OPT_SWITCH	
EDIT_MODE	:{*StructureInsert, StructureReplace*}	
ELEMENT_INFO	:[type: ELEMENT_TYPE, length: CARD16]	
ELEMENT_POS	:[whence: {*Beginning, Current, End*}, offset: INT32]	
ELEMENT_RANGE	:[position1, position2: ELEMENT_POS]	
ELEMENT_REF	:[structure_id: STRUCTURE_ID, offset: CARD32]	
ELEMENT_TYPE	:CARD16	
ENUM_TYPE	:{*MarkerType, ATextStyle, InteriorStyle, HatchStyle, LineType, SurfaceEdgeType, PickDeviceType, PolylineInterpMethod, CurveApproxMethod, ReflectionModel, SurfaceInterpMethod, SurfaceApproxMethod, ModelClipOperator, LightType, ColorType, FloatFormat, HLHSRMode, PromptEchoType, DisplayUpdateMode, ColorApproxType, ColorApproxModel, GDP, GDP3, GSE, TrimCurveApproxMethod, RenderingColorModel, ParametricSurfaceCharacteristics, Escape, PickOneMethod, PickAllMethod*}	*c1 *c2
ENUM_TYPE_INDEX	:INT16	
EXTENT_INFO	:[lower_left:COORD_2D, upper_right:COORD_2D, concatpoint: COORD_2D]	
FACET	:[facet_data: OPT_DATA, vertices: LISTofVERTEX]	
FLOAT	:floating point value[4]	

[3] 登録された色型は「拡張情報」の章を参照のこと.

[4] 登録された浮動小数型は「拡張情報」参照のこと.

FLOAT_FORMAT	:ENUM_TYPE_INDEX （*FloatFormat* とともに使用する）	
FONT_ID	:{PEX_FONT_ID, X11_FONT_ID}	
HALFSPACE	:[point: COORD_3D, vector: VECTOR_3D]	
HALFSPACE_2D	:[point: COORD_2D, vector: VECTOR_2D]	
HATCH_STYLE	:ENUM_TYPE_INDEX （*HatchStyle* とともに使用する）	
HLHSR_MODE	:ENUM_TYPE_INDEX （*HLHSRMode* とともに使用する）	
INT8	: 符号つき 8 ビット整数	
INT16	: 符号つき 16 ビット整数	
INT32	: 符号つき 32 ビット整数	
INTERIOR_STYLE	:ENUM_TYPE_INDEX （*InteriorStyle* とともに使用する）	
ISTRING	:LISTofMONO_ENCODING	
LIGHT_TYPE	:ENUM_TYPE_INDEX （*LightType* とともに使用する）	
LINE_TYPE	:ENUM_TYPE_INDEX （*LineType* とともに使用する）	
LOOKUP_TABLE_ID	:RESOURCE_ID	
MARKER_TYPE	:ENUM_TYPE_INDEX （*MarkerType* とともに使用する）	
MATCH_DRAW_TYPE	:{*DontCare, Window, Pixmap, Buffer*}	*c1
MATRIX	:FLOAT[4][4][5]	
MATRIX_3X3	:FLOAT[3][3][5]	
MONO_ENCODING	:[char_set: CARD16, char_set_width: {*csByte, csShort, csLong*}, encoding_state: CARD8, string: LISTofCHARACTER]	
NAME	:CARD32	
NAME_SET_ID	:RESOURCE_ID	
NAME_SET_PAIR	:[incl: NAME_SET_ID, excl: NAME_SET_ID]	
NPC_SUBVOLUME	:[min: COORD_3D, max: COORD_3D]	
OPERATOR	:ENUM_TYPE_INDEX （*ModelClipOperator* とともに使用する）	
OPT_COLOR	:optional COLOR[6]	
OPT_DATA	:[color: OPT_COLOR, normal: OPT_NORMAL, edge: OPT_SWITCH]	
OPT_NORMAL	:optinal VECTOR_3D[6]	
OPT_SWITCH	:optinal SWITCH[6]	
OUTPUT_CMD	:[element_type: ELEMENT_TYPE, size: CARD16, data:[7]]	
PC_BITMASK	:CARD32[3]	

[5]行列は実際には浮動小数の一次元配列として通過する．たとえば 4 × 4 行列について，x 方向の移動を表現する行列要素は配列の 4 番目の要素で，y 方向移動の係数は 8 番目にあたる．3 × 3 行列も同様である．

[6]リクエストの中に出現してもしなくてもよいパラメータ．しかし，その有無はリクエストの中で先立つパラメータから引用できる．

[7]出力コマンドとして通過できるデータレコードの詳細については第 3 章「出力コマンド」を参照のこと．

PEX_FONT_ID	:RESOURCE_ID	
PHIGS_WKS_ID	:RESOURCE_ID	
PIPELINE_CONTEXT_ID	:RESOURCE_ID	
PICK_ALL_METHOD	:ENUM_TYPE_INDEX （PickAllMethod とともに使用する）	*c2
PICK_ALL_STATE	:{MoreHits, NoMoreHits, MayBeMoreHits}	*c1
PICK_DATA	:{DC_HIT_BOX_DATA, NPC_SUBVOLUME}	*c1
PICK_DEVICE_TYPE	:ENUM_TYPE_INDEX （*PickDeviceType* とともに使用する）	
PICK_ELEMENT_REF	:[s_id: STRUCTURE_ID, offset: CARD32, pickid: CARD32]	
PICK_MEASURE_ID	:RESOURCE_ID	*c1
PICK_ONE_METHOD	:ENUM_TYPE_INDEX （PickOneMethod とともに使用する）	*c2
PICK_RECORD	:[pick_type: PICK_DEVICE_TYPE, hit_box: PICK_DATA]	*c1
PICK_STATUS	:{NoPick, Ok, Aborted}	*c2
PIXMAP_ID	:RESOURCE_ID	
POLYLINE_INTERP	:ENUM_TYPE_INDEX （*PolylineInterpMethod* とともに使用する）	
PROMPT_ECHO_TYPE	:ENUM_TYPE_INDEX （*PromptEchoType* とともに使用する）	
PSC_TYPE	:ENUM_TYPE_INDEX （*ParametricSurfaceCharacteristics* とともに使用する．）	
PSURF_CHAR	:[psc_type: PSC_TYPE, psc_data: LISTofCARD8]	
REFLECTION_ATTR	:[ambient_coef: FLOAT, diffuse_coef: FLOAT, specular_coef: FLOAT, specular_conc: FLOAT, transmission_coef: FLOAT, specular_color: COLOR_SPECIFIER]	
REFLECTION_MODEL	:ENUM_TYPE_INDEX （*ReflectionModel* とともに使用する）	
RENDERER_ID	:RESOURCE_ID	
RENDERER_STATE	:{*Idle, Rendering, Picking*}	*c1
RENDERER_TARGET	:[depth: CARD8, type: MATCH_DRAW_TYPE, visual: VISUAL_ID]	*c1
RESOURCE_ID	:32 ビット識別子	
SEARCH_CONTEXT_ID	:RESOURCE_ID	
SHAPE	:{*Convex, Nonconvex, Complex, Unknown*}	
STRING	:LISTofCARD8	
STRUCTURE_ID	:RESOURCE_ID	
STRUCTURE_INFO	:[id: RESOURCE_ID, priority: FLOAT]	
SURFACE_APPROX	:[approx_method: SURFACE_APPROX_METHOD, u_tolerance, v_tolerance: FLOAT]	
SURFACE_APPROX_METHOD	:ENUM_TYPE_INDEX （*SurfaceApproxMethod* とともに使用する）	

SURFACE_EDGE_TYPE	:ENUM_TYPE_INDEX （*SurfaceEdgeType* とともに使用する）
SURFACE_INTERP	:ENUM_TYPE_INDEX （*SurfaceInterpMethod* とともに使用する）
SWITCH	:{*Off, On*}
TABLE_ENTRY	:{data: [8] }
TABLE_INDEX	:CARD16
TABLE_TYPE	:{*LineBundle, MarkerBundle, TextBundle, InteriorBundle, EdgeBundle, Pattern, TextFont, Color, View, Light, DepthCue, ColorApprox*}
TEXT_ALIGNMENT	:[vertical: TEXT_VALIGNMENT, horizontal: TEXT_HALIGNMENT]
TEXT_HALIGNMENT	:{*HalignNormal, HalignLeft, HalignRight, HalignCenter*}
TEXT_PATH	:{*PathRight, PathLeft, PathUp, PathDown*}
TEXT_PRECISION	:{*String, Char, Stroke*}
TEXT_VALIGNMENT	:{*ValignNormal, ValignTop, ValignCap, ValignHalf, ValignBase, ValignBottom*}
TRIM_CURVE	:[visibility: SWITCH, order: CARD16, type: COORD_TYPE, approx_method: TRIM_CURVE_APPROX_METHOD, tolerance: FLOAT, tmin, tmax: FLOAT, knots: LISTofFLOAT, points: LISTofCOORD]
TRIM_CURVE_APPROX_METHOD	:ENUM_TYPE_INDEX （*TrimCurveApproxMethod* とともに使用する）
TYPE_OR_TABLE_INDEX	:{ENUM_TYPE_INDEX, TABLE_INDEX}
UPDATE_STATE	:{*NotPending, Pending*}
VECTOR_2D	:[x, y: FLOAT]
VECTOR_3D	:[x, y, z: FLOAT]
VERTEX	:[point: COORD_3D, data: OPT_DATA]
VIEWPORT	:[min: DEVICE_COORD, max: DEVICE_COORD, use_drawable: BOOLEAN]
VIEW_REP	:[index: TABLE_INDEX, clip_flags: BITMASK, clip_limits: NPC_SUBVOLUME, orientation: MATRIX, mapping: MATRIX]
VISUAL_ID	:RESOURCE_ID *c1
VISUAL_STATE	:{*Correct, Deferred, Simulated*}
WINDOW_ID	:RESOURCE_ID
WKS_BITMASK	:CARD32[2]
X11_FONT_ID	:RESOURCE_ID

[8] テーブル・エントリとして通過できるデータレコードの詳細は「ルックアップ・テーブル」の章を参照のこと.

2.5.7 エラー (error)

PEX のリソースを変更するようなリクエストの処理中にエラーが発生したときは，リソースのいくつかの部分（あるいはリソース属性のサブセット）が変えられている可能性がある. *p

PEX プロトコルは X11 プロトコルのエラーコード (error code)と同じセットを用いる．追加のエラーコードは PEX 定義エラーとして供給される．次のエラーコードはいくつかの PEX リクエストから返ってくる.

ColorType
: 設定された色型がサポートされていない.

FloatingPointFormat
: 設定された浮動小数点形式がサポートされていない.

Label
: 設定されたラベルがストラクチャ内に存在しない.

LookupTable
: ルックアップ・テーブル (lookup table)に対する値が異常か，定義されたルックアップ・テーブル・リソースの名前でない.

NameSet
: 名前セット (name set)に対する値が異常か，定義された名前セット名でない.

OutputCommand
: 出力コマンドのいくつかのパラメータが異常か，範囲外か，不適切である.

Path
: ストラクチャ・ネットワーク・パス (structure network path)の値が不適切か異常な値を含む.

PEXFont
: PEX フォント (PEX font)に対する値が異常か，定義されたリソース名でない.

PhigsWKS
: PHIGS ワークステーションに対する値が異常か，定義されたリソース名でない.

PickMeasure
: ピックメジャ (pick measure)に対する値が異常か，定義されたリソース名でない.

PipelineContext
: パイプライン・コンテキスト (pipeline context)に対する値が異常か，定義されたリソース名でない.

Renderer
: レンダラ (renderer)に対する値が異常か，定義されたリソース名でない.

RendererState
: **PEXBeginRendering** リクエストを受けた時，レンダラがレンダリング (Rendering)状態であった.

SearchContext
: サーチ・コンテキスト (search context)に対する値が異常か，定義されたリソース名でない.

Structure
: ストラクチャに対する値が異常か，定義されたリソース名でない.

2.6 事象 (event)

*c1

すべての PEX の事象は X の事象と同じ機構を使う.

PEX は新しい事象 **MAX_HITS_REACHED** を定義する. **PEXGetImpDepConstants** はこの事象がサポートされるか否かを決定するために使用される. この事象はクライアント側がピック探訪 (pick traversal) して記録可能な hits の最大数に到達したことを示す. この事象はレンダラの識別子を含む. クライアントはこの事象を受け取ると, ピック探訪を停止して追加パスが必要かどうか決定しなければならない.

2.7 パディング (padding)

2 ないし 4 バイト境界に配置される値は, いくつかのリクエストにおいてパッド・バイトの挿入を必要とする. パッド・バイトの値は未定である.

2.8 拡張情報

ここでのリクエストは PEX 拡張についての静的な情報とサポートしている事柄について返答を受ける．設定された能力とトレードオフに関する情報は，個々の PEX サーバのインプリメンテーションを記述したドキュメントの中にあるはずである．（たとえば最上の HLHSR 技法，浮動小数形式，直値色形式は何か，など．）

2.8.1 拡張情報を得る

名前:

PEXGetExtensionInfo

リクエスト:

client_protocol_major_version: CARD16
client_protocol_minor_version: CARD16

リプライ:

protocol_major_version: CARD16
protocol_minor_version: CARD16
vender: STRING
release_number: CARD32
subset_info: CARD32

エラー:

None

説明:

client_protocol_major_version と *client_protocol_minor_version* はプロトコルのバージョンが何であるかを示す．返ってきたプロトコル・バージョンの番号は PEX 拡張が実際にサポートしているプロトコルを示す．これはクライアントから送ったバージョンとは異なるかもしれない．PEX 拡張は同時に複数のバージョンをサポートすることが可能である（必要条件ではない）．
protocol_major_version と *protocol_minor_version* は，必要とされる PEX プロトコルの将来の改版のための機構である．一般に，メジャーな版は非互換な改版で，マイナーな版は小さな互換範囲での改版である．このドキュメントに定義されたプロトコルをサポートするサーバは *protocol_major_version* が 5 で，
protocol_minor_version が 1 を返すだろう．ベンダー (*vender*) ・パラメータは ISO-LATAN1 文字の文字列で ある．これは PEX 拡張を供給したベンダーを表している．リリース番号 (*release_number*)は 32 ビット値で，その意味はベンダーに依存している．*subset_info* の上位 16 ビットはベンダーのためにリザーブされている．下位 16 ビットは，PEX サーバがフルインプリメンテーションか，標準的なサブセットかを示す情報を含んでいる． *c1 *c1 *c1

もし *subset_info* の下位 16 ビットが 0 ならば，拡張は完全な PEX のインプリメンテーションである．もし最下位ビットが 1 ならば，PEX 拡張は「即時レンダリング (immediate rendering)」をサポートしている．もし最下位の次のビットが 1 ならば，「PHIGS ワークステーション (PHIGS workstation)」をサポートしている．もし最下位から 3 番目のビットが 1 ならば，「ストラクチャ・レンダリング (structure rendering)」をサポートしている．もしサーバがサポートしていないようなリクエストを受けたならば，*Request* エラーを返す．各サブセット (subset)の定義については Appendix A を参照.

文字列 "X3D-PEX" は PEX 拡張の存在を示すために，X のリクエスト **ListExtensions** によって返答されなければならない．同じ文字列は X リクエスト **QueryExtension** においてクライアントによって使用される.

2.8.2 列挙型情報を得る

名前:

PEXGetEnumeratedTypeInfo

リクエスト:

drawable_id: DRAWABLE_ID
enum_types: LISTofENUM_TYPE
item_mask: BITMASK

リプライ:

types: LISTofLISTofVALUE

エラー:

Drawable, Match, Value *c2

説明:

このリクエストは *enum_types* によって設定された列挙型 (enumerated type)に関する情報を返す. この列挙型情報について, *drawable_id* によって示されたドローアブル (drawable)と同じルート (root)と深さ (depth)を持つドローアブルに対してサポートされていることものである. *item_mask* は各列挙型を記述するために戻るべきデータを示す. 列挙型の記述子の構成要素（および *item_mask* の対応するビット）は以下のようなものである.

index :ENUM_TYPE_INDEX
mnemonic :STRING

もし *index* ビットのみが *item_mask* の中でセットされていれば, 指標値のリスト（ENUM_TYPE_INDEX 型）は *enum_types* リストの中で設定された各列挙型に対する定義値を返す. もし *mnemonic* ビットのみがセットされていれば, 唯一の記述子（ISO-LATAN1 の文字列）が定義値（STRING型）として返る. もし *index, mnemonic* ビットが両方セットされていれば index/mnemonic 対がリクエストされた各列挙型の定義値として返る. もしどちらのビットもセットされていなければ, *enum_types* の中の各エントリに対してサポートしている型の数を表すカウントのリストを返す.

異なる列挙型と登録された値を以下に掲げる. 各登録値に続いて, 返ってくるニューモニック文字列と簡単な説明がある. 文字列は ISO-LATAN1 文字が用いられる. 文字列は以下の通りに返る. 0 より小さい列挙型値はインプリメンテーション依存である（それらに対するインプリメンテーション・ドキュメントを参照）. 掲げられた値より大きい数は将来のためにリザーブされている.

MarkerType

マーカ型(marker type)はマーカ要素(marker primitive)をレンダリングする際に表示される要素の形状を設定する．登録値は以下の通りである．

1 Dot	"." はマーカ位置に常に最小の表示点として表示される． *marker_scale* 属性は無視.
2 Cross	"+" はマーカ位置に交差がくるように表示される.
3 Asterisk	"*" はマーカ位置に交差がくるように表示される.
4 Cricle	"o" はマーカ位置に中心がくるように表示される.
5 X	"x" はマーカ位置に交差がくるように表示される.

ATextStyle

注釈文字列 (annotation text)の様式は注釈文字列の要素をレンダリングする際に用いる様式を定義する．登録値は以下の通りである．

1 NotConnected	注釈文字列要素は参照点に結合した線分を持たないものとして表示する.
2 Connected	注釈文字列要素は参照点に結合した線分を伴って表示する．この線分はその時点の線属性集合に依存する.

LineType

線種型 (line type)は折線 (polyline)と曲線 (curve)要素をレンダリングする際の様式を定義する．登録された値は以下の通りである．

1 Solid	実線 (solid)で表示する.
2 Dashed	破線 (dashed line)で表示する.
3 Dotted	点線 (dotted line)で表示する.
4 DashDot	一点鎖線 ("dots and dashes" line)で表示する.

Dashed, *Dotted*, *DashDot* の場合, 折線の開始点や，クリップ (clip)されたセグメント (segment)の開始点や，折線の各端点において，線の表示を再開するか，連続するかは，インプリメンテーションに依存する.

PolylineInterpMethod

折線シェーディング法 (polyline interpolation method)は各端点に設定された色を持つ折線要素をレンダリングする際の様式を定義する．デプス・キューイング (depth-cueing)は折線シェーディング法に無関係に，折線に対する後処理 (post-processing)として適用される．登録値は以下の通りである.

1 None	折線の端点間のシェーディングは実行しない．もし色値が一つの折線セグメントの端点ごとに異なるならば， i 番目の端点と $i+1$ 番目の端点の間にラインを描くのに i 番目の端点の色値を用いるか，あるいはセグメント全体にわたって平均色を計算して用いるかはインプリメンテーションに依存する.
2 Color	折線の端点色（もし存在するならば）が用いられる．各折線にわたる色値は端点上の色値を線形補間 (linearly interpolating)する.

CurveApproxMethod

曲線近似法 (curve approximation method)は NURB 曲線をレンダリングする際の方法を定義する．登録値は以下の通りである.

1 (imp.dep.)	この値はどのインプリメンテーションでもサポートされるが, 以下が異なるかもしれない．それは他の型の 1 つと同じニューモニックと定義を持ったり, 登録された型にはない方法であったりするかもしれない.
2 ConstantBetweenKnots	この方法は節点 (knot)の組の間を等間隔にパラメータ増加させた折線を作る．許容誤差 (tolerance)は曲線の折線を制御する．もし許容誤差が整数でなければ切捨てられ, 整数部のみ使用される．もし *tolerance* が 0 以下であれば, 曲線はパラメータの極値とパラメータ範囲内にある節点において評価される．もし *tolerance* が 0 より大きいならば，曲線はパラメータの極値と，パラメータ範囲内にある節点，および節点の各端点組間の *tolerance* によって設定される位置の数において評価される.

3 WCS_ChordalSize	この方法は世界座標系 (world coordinates)内で各ライン・セグメント（弦）の長さが許容誤差より小さくなるまで曲線を折線化する.
4 NPC_ChordalSize	この方法は正規化投影座標系 (normalized project coordinates)内で各ライン・セグメント（弦）の長さが許容誤差より小さくなるまで曲線を折線化する.
5 DC_ChordalSize	この方法は装置座標系 (device coordinates)内で各ライン・セグメント（弦）の長さが許容誤差より小さくなるまで曲線を折線化する.
6 WCS_ChordalDev	この方法は世界座標系内でラインと曲線間の最大偏差が許容誤差より小さくなるまで曲線を折線化する.
7 NPC_ChordalDev	この方法は正規化投影座標系内でラインと曲線間の最大偏差が許容誤差より小さくなるまで曲線を折線化する.
8 DC_ChordalDev	この方法は装置座標系内でラインと曲線間の最大偏差が許容誤差より小さくなるまで曲線を折線化する.
9 WCS_Relative	この方法は世界座標系内のスケールに独立した許容誤差に基づいた相対的レベルを維持する.
10 NPC_Relative	この方法は正規化投影座標系内のスケールに独立した許容誤差に基づいた相対的レベルを維持する.
11 DC_Relative	この方法は装置座標系内のスケールに独立した許容誤差に基づいた相対的レベルを維持する.

InteriorStyle

内部様式 (interior style)は領域要素をレンダリングする際に用いる様式を定義する. 登録値は以下の通りである.

1 Hollow	領域の内部様式はフィル (fill)でないが, 境界は領域色を用いて表示される. もしモデリングの結果, 領域がクリッピングされるならば境界はクリッピングされた領域に沿って表示される.
2 Solid	領域の内部は領域色でフィルされる.
3 Pattern	領域の内部はパターン・テーブル (pattern table)を使ってフィルされる. このパターン・テーブルのエントリは内部様式指標によって定義される.
4 Hatch	領域の内部は領域色と, 内部様式指標によって定義されるハッチ (hatch)様式によってフィルされる.
5 Empty	領域の内部はまったく表示しない.

HatchStyle

ハッチ様式 (hatch style)は内部様式が *Hatch* に設定された時にレンダリングする方法を定義する. 現在は登録されたハッチ様式値はない.

SurfaceEdgeType

外周線型 (surface edge type)は領域 (surface)の外周線 (edge)をレンダリングする際の様式を定義する. 登録値は以下の通り.

1 Solid	実線で表示する.
2 Dashed	破線で表示する.
3 Dotted	点線で表示する.
4 DashDot	一点鎖線で表示する.

Dashed, *Dotted*, *DashDot* の場合, 外周線の開始点（クリッピングされたセグメントの開始点を含む）や外周線の各端点において, 線の表示を再開するか, 連続するかは, インプリメンテーションに依存する.

ReflectionModel

反射モデル (reflection model)はレンダリングの際, シェーディング (shading)を実行する方法を定義する. シェーディングへの入力は *intrinsic color* であり, 出力は *shaded color* である. もしある点で反

射モデルが評価できるような法線(normal)が存在するならば, これが使用される. 一方, 点を含むような面に対して法線が存在するならば, 反射モデルが評価される. もし法線が存在しなければ（もし可能ならば）法線を使わずに反射モデルが評価される. 登録値は以下の通りである.

1 NoShading	シェーディングは実行しない. 領域色 (surface color)は光源 (light source)の照明 (illumination)に影響されない（実質的に shaded color = intrinsic color）.
2 Ambient	照明方程式 (lighting equation)の環境項 (ambient term)のみ用いる. shaded color は環境光源の下で見える intrinsic color となる.
3 Diffuse	照明方程式の環境項および拡散項 (diffuse term)のみ用いる. shaded color は環境光源 (ambient light)に, 各光源からの拡散反射 (diffuse reflection)を加味したように見える intrinsic color となる.
4 Specular	照明方程式の環境項および拡散項および鏡面項 (specular term)を用いる. shaded color は *Diffuse* に鏡面反射 (specular reflection)を加味したように見える.

SurfaceInterpMethod

領域シェーディング法 (surface interpolation method)は領域要素をレンダリングする際に領域内部の色値を計算する方法を定義する. デプス・キューイングは領域シェーディング法とは無関係に後処理として適用される. 登録値は以下の通りである.

1 None	単一の光源から計算される色を領域全体に適用する. 領域内部や外周線にわたるシェーディングはおこなわない.
2 Color	現在の *reflection_model* にしたがって領域の端点において色を計算する. これらの色値は領域の内部と外周線にわたって線形シェーディングされる.
3 DotProduct	照明方程式の内積項 (dot product)を各端点で計算する. これらの内積は線形シェーディングされ, シェーディングは領域内部や領域のエッジにわたる各画素を計算するためにこれらの値を用いる.
4 Normal	側面を横切る法線をシェーディングするように試み, 領域内部や外周線における各画素について, できるだけ正確にシェーディング計算をおこなう.

SurfaceApproxMethod

曲面近似法 (surface approximation method)は NURB 領域要素を表示する方法を定義する. 登録値は以下の通りである.

1 (imp.dep.)	この値は各インプリメンテーションでサポートされるが, 以下が異なるかもしれない. それは他の型の 1 つと同じニューモニックと定義を持ったり, 登録された型にはない方法であったりするかもしれない.
2 ConstantBetweenKnots	この方法は節点の組の間を等間隔なパラメータ増加させた平面を作る. 2 つの許容誤差が 2 次元パラメータ空間においてそれぞれ平面をコントロールする. もし許容誤差が整数でなければ切捨てられ, 整数部のみ使用される. もし *u_tolerance* が 0 以下であれば, 領域は u 方向における u パラメータの極値およびパラメータ範囲内の u 節点において評価される. もし *u_tolerance* が 0 より大きいならば, 領域は u 方向における u パラメータの極値およびパラメータ範囲内の u 節点, および節点間で *u_tolerance* で設定された位置において評価される. *v_tolerance* の値も同様に v 方向における制御をおこなう.
3 WCS_ChordalSize	この方法は世界座標系内で各ライン・セグメント（弦）の u 方向の長さが u の許容誤差より小さくなるまで領域を平面化するものである. v 方向についても同様である.
4 NPC_ChordalSize	この方法は正規化投影座標系内で各ライン・セグメント（弦）の u 方向の長さが u の許容誤差より小さくなるまで領域を平面化するものである. v 方向についても同様である.
5 DC_ChordalSize	この方法は装置座標系内で各ライン・セグメント（弦）の u 方向の長さが u の許容誤差より小さくなるまで領域を平面化するものである. v 方向についても同様である.

6 WCS_PlanarDev　この方法は領域を平面化する．この方法は世界座標系内で平面と曲面間の最大偏差が許容誤差 *u_ tolerance* より小さくなるまで領域を平面化するものである．

7 NPC_PlanarDev　この方法は正規化投影座標系内で平面と領域間の最大偏差が許容誤差 *u_tolerance* より小さくなるまで曲線を平面化するものである．

8 DC_PlanarDev　この方法は装置座標系内で平面と領域間の最大偏差が許容誤差 *u_tolerance* より小さくなるまで曲線を平面化するものである．

9 WCS_Relative　この方法は世界座標内系のスケールに独立した許容誤差 *u_tolerance* に基づいた相対的レベルを維持するものである．

10 NPC_Relative　この方法は正規化座標系内のスケールに独立した許容誤差 *u_tolerance* に基づいた相対的レベルを維持するものである．

11 DC_Relative　この方法は装置座標系内のスケールに独立した許容誤差 *u_tolerance* に基づいた相対的レベルを維持するものである．

TrimCurveApproxMethod

トリム曲線近似法 (trim curve approximation method)はトリム曲線 (trim curve)を持つ NURB 領域要素をレンダリングする際に用いる方法を定義する．登録値は以下の通りである．

1 (imp.dep.)　*TrimCurveApproxMethod* のこの値は各インプリメンテーションでサポートされるが, 以下が異なるかもしれない．それは他の型の 1 つと同じニューモニックと定義を持ったり, 登録された型にはない方法であったりするかもしれない．

2 ConstantBetweenKnots　この方法は節点の組の間を等間隔なパラメータ増加させた折線を作る．許容誤差はトリム曲線の折線化をコントロールする．もし許容誤差が整数でなければ切捨てられ, 整数部のみ使用される．もし *tolerance* が 0 以下であれば，トリム曲線はパラメータの極値およびパラメータ範囲内の節点において評価される．もし *tolerance* が 0 より大きいならば, トリム曲線はパラメータの極値およびパラメータ範囲内の節点, および節点間で *tolerance* で設定された位置において評価される．

ModelClipOperator

モデル・クリップ法 (model clip method)は，現在の複合したモデリング・クリッピング・ボリュームによって設定された半空間 (halfspace)を結合するためのオペレーションを定義する．登録値は以下の通りである．

1 Replace　設定された半空間は現在の複合したモデル・クリッピング・ボリュームを新しいボリュームと置き換えて作られる．

2 Intersection　設定された半空間は新しい複合したモデル・クリッピング・ボリュームを求めるために現在の複合したモデル・クリッピング・ボリュームとの共通部分をとる．

LightType

光源型 (light type)はシェーディングにおける光源の特性を定義する．登録値は以下の通りである．

1 Ambient　光源はすべての領域要素に影響を与える．環境光源は色属性のみを持つ．

2 WCS_Vector　光源は世界座標系において色値と方向ベクトル (direction vector)を持つ．

3 WCS_Point　光源は世界座標系において色値, 位置, および 2 つの減衰係数 (attenuation coefficient)を持つ．

4 WCS_Spot　光源は世界座標系において色値, 位置, 方向ベクトル, 集中指数 (concentration exponent)，2 つの減衰係数，広がり角 (spread angle)を持つ．

ColorType

色型 (color type)は色値の形式を定義する．登録値は以下の通りである．

0 Indexed　一つの色は符号なし 16 ビット整数として処理される．（すなわち TABLE_INDEX 型である．） 整数値は色ルックアップ・テーブルへの指標として扱われる．指標で示された色値の参照は，レンダリングにあたって実際の色が必要となった時点でおこなわれる．

1 RGBFloat　一つの色は 3 つの浮動小数値として処理される．これは red[0-1], green[0-1], blue[0-1] の順である．この形式の色は次の型を持つ．
COLOR_RGB_FLOAT:[r,g,b:FLOAT]

2 CIEFloat　一つの色は 3 つの浮動小数値として処理される．これは u[0-1], v [0-1] （CIELUV 系）と輝度 (luminance)[0-1] である．この形式の色は次の型を持つ．
COLOR_CIE_FLOAT:[u,v,luminace:FLOAT]

3 HSVFloat　一つの色は 3 つの浮動小数値として処理される．これは色相 (hue)[0-1]（赤を 0 としたときの位相角），彩度 (saturation)[0-1]，明度 (value)[0-1] である．この形式の色は次の型を持つ．
COLOR_HSV_FLOAT: [hue,saturation,value:FLOAT]

4 HLSFloat　一つの色は 3 つの浮動小数値として処理される．これは色相 (hue)[0-1]（赤を 0 としたときの位相角），明度 (lightness)[0-1], 彩度 (saturation)[0- 1] である．この形式の色は次の型を持つ．
COLOR_HLS_FLOAT:[hue,lightness,saturation:FLOAT]

5 RGBInt8　一つの色は 4 バイトの単位として処理される．これは red, green, blue の順である．この形式の色は次の型を持つ．
COLOR_RGB_INT8:[r,g,b,pad:CARD8]

6 RGBInt16　一つの色は 8 バイトの単位として処理される．これは red, green, blue の順である．この形式の色は次の型を持つ．
COLOR_RGB_INT16:[R,G,B,pad:CARD16]

FloatFormat

浮動小数形式は浮動小数値の形式を定義する．登録値は以下の通りである．

1 IEEE_754_32　IEEE_754 標準の 32 ビット浮動小数値

2 DEC_F_Floating　DEC_F- 浮動小数値

3 IEEE_754_64　IEEE_754 標準の 64 ビット浮動小数値

4 DEC_D_Floating　DEC_D- 浮動小数値

HLHSRMode

HLHSR モード (HLHSR mode)は隠線／隠面消去をおこなう方法を定義する．登録値は以下の通り．

1 Off　すべての出力要素は処理の順に表示される．隠面を消去しない．

2 ZBuffer　デプスーか z- バッファ (z-buffering)手法を使って各画素の可視性 (visibility)判定をする．z- バッファ手法および Z 値の精度（ビット数）は装置に依存する．この手法は変換データのための領域が不必要であり，一つの画像に要素を順次重ねていくものである．したがって HLHSR 手法は線形オーダである．

3 Painters　出力要素は処理中はバッファされる．flush = *True* によって「レンダリング終了」が起きると, バッファ中の要素は平均の depth に基づいてソートされ, 背景から前景に向かってレンダリングされる．この手法は少ない要素に対してはきわめて高速であるが, 多くの中間データ領域を必要とする．これは全体として結果の正しさは保証しない．なぜならば重畳が循環したり, オブジェクトを貫通したりする場合にはうまくいかないからである．

4 Scanline　出力要素は処理中はバッファされる．flush = *True* によって「レンダリング終了」が起きると, バッファ中の要素はソートされ, スキャンライン (scan line)の順に可視性判定がおこなわれる．この手法は少ない要素に対してはきわめて高速であるが，出力要素のバッファとソーティング部において中間データ領域が必要である．

5 HiddenLineOnly　可視のラインのみが表示される．出力要素はそれらが受けられるものしてバッファされるかもしれない．flush = *True* によって「レンダリング終了」が起きると, バッファ内の要素はソートされ, 隠線消去処理が実行される．

6 ZBufferID　これは *HLHSR_identifier* が探訪の間に z- バッファリングを可能／不可能にするために使われることを除いて，*ZBuffer* と同じである．　*c1

PickDeviceType *c2

ピック装置型(pick device type)はワークステーション・ピッキング(workstation picking)（ピック装置(pick device)とピックメジャ(pick measure)を使う）やレンダラ・ピッキング(renderer picking)を実現する際のピック装置を定義する．もしピックメジャが，登録されたピック装置型の一つを用いて作成されると，ピック装置識別子の *pick_data_rec* 要素は無視され，省略値のプロンプト(prompt)とエコー型(echo type)は *EchoPrimitive* である． *pick_data_rec* 要素は未登録のピック装置型に関連して使用される．

1 DC_HitBox	ピック口径(pick aperature)をピック位置(pick position)とピック距離(pick distance)によって設定する．これらは共に装置座標系(device coordinates)で表現される．ヒット・ボックス(hit box)の形状（矩形，円など）はインプリメンテーションに依存する．ピック距離はヒット・ボックスの半分幅か半径として定義される．
2 NPC_HitVolume	ピック口径は NPC 空間で平行筒(parallelpipe)を示す 2 点によって設定される．

PickOneMethod *c2

単一ピック法(pick one method)はレンダラでの単一ピック探訪でのピック基準を設定するものである．登録値は以下の通りである．

1 Last	最後にピックされた要素を返す．
2 ClosestZ	ピック口径(pick aperature)の中で前方クリップ面(front clipping plane)に最も近い z 値を持つ要素を返す．もし複数の要素が同じ近さにあるときは，これらのどれかが返されるだろう．
3 VisibleAny	ピックされた可視（現在の HLHSR モードを考慮した後の）の要素が返されるだろう．
4 VisibleClosest	可視（現在の HLHSR モードを考慮した後の）で，かつピック位置に最も近い要素を返す．どの要素が最も近いか決定するアルゴリズムはインプリメンテーション依存である．

PickAllMethod *c2

全ピック法(pick all method)はレンダラの全ピック探訪のおけるピック基準を設定する．登録されている値は以下の通り．

1 All	ピック口径に含まれるか，または交差するすべての要素を返す．
2 Visible	ピック口径に含まれるか，または交差する可視の要素を返す．

PromptEchoType

プロンプト・エコー型(prompt echo type)はピッキング操作の間のプロンプトとエコーについて定義する．登録値は以下の通りである．

1 EchoPrimitive	ピックされた要素を少なくとも短時間顕示(highlight)するためにインプリメンテーション依存の技術を使う．
2 EchoStructure	ピックされた要素として同じピック ID を持つ，隣接した要素グループをエコーする．あるいは同じピック ID を持つストラクチャのすべての要素をエコーする．（この型についてどちらの解釈をインプリメントするかは自由である．）
3 EchoNetwork	ピックされた要素を含むポストされたストラクチャ全体をエコーする．

DisplayUpdateMode

表示更新モード(display update mode)は表示されている画像に変化を与えるときの方法を定義する．登録値は以下の通りである．

1 VisualizeEach	変化が生起する通りに見せる． (PHIGS-ASAP)
2 VisualizeEasy	"easy to do"の変化があったときのみ見せる． (PHIGS-WAIT/UWOR) "easy to do"であるような事柄は *IMM* の動的変更(dynamic modification)があるとか，表示画像の再生成なしで更新できるような場合である．このような操作の結果は再表示と同等の効果があるが，完全な再探訪やディスプレイのクリアはない．

3 VisualizeNone	いかなる変化も見せない．(PHIGS-WAIT/NIVE)　変化が生じても，画像は陽にリクエストされないかぎり再生成されない.
4 SimulateSome	シミュレート (simulate)できる程度の簡単な変化は見せ，変化をシュミレートする．(PHIGS-WAIT/UQUM)
5 VisualizeWhenever	すべての変化を見せる．もし再表示が必要ならばサーバの都合において実行する．再表示は見せるためのいくつかの変化を起こす．クライアントはすべての変化を見せることを保証するために，更新のワークステーション・リクエストを発行できる．(PHIGS-ASTI/NIVE)

暗黙の再表示は，表示更新が *VisualizeEach* か *VisualizeWhenever* のいずれかであるときに実行されることを注意すること．もしこのような再表示が発生したならば，ディスプレイはクリアされ，ポスト (post)されたストラクチャ・リストを探訪することによって生成できない出力（たとえば core X からの出力）などは失われる．*VisualizeEasy* , *VisualizeNone* , *SimulateSome* は暗黙の再表示は起こさない.

ColorApproxType

色近似型 (color approximation type)はレンダラがレンダリング・パイプライン (rendering pipeline)の色値を表示可能な画素値に変換する方法を記述する．登録値は以下の通りである.

1 ColorSpace	レンダリング・パイプラインの色値は 3 つの独立した色要素を持つ色に変換される.
2 ColorRange	レンダリング・パイプラインの色値は単一の色指標に変換される.

ここに挙げた型によって，輝度計算とデプス・キューイングによって求めた色値を単一の値（たとえば 8 ビット擬似色ディスプレイに対して）に変換するか，3 値（たとえば 24 ビット直値色のディスプレイに対して）に変換するか，を制御することができる.

ColorApproxModel

色近似モデル (color approximation model)は，レンダリングの色近似フェイズで，色のフィルタリング (filtering)やサンプリング (sampling)をどの空間でおこなうかを記述する．登録値は以下の通りである.

1 RGB	red, green, blue
2 CIE	CIELUV 系の u, v, および輝度
3 HSV	色相, 彩度, 明度
4 HLS	色相, 明度, 彩度
5 YIQ	(NTSC) 輝度 (Y), 位相（広帯域, オレンジ－シアン），矩形波（狭帯域，マゼンタ－緑）

RenderingColorModel

レンダリング色モデル (rendering color model)は，レンダリング・パイプラインにわたる色のシェーディング方法をおこなう色モデルを定義する．反射方程式はレンダリング色モデルによって設定された色空間で記述されなければならない.

1 (imp.dep.)	インプリメンテーション依存の色空間
2 RGB	red, green, blue 色モデル
3 CIE	CIELUV 系の u, v, および輝度の色モデル
5 HSV	色相, 彩度, 明度の色モデル
6 HLS	色相, 明度, 彩度の色モデル

ParametricSurfaceCharacteristics

1 None	現在の領域属性に対して, 追加する領域特性はない.
2 (imp.dep.)	インプリメンテーション依存によって領域の形状を表示する．この手法は領域の前面と後面の区別をしない．表現の仕方は要素属性の適切な組み合わせによって制御される．内部属性にって示された内部レンダリングと表現がどのように関連するかはインプリメンテーション依存である．データレコードはこの型を無視する.

3 IsoparametricCurves　等パラメトリック曲線(isoparametric curve)を領域上に描く．データレコードはパラメータ次元の，それぞれに描くための曲線数と配置を含む．もし配置が *Uniform* ならば，設定された曲線数は領域のパラメータ限界内に均等に配置される．曲線はまたパラメータの極値でも描かれる．もし配置が *NonUniform* ならば, 設定された曲線数は各節点対間で均等に配置される．曲線は節点上でも描かれる．いずれの場合でも等パラメトリック曲線の割当てのみが, トリミング曲線によって定義されるような領域内部において表示される．この手法は領域の前面と後面との区別をしない．折線化と等パラメトリック曲線の表示はそれぞれ表面の近似基準, 折線の属性によって制御される．等パラメトリック曲線は内部様式や裏面内部様式属性によって示される内部レンダリングに追加することによって表示される．等パラメトリック曲線は, 要素のフィルあるいは空洞の内部よりも表示優先度が高いが, 要素の外周線よりも優先度が低い.

4 MC_LevelCurves　レベル曲線 (level curve)を領域上に描く．この曲線は領域と，モデリング座標の座標軸ベクトルに垂直な平面の有限集合との交差に対応する．平面の位置はモデリング座標系の原点 P_0 と方向ベクトル $\vec{V}$ とによって定義される直線に沿った交差点の並びによって設定される.

$$P_i = P_0 + t_i\vec{V}$$

この t_i は交差点を設定するパラメータの並びである．これらは次の範囲にある.

$$-\infty < t_i < \infty$$

P_i は直線に垂直な平面の交差点である． P_0 は モデリング座標系の原点, $\vec{V}$ は モデリング座標系内の方向ベクトル である． i 番目の平面は方向ベクトル $\vec{V}$ に垂直で, 点 P_i で直線と交差する．データレコードは原点 P_0 , 方向ベクトル $\vec{V}$, パラメータ t_i のリストである.
この手法は領域の前面と後面とを区別しない．折線化とレベル曲線の表示は, それぞれ曲面近似基準と折線属性によって制御される．曲線は内部様式や裏面内部様式属性によって示される内部レンダリングに追加することによって表示される．レベル曲線は, 要素のフィルあるいは空洞の内部よりも表示優先度は高いが, 要素の外周線よりも優先度が低い.

5 WC_LevelCurves　レベル曲線を領域上に描く．この曲線は領域と, モデリング座標の座標軸ベクトルに垂直な平面の有限集合との交差に対応する．平面の位置はモデリング座標系の原点 P_0 と方向ベクトル $\vec{V}$ とによって定義される直線に沿った交差点の並びによって設定される.

$$P_i = P_0 + t_i\vec{V}$$

この t_i は交差点を設定するパラメータの並びである．これらは次の範囲にある.

$$-\infty < t_i < \infty$$

P_i は直線に垂直な平面の交差点である． P_0 は 世界座標系の原点, $\vec{V}$ は 世界座標系内の方向 である． i 番目の平面は方向ベクトル $\vec{V}$ に垂直で, 点 P_i で直線と交差する．データレコードは原点 P_0 , 方向ベクトル $\vec{V}$, パラメータ t_i のリストである.
この手法は領域の前面と後面とを区別しない．折線化とレベル曲線の表示は, それぞれ曲面近似基準と折線属性によって制御される．曲線は内部様式や裏面内部様式属性によって示される内部レンダリングに追加することによって表示される．レベル曲線は, 要素のフィルあるいは空洞の内部よりも表示優先度は高いが, 要素の外周線よりも優先度が低い.

GDP

GDP 型はPEX 拡張インプリメンテーションでサポートされる（2 次元）一般化表示要素 (GDPs) を設定する．現在，登録されている値はない． *c2

GDP3

GDP3 型はPEX 拡張インプリメンテーションでサポートされる（3 次元）一般化表示要素 (GDP3s) を設定する．現在，登録されている値はない． *c2

GSE

GSE 型はPEX 拡張インプリメンテーションでサポートされる一般化構造要素 (GSEs) を設定する．現在, 登録されている値はない． *c2

Escape *c1

エスケープ (escape)は， PEX 拡張インプリメンテーションによってサポートされる Escapeリクエストを設定する．登録された Escape は Appendix E に示されている．以下のものがある．

1 SetEchoColor　　レンダラのエコー色 (echo color)を変更する．

2.8.3 インプリメンテーション依存の定数を得る

名前:

PEXGetImpDepConstants

リクエスト:

fp_format: FLOAT_FORMAT
drawable_example: DRAWABLE_ID
names: LISTofCONSTANT_NAME

リプライ:

constants: LISTofVALUE

エラー:

Drawable, Match, Value, FloatingPointFormat *c2

説明:

このリクエストによって，クライアントは PEX サーバ拡張機能におけるインプリメンテーション依存の定数を問い合わせることができる．単一の CARD32 あるいは FLOAT がリクエストされた各値に対して返ってくる．これらの値は *names* の中のリクエスト値に対して *constants* の中に順番に返る．浮動小数値は *fp_format* によって設定された形式で返る．返ってくるインプリメンテーション依存の定数は *drawable_example* と同じ root と depth を持つ drawable に対して使用される値に基づいている．

PEX はすべての PEX 拡張が返すべき標準の定数名の数を決めている．これらの標準定数名は上位ビットが 0 である 16 ビット整数である．追加されたインプリメンテーション依存の定数は，上位ビットが 1 である 16 ビット整数を使った PEX サーバ拡張により定義することができる．標準の定数名は以下の通りである．

NominalLineWidth	CARD32	「標準」のライン幅（画素）.
NumSupportedLineWidths	CARD32	サポートされた線幅の数（値 0 は最小および最大幅の間の, 分数幅を含むすべての線幅をサポートすることを示す）.
MinLineWidth	CARD32	表示できる最も細い線の幅（画素）.
MaxLineWidth	CARD32	表示できる最も太い線の幅（画素）.
NominalEdgeWidth	CARD32	「標準」の外周線の幅（画素）.
NumSupportedEdgeWidths	CARD32	サポートされた外周線幅の数（値 0 は最小および最大幅の間の, 分数幅を含むすべての外周線幅をサポートすることを示す）.
MinEdgeWidth	CARD32	表示できる最も細い外周線の幅（画素）.
MaxEdgeWidth	CARD32	表示できる最も太い外周線の幅（画素）.
NominalMarkerSize	CARD32	「標準」のマーカの寸法（高さあるいは幅，画素）.
NumSupportedMarkerSizes	CARD32	サポートされたマーカ寸法の数（値 0 は最小および最大サイズの間の, 分数幅を含むすべてのマーカ寸法をサポートすることを示す）.
MinMarkerSize	CARD32	表示できる最小のマーカの寸法（高さあるいは幅，画素）（この最小値は常に最小の点として表示されるマーカである *Dot* 型は除外する．）.
MaxMarkerSize	CARD32	表示できる最大のマーカの寸法（高さあるいは幅，画素）（この最大値は常に最小の点として表示されるマーカである *Dot* 型は除外する．）.
ChromaticityRedU	FLOAT	（おそらく最適な）ディスプレイ装置の赤チャネルに対する CIELUV 系の u 色度係数を返す.
ChromaticityRedV	FLOAT	（おそらく最適な）ディスプレイ装置の赤チャネルに対する CIELUV 系の v 色度係数を返す.
LuminanceRed	FLOAT	（おそらく最適な）ディスプレイ装置の赤チャネルに対する CIELUV 系の明度係数を返す.
ChromaticityGreenU	FLOAT	（おそらく最適な）ディスプレイ装置の緑チャネルに対する CIELUV 系の u 色度係数を返す.
ChromaticityGreenV	FLOAT	（おそらく最適な）ディスプレイ装置の緑チャネルに対する CIELUV 系の v 色度係数を返す.
LuminanceGreen	FLOAT	（おそらく最適な）ディスプレイ装置の緑チャネルに対する CIELUV 系の明度係数を返す.
ChromaticityBlueU	FLOAT	（おそらく最適な）ディスプレイ装置の青チャネルに対する CIELUV 系の u 色度係数を返す.
ChromaticityBlueV	FLOAT	（おそらく最適な）ディスプレイ装置の青チャネルに対する CIELUV 系の v 色度係数を返す.
LuminanceBlue	FLOAT	（おそらく最適な）ディスプレイ装置の青チャネルに対する CIELUV 系の明度係数を返す.
ChromaticityWhiteU	FLOAT	（おそらく最適な）ディスプレイ装置の白チャネルに対する CIELUV 系の u 色度係数を返す.
ChromaticityWhiteV	FLOAT	（おそらく最適な）ディスプレイ装置の白チャネルに対する CIELUV 系の v 色度係数を返す.
LuminanceWhite	FLOAT	（おそらく最適な）ディスプレイ装置の白チャネルに対する CIELUV 系の明度係数を返す.

MaxNamesetNames	CARD32	名前セットとして許された名前数の最大値. 名前としては 0 から (*MaxNamesetNames* − 1) の範囲にある.
MaxModelClipPlanes	CARD32	設定しうるモデル・クリッピング平面の最大値.
TransparencySupported	CARD32	透過係数がシェーデングにおいて利用されるかどうかによって, *True* あるいは *False* を返す.
DitheringSupported	CARD32	ディザリング・ヒント (dithering hint)が実際にディザリング (dithering)を実行するかどうかによって, *True* あるいは *False* を返す.
MaxNonAmbientLights	CARD32	一時に利用できる非環境光源の最大数.
MaxNURBOrder	CARD32	サポートする最大の NURB オーダ.
MaxTrimCurveOrder	CARD32	トリム曲線の最大のオーダ.
BestColorApproxValues	CARD32	画素がシフトや加算によって構成できるために, もし色近似乗数が 2 の巾乗である方がパフォーマンスがよいか, あるいはほとんど差がないのならば *ColorApproxAnyValues* の方がよいかを示す定数 *ColorApproxPowersOf2* を返す.
DoubleBufferingSupported	CARD32	サーバがダブル・バッファリング (double-buffering)をサポートするかどうかによって, *True* あるいは *False* を返す.
MaxHitsEventSupported	CARD32	*MaxHitsReached* 事象がサポートされているかどうかによって *True* か *False* を返す.

2.8.4 レンダラ対象のマッチング

*c1

名前:

PEXMatchRendererTargets

リクエスト:

drawable_id: DRAWABLE_ID
depth: CARD8
type: MATCH_DRAW_TYPE
visual: VISUAL_ID
max_targets: CARD32

リプライ:

targets: LISTofRENDERER_TARGETS

エラー:

Drawable, Value

説明:

これはサーバがサポートできる visual, depth, drawables についての情報を返す. *drawable_id* はどの screen をクライアントが対象にしているかをサーバに伝え, *depth*, *visual*, *type* はどのデータが対象かを設定する. *depth* に対して 0 を, *visual* に対しては 0 を, *type* に対しては *DontCare* を設定すると, ワイルドカード (wildcard)となる. これはすべての depth, visual, drawable type にマッチする. *max_targets* はクライアントに返してもらいたい最大の対象数を示す.

2.8.5 エスケープ

*c1

名前:

PEXEscape

リクエスト:

escape_id: CARD32
escape_data: LISTofCARD8

エラー:

Value, *escape-specific*

説明:

このリクエストは登録された PEX のエスケープ, あるいはベンダーが設定したリクエストを送るために使用する. 登録されたエスケープは Appendix E に定義してある. 登録されたエスケープに対して *escape_id* は正の数となる. ベンダー定義のエスケープに対しては, *escape_id* は次のように符号化される. 最上位ビットはセットされ, 続く 15 ビットは *vendor_id* を含み, 下位 16 ビットはベンダーが定義した *request_id* を含む. *vendor_id* は MIT の登録によって一意に割り当てられた値である.

ベンダー定義のエスケープに対して, どのように *escape_data* をエンコード化するか設定しなければならない. たとえばあるベンダーが *escape_data* の最初の値として一つの浮動小数値を常に通過させるように設定したとする. 他のベンダーはある sub-request の数値を通過させるように設定するかもしれない. すべての X リクエストと同様, このリクエストは 4 バイトの倍数でなければならない.

もし *escape_id* がサーバによって識別されないならば, あるいは *escape_id* がリプライを生成するエスケープを指すならば, 一つの Value エラーが生じる. このリクエストによって返る他のすべてのエラーはエスケープ定義 (escape-specific)である. このリクエストはいかなる新しい事象やエラーも生成しない. X 拡張のメカニズムは新しい事象や新しいエラーを生成するような, ベンダー定義リクエストに対して用いられるべきである.

2.8.6 リプライをともなうエスケープ

*c1

名前:

PEXEscapeWithReply

リクエスト:

escape_id: CARD32
escape_data: LISTofCARD8

リプライ:

escape_id: CARD32
escape_reply_data: LISTofCARD8

エラー:

Value, *escape-specific*

説明:

このリクエストはリプライを生成するような，登録済の PEX エスケープ，あるいはベンダー特有のエスケープを発行するのに使われる．サポートされているエスケープは **PEXGetEnumeratedTypeInfo** を用いて決定できる．現時点ではリプライをともなうエスケープは登録されていない．

ベンダーのエスケープについては，*escape_data* と *escape_reply_data* をどのようにエンコード化するかはベンダーの責任である．ベンダーはエンコーディング・ドキュメントにある規則にしたがってエンコードしなければならない．

escape_id のエンコードは **PEXEscape** に記述されている．もし *escape_id* がサーバに認識されない場合，あるいはリプライを生成しないエスケープを指す場合は，Value エラーが返る．このリクエストで返る他のすべてのエラーはエスケープ定義である．このリクエストはいかなる新しい事象やエラーも生成しない．X 拡張のメカニズムは新しい事象や新しいエラーを生成するような，ベンダー定義リクエストに対して用いられるべきである．

3 出力コマンド

この章では出力コマンド (output command)について定義する．出力コマンドはレンダラ (renderer)によって処理可能であるか，もしくはストラクチャ要素 (structure element)として格納可能であるコマンドをいう．おのおのの出力コマンドの記述形式に関しては後述する． 出力コマンドは **PEXRenderOutputCommands** リクエストによりレンダラに直接処理されるためにサーバに送られるか， **PEXStoreElements**リクエストによりストラクチャに格納されるためにサーバに送られるか，またはクライアントからの **PEXFetchElements**リクエストによりストラクチャから展開される．

出力コマンドは常に同様の形態で処理される．レンダラによって直接処理されるか，ストラクチャ探訪 (structure traversal)の一部として処理されるかは問題ではない． **PEXRenderOutputCommands** リクエストによりサーバに送られると，出力コマンドはエラー終了またはデータ・リストが完全に終了するまで処理がおこなわれる．エラー終了した場合，そのコマンドおよびそれに続くすべての出力コマンドは破棄され， *OutputCommand* エラーがクライアントに返される．同様に，もし **PEXStoreElements** リクエストが出力コマンドをサーバに送るために使用されるならば，最初のエラーに対応する出力コマンドとそれに続くすべての出力コマンドは破棄され， *OutputCommand* エラーがクライアントに返される．したがってストラクチャ・リソースには誤ったコマンド要素や不当なパラメータが含まれることは許されない．

OutputCommand エラーは， **PEXRenderOutputCommands** リクエストか **PEXStoreElements** リクエストのどちらかによってのみ発生する． 2つのリクエストに対しておこなわれるチェックは同じものである．出力コマンドがレンダラにおいて実行される際に，サーバによってサポートされていない属性の指定，範囲外のテーブル指標 (table indices)の指定，および未定義なテーブル指標の指定がなされたときには，省略値に置き換えられ，エラーは報告されない．

3.1 データ形式

上述した各リクエストにおいて FLOAT_FORMAT なるパラメータが使用される．この形式指定は，クライアントからサーバにデータを転送する上記のリクエストにおいて，いかなる型の浮動小数点データが含まれるかをサーバに示すために用いられる．上記のリクエストのうち，クライアントへのデータの返還を求めるものに対しては，形式指定はサーバに対して，いかなる型の浮動小数点データをクライアントへ送り返すべきかを指示する．

通常，色値 (color value)は色型 (color type)と（色）値の組の形で渡される． 色型は指標値か直値のいずれであるかを示している．データ付き出力要素（おそらくは多くの色値を含んでいる）の場合，色型は一度だけ宣言され，すべての色値はその指定にしたがわなければならない． PEX サーバは，指標により指示された色および少なくともひとつの直値を取り扱えなければならない．実際の色は，指標を使用することにより，色ルックアップ・テーブル (color lookup table)から得られる． または， RGB, HSV, HLS, CIELUV 等の一定の型をした直値として得られる．直値の型としてあらかじめ登録されているものの一覧は，「拡張機能」の章に記述されている． PEX サーバには任意の装置依存な色直値型を登録することができるが，クライアントにより問い合わせられたときには，登録した直値を元の（クライアントが扱う）型に変換できなくてはならない．

3.2 エラー

PEX サーバに対して，出力コマンドを送出したときに通知されるエラー (error)を以下の節に記述する． PEX サーバが，ある特殊な型の出力コマンドを格納または実行した際に検出するエラーに関しては，「出力コマンド概説」の章に個別に記されている．

3.2.1 浮動小数点データ・エラー

浮動小数点データ型は， **PEXStoreElements**, **PEXRenderOutputCommands** リクエストにおいて，一度だけ指定される． PEX サーバによってサポートされていない型が指定されたときには， *FloatingPointFormat* エラーが通知され，出力コマンドは実行されない．仮に出力コマンドが浮動小数点データを含んでいないときでも同様である．リスト中の出力コマンドはすべて無視される．

3.2.2 色型エラー

通常，色の型は色ごとに指定されるが，いくつかのケースでは出力コマンドごとに指定される．出力コマンドにより，PEX サーバによってサポートされていない型が指定されたときには，*OutputCommand* エラーが通知される．不適な指定がなされる前に発行された出力コマンドは関係なく処理されるが，不適な指定がなされた出力コマンド，もしくは指定の後に発行されたものに関してはすべて無視される．

3.2.3 データ長エラー

出力コマンドが，指定された出力コマンド・リスト長を越えたときには，コマンドの実行は中断され，*OutputCommand* エラーが通知される．不当な指定がなされる前に発行された出力コマンドは実行されるが，指定されたリスト長を越える出力コマンドおよび以降の出力コマンドはすべて無視される．

3.2.4 出力コマンド・エラー

出力コマンド中に不当な数値が指定された場合，出力コマンド・リストの実行は中断され，*OutputCommand* エラーが通知される．不当な指定がなされる前に発行された出力コマンドは実行されるが，不当な指定がなされた出力コマンドおよび以降の出力コマンドに関してはすべて無視される．

サポートされていない数値のすべてが不当というわけではない．いくつかの数値に関しては装置依存であることを許している．（たとえば，線種における負の値など）．一般的に先のような型の数値を含む出力コマンドでは自由な指定が許されている．出力コマンドが実行される際，PEX サーバによりサポートされていない数値が指定されていたときには省略値が用いられ，エラーは報告されない．

2 つ目の数値型グループは，PEX サーバに準拠することにより確定する一定の型を持った数値グループであり，外部から問い合わせることはできない．（たとえば，文字列方向など）．PEX サーバにより，これらの数値が定められた範囲から外れていることが検出された場合，上記の規則にしたがい *OutputCommand* エラーが通知される．

3 つ目の数値型グループは，一定の型を持ってはいるが PEX サーバのインプリメントにはすべて必要ではないものの数値グループである．PEX サーバによりサポートされている数値は外部から問い合わせ可能である．（たとえば，内部様式など）．出力コマンドが実行される際，PEX サーバによってサポートされていない数値は省略値に置き換えられ，エラーは通知されない．

出力コマンドがビット・マスク値を含んでいるときには，そのビット・マスク値に未定義なビット指定がなされていたら，PEX サーバは *OutputCommand* エラーを通知しなればならない．上記のエラーに関しては以下の出力コマンド概説においては記述されていない．

クライアントは出力コマンドによって PEX サーバに単位長の法線を渡す必要はない．単位長ではない法線付きの要素を処理したときの効果はインプリメント依存である

3.3 出力コマンド概説

以下にPEXにおいて提供されている出力コマンド形式一覧を示す．おのおのの出力コマンドはOUTPUT_COMMANDで規定された構造体の形式をしており，出力コマンドを特定する16ビットのオペレーション・コード・フィールド（出力コマンドがストラクチャに格納されているときには，ストラクチャ要素を特定するための識別子として機能する），出力コマンドのサイズを4バイト単位で指定する16ビットのサイズ・フィールド，および出力コマンドに固有なデータ・フィールドからなる．オペレーション・コード・フィールドの高位ビットは出力コマンドがPEXの標準コマンドか（高位ビットは0），非標準コマンドまたは固有のコマンド（高位ビットは1）であるかを示すためにリザーブされている．

Marker type

marker_type: MARKER_TYPE

マーカ種属性(*marker_type*)を変更する．指定されたマーカ種がPEXサーバにサポートされていなかったときには，マーカ種＝3 （*MarkerAsterisk*）が使用される．

任意の整数値がマーカ種への指標として指定可能である．

Marker scale

scale: FLOAT

マーカ寸法属性(*marker_scale*)を変更する．指定された値は標準のマーカ寸法(marker size)と掛け合わされ，サーバがサポートするサイズのうち，求められたサイズに最も近いものが選択される．（**PEXGetImpDepConstants**を参照．）

Marker color index

color: TABLE_INDEX

マーカ色属性(*marker_color*)を変更する．色型を *Indexed* へ設定し， *color* で指定される指標への色値の設定をおこなう．指定された指標が未定義であったときには，色指標1が使用される．色指標1が未定義であったときにはAppendix Dに示された省略値が使用される．

Marker color

color: COLOR_SPECIFIER

マーカ色属性(*marker_color*)を変更する．色型とその型にしたがう色値を設定する． 色型が *Indexed* に設定され，指定された指標が未定義であったきには，色指標1が使用される．色指標1が未定あったときにはAppendix Dに示された省略値が使用される．

Marker bundle index

index: TABLE_INDEX

マーカ束指標属性(*marker_bundle_index*)を変更する．未定義のマーカ束指標が指定されたときには，指標1が省略値として使用される．指標1が未定義であったときにはAppendix Dに示された省略値が使用される．

束指標として0が指定されたときには， *OutputCommand* エラーが通知される． *c2

Text font index

index: TABLE_INDEX

文字列フォント指標属性(*text_font_index*)を変更する．文字列フォント指標は文字列フォント・テーブル(text font table)におけるエントリ（フォント・グループ(font group)）を特定する．文字要素は選ばれたエントリにしたがい，レンダリングされる．未定義のフォント指標が指定されたときには，指標1が省略値として使用される． 指標1が未定義であったときにはAppendix Dに示された省略値が使用される

フォント指標として1より小さな値が指定されたときには, *OutputCommand* エラーが通知される．

Text precision

precision: TEXT_PRECISION

文字列表示精度属性 (*text_precision*)を変更する.

文字列 (text)や注釈文字列 (annotation text)要素が表示されるとき，文字列に含まれる ISTRING で規定されるすべての文字は同じ精度で表示される．このことは文字列フォント指標 (text font index)により指定されたフォント・グループが X/PEX フォントからなっていた，もしくは文字列の ISTRING のいくつかが X フォントであったとき，全体の文字列の表示精度は少なくも *Char* 精度まで落とされなければならないことを示す.

char_set 値の設定が現行のフォント・グループにないときには，すべての文字列は省略値のフォント・グループを用いてレンダリングされる． *char_set* 値の設定が省略値のフォント・グループにないときには，インプリメンテーション依存の方法により文字列の一部が表示される.

Character expansion

expansion: FLOAT

文字拡大率属性 (*char_expansion*)を変更する．指定された文字拡大率の大きさのみが処理の対象となる．指定された文字拡大率は，最小／最大文字拡大率と比較される．これらの数値は選択されたフォント・テーブル・エントリにおけるフォント・グループ (font group)にあるフォント・ファイル (font file)にしたがい，クライアントが X/PEX のいずれのフォントを使用しているかにかかっている．たとえば，クライアントが PEX のフォント・ファイルをオープンしているとき（つまり，拡縮および回転が可能なストローク・フォント (stroke font)のとき）には，任意な文字拡大率を指定することが可能である．文字拡大率が最小文字拡大率より小さいときには，最小文字拡大率が使用され，最大文字拡大率より大きいときには最大文字拡大率が使用される.

Character spacing

spacing: FLOAT

文字間隔属性 (*char_spacing*)を変更する.

エラーおよび省略値は定義されていない.

Text color index

color: TABLE_INDEX

レンダラ・リソース内の文字列色属性 (*text_color*)を変更し，色型属性を *Indexed* に設定する．さらに *color* により指定される指標に色値を設定する．指定された指標が未定義であったならば，指標 1 が用いられる．指標 1 の色値が未定義ならば Appendix D に示された省略値の色値が用いられる.

Text color

color: COLOR_SPECIFIER

レンダラ・リソース内の文字列色属性 (*text_color*)を変更し，色型および色値を指定にしたがい設定する． 色型が *Indexed* に設定され，指定された指標が未定義であったならば，指標 1 が用いられる．指標 1 の色値が未定義ならば Appendix D に示された省略値の色値が用いられる.

Character height

height: FLOAT

文字高属性 (*char_height*)を変更する．指定された文字高 (character height)または計算により求められた幅がサポートされていなかったときには，文字高と幅は，サポートされているものの中から指定されたものに最も近いものが選択される．これらの数値は選択されたフォント・テーブル・エントリにおけるフォント・グループにあるフォント・ファイルにしたがい，クライアントが X/PEX のいずれのフォントを使用しているかにかかっている．たとえば，クライアントが PEX のフォント・ファイルをオープンしているとき（つまり，拡縮および回転が可能なストローク・フォントのとき）には，任意な文字高を指定することが可能である.

Character up vector

up: VECTOR_ 2D

文字上方向ベクトル属性 (*char_up_vector*) を変更する．文字上方向ベクトルが縮退 (degenerate) した（長さ＝０）ときには，＜０，１＞の値が使用される．

Text path

path: TEXT_ PATH

文字列方向属性 (*text_path*) を変更する．

文字列方向として，*PathRight*, *PathLeft*, *PathUp*, *PathDown* 以外のパラメータが与えられたときには *OutputCommand* エラーが通知される．

Text alignment

alignment: TEXT_ALIGNMENT

文字列配置属性 (*text_alignment*) を変更する． 水平位置指定が *HalignNormal*, *HalignLeft*, *HalignCenter*, *HalignRight* のいずれでもないか，垂直位置指定が *ValignNormal*, *ValignTop*, *ValignCap*, *ValignHalf*, *ValignBase*, *ValignBottom* のいずれでもないときには， *OutputCommand* エラーが通知される．

Annotation text height

height: FLOAT

注釈文字列高属性 (*atext_height*) を変更する．指定された注釈文字列高または計算により求められた幅がサポートされていなかったときには，注釈文字列高と幅は，サポートされているものの中から指定されたものに最も近いものが選択される．これらの数値は選択されたフォント・テーブル・エントリにおけるフォント・グループにあるフォント・ファイルにしたがい，クライアントが X/PEX のいずれのフォントを使用しているかにかかっている．たとえば，クライアントが PEX のフォント・ファイルをオープンしているとき（つまり，拡縮および回転が可能なストローク・フォントのとき）には，任意な注釈文字列高を指定することが可能である．

Annotation text up vector

up: VECTOR_2D

注釈文字上方向ベクトル属性 (*atext_up_vector*) を変更する．注釈文字上方向ベクトルが縮退した（長さ＝０）ときには，＜０，１＞の値が使用される．

Annotation text path

path: TEXT_PATH

注釈文字列方向属性 (*atext_path*) を変更する．

path への指定が *PathRight*, *PathLeft*, *PathUp*, *PathDown* のいずれでもないときには *OutputCommand* エラーが返される．

Annotation text alignment

alignment: TEXT_ALIGNMENT

注釈文字列配置属性 (*atext_alignment*) を変更する．

水平方向への指定が *HalignNormal*, *HalignLeft*, *HalignCenter*, *HalignRight* 以外，垂直方向への指定が *ValignNormal*, *ValignTop*, *ValignCap*, *ValignHalf*, *ValignBase*, *ValignBottom* 以外のときには *OutputCommand* エラーが返される．

Annotation text style

index: ATEXT_STYLE

注釈文字列様式属性 (*atext_style*)を変更する．指定された文字列様式が PEX サーバによりサポートされていなかったならば，注釈文字列様式 1 （*ATextNotConnected*）が使用される．

任意の整数値が様式として指定可能である．

Text bundle index

index: TABLE_INDEX

文字列束指標属性 (*text_bundle_index*)を変更する．未定義な文字列束指標が指定されたときには，省略値である束指標 1 が使用される．束指標 1 が未定義であったならば Appendix D に示された省略値が使用される．

束指標値に 0 が指定されたときには, *OutputCommand* エラーが返される． *c2

Line type

line_type: LINE_TYPE

線種属性 (*line_type*)を変更する．指定された線種が PEX サーバによりサポートされていなかったならば，線種 1 (*LineTypeSolid*) が使用される．

任意の整数値が線種として指定可能である．

Line width

width: FLOAT

線幅属性 (*line_width*)を変更する．指定された線幅値は標準の線幅に掛け合わされる （**PEXGetImpDepConstants** 参照）．結果はサーバがサポートしている線幅のうち最も近いものにマッピングされる．

Line color index

color: TABLE_INDEX

線色属性 (*line_color*)を変更する．色型を *Indexed* に設定するとともに *color* より指定される指標に色を設定する．指定された指標値が未定義であったならば，指標 1 が使用される．指標 1 が未定義であったならば Appendix D に示された省略値が使用される．

Line color

color: COLOR_SPECIFIER

線色属性 (*line_color*)を変更する．色型と色値を指定にしたがい，設定する．色型が *Indexed* で，指定された色指標値が未定義であったならば，色指標 1 が使用される．色指標 1 が未定義であったならば Appendix D に示された省略値が使用される．

Curve approximation

approx: CURVE_APPROX

曲線近似属性 (*curve_approx*)を変更する．指定された近似方法が PEX サーバによりサポートされていなかったときには，方法 1 (インプリメンテーション依存) が選択される．

曲線近似法として任意の整数値が指定可能である．

Polyline interpolation method

polyline_interp: POLYLINE_INTERP

折線シェーディング法属性 (*polyline_interp*)を変更する．指定されたシェーディング法が PEX サーバによりサポートされていなかったときには，方法 1 (*PolylineInterpNone*) が選択される．

折線シェーディング法として任意の整数値が指定可能である．

Line bundle index

index: TABLE_INDEX

線束指標属性 (*line_bundle_index*)を変更する．未定義な線束指標が指定されたときには，指標 1 が使用される．指標 1 が未定義であったならば Appendix D に示された省略値が使用される．

線束指標に 0 が指定されたときには， *OutputCommand* エラーが返される． *c2

Surface interior style

interior_style: INTERIOR_STYLE

内部様式属性 (*interior_style*)を変更する．未定義な内部様式が指定されたときには，指標 1 (*InteriorStyleHollow*) が使用される．

内部様式として *InteriorStyleHollow*, *InteriorStyleSolid*, *InteriorStylePattern*, *InteriorStyleHatch*, *InteriorStyleEmpty* 以外が指定されたときには， *OutputCommand* エラーが返される．

Surface interior style index

index: TYPE_OR_TABLE_INDEX

内部様式指標属性 (*interior_style_index*)を変更する．現行の *interior_style* が *InteriorStylePattern* か *InteriorStyleHatch* であるときには，指定された指標は領域要素のレンダリング様式を特定するために用いられる．*InteriorStylePattern* 指定において，指定されたパターン・テーブル指標が未定義であったときには，指標 1 が使用される．[9] *InteriorStyleHatch* 指定において指定された指標はハッチ様式を特定する．指標値は正または負の値である．指定されたハッチ様式が定義されていなかったときには，指標 1 が使用される．指標 1 が未定義であったならば，結果はインプリメンテーション依存となる．

Surface color index

color: TABLE_INDEX

領域色指標 (*surface_color*)を変更する．色型を *Indexed* に設定するとともに *color* で指定される指標位置に色値を格納する． 指定された指標値が未定義であったときには，指標 1 が使用される．指標 1 が定義されていなかったならば Appendix D に示された省略値の色値が用いられる．

Surface color

color: COLOR_SPECIFIER

領域色指標 (*surface_color*) を変更し，色型と色値を *color* の指定にしたがって設定する．色型が *Indexed* で，指定された指標値が未定義であったならば，指標 1 が使用される．指標 1 が未定義であったときには Appendix D に示された省略値の色値が使用される．

Surface reflection attributes

attr: REFLECTION_ATTR

領域反射特性属性 (*reflection_attr*)を変更する．

エラーおよび省略値は定義されていない．

Surface reflection model

reflection_model: REFLECTION_MODEL

反射モデル属性 (*reflection_model*)を変更する．指定された属性が PEX サーバによりサポートされていなかったときには，モデル 1 (*ReflectionNoShading*) が代用される．

反射モデルとして任意の整数値が指定可能である．

[9] PHIGS は *InteriorStylePattern* について 0 より大きな 内部様式指標を持つように要求しているが, PEX では実現が困難であるため, 省略値動作を代わりに設定している．

Surface interpolation method

surface_interp: SURFACE_INTERP

内部シェーディング法属性(*surface_interp*)を変更する．指定された内部シェーディング法が PEX サーバによりサポートされていなかったときには， 方法 1 (*SurfaceInterpNone*) が使用される．

内部シェーディング法として任意の整数値が指定可能である．

Backface Surface interior style

interior_style: INTERIOR_STYLE

裏面内部様式属性(*bf_interior_style*)を変更する．指定された様式がサポートされていなかったときには，様式 1 (*InteriorStyleHollow*) が使用される．

指定された様式が *InteriorStyleHollow*, *InteriorStyleSolid*, *InteriorStylePattern*, *InteriorStyleHatch*, *InteriorStyleEmpty* 以外のときには， *OutputCommand* エラーが返される．

Backface Surface interior style index

index: TYPE_OR_TABLE_INDEX

裏面内部様式指標属性(*bf_interior_style_index*)を変更する．指定された様式が *InteriorStylePattern* または *InteriorStyleHatch* であるときには，指標は裏面領域要素のレンダリング様式を特定する． *InteriorStylePattern* 指定において，指定された様式が未定義であったときには，指標 1 が使用される． *InteriorStyleHatch* 指定において，指標はハッチ型を選択する．指標値は正または負の値が指定可能である．指定されたハッチ型がサポートされていなかったときには，表示結果はインプリメンテーション依存である．

Backface Surface color index

color: TABLE_INDEX

裏面領域色属性(*bf_surface_color*)を変更する．色型を *Indexed* に設定し， *color* で指定される指標位置に色値を格納する．指定された色指標が定義されていなかったときには，色指標 1 が使用される．色指標 1 が未定義であったときには Appendix D に示された省略値の色値が用いられる．

Backface Surface color

color: COLOR_SPECIFIER

裏面領域色属性(*bf_surface_color*)を変更する．色型と色値を指定にしたがって設定する．指定された色型が *Indexed* で，指定された指標が未定義であったならば，指標 1 が使用される．指標 1 が未定義であったときには Appendix D に示された省略値の色値が使用される．

Backface Surface reflection attributes

attr: REFLECTION_ATTR

裏面反射特性属性(*bf_reflection_attr*)を変更する．

エラーおよび省略値は定義されていない．

Backface Surface reflection model

reflection_model: REFLECTION_MODEL

裏面反射モデル属性(*bf_reflection_model*)を変更する．指定された属性が PEX サーバによりサポートされていなかったときには，モデル 1 (*ReflectionNoShading*) が代用される．

反射モデルとして任意の整数値が指定可能である．

Backface Surface interpolation method

surface_interp: SURFACE_INTERP

裏面内部シェーディング法属性 (*bf_surface_interp*)を変更する．指定された内部シェーディング法が PEX サーバによりサポートされていなかったときには， 方法 1 (*SurfaceInterpNone*) が使用される．

内部シェーディング法として任意の整数値が指定可能である．

Surface approximation

approx: SURFACE_APPROX

曲面近似属性 (*surface_approx*)を変更する．指定された曲面近似法が PEX サーバによりサポートされていなかったときには，方法 1 (インプリメンテーション依存) が使用される．

曲面近似法として任意の整数値が指定可能である．

Facet culling mode

culling_mode: CULL_MODE

面識別モード属性 (*culling_mode*)を変更する．指定されたモードが *None, BackFaces, Frontfaces* 以外であったときには *OutputCommand* エラーが返される．

Facet distinguish flag

distinguish: BOOLEAN

面識別フラグ属性 (*distinguish*)を変更する．

指定されたモードが *True, False* 以外であったときには *OutputCommand* エラーが返される． *c1

Pattern size

size: VECTOR_2D

パターン寸法属性 (*pattern_size*)を変更する．指定されたパターン寸法の大きさのみが処理の対象となる．現在の内部様式属性 (*interior_style*) が *InteriorStylePattern* であったならば，指定されたパターン寸法が使用される．寸法を示す要素がひとつでも 0 であったときには，出力コマンドは無視される．

Pattern reference point

point: COORD_2D

パターン参照点属性 (*pattern_ref_pt*)，パターン参照ベクトル 1 (*pattern_ref_vec1*)，パターン参照ベクトル 2 (*pattern_ref_vec2*) を変更する．参照点の Z 座標は 0 に設定される．パターン参照ベクトル 1 は＜1， 0， 0＞，パターン参照ベクトル 2 は＜0， 1， 0＞に設定される．

エラーおよび省略値は定義されていない．

Pattern reference point and vectors

ref_pt: COORD_3D

vector1: VECTOR_3D

vector2: VECTOR_3D

パターン参照点属性 (*pattern_ref_pt*)，パターン参照ベクトル 1 (*pattern_ref_vec1*)，パターン参照ベクトル 2 (*pattern_ref_vec2*) を変更する．パターン・ベクトル (pattern vector)が縮退していたとき（ベクトルのひとつが長さ 0 の指定をなされた，または 2 つのベクトルが平行なとき），出力コマンドは無視される．

Interior bundle index

index: TABLE_INDEX

内部様式束指標属性 (*interior_bundle_index*)を変更する．指定された束指標値が未定義であったときには，束指標 1 が使用される．束指標 1 が未定義であったならば Appendix D に示された省略値の内部様式値が適用される．

指定された束指標値が 0 のとき， *OutputCommand* エラーが返される． *c2

Surface edge flag

onoff: SWITCH

外周線属性 (*surface_edges*) を変更する．

指定された値が *On*, *Off* 以外であったときには， *OutputCommand* が返される．

Surface edge type

edge_type: SURFACE_EDGE_TYPE

外周線種属性 (*surface_edge_type*) を変更する．指定された外周線種が PEX サーバによりサポートされていなかったときには，外周線種 1 (*SurfaceEdgeSolid*) が使用される．

外周線種として任意の整数値が指定可能である．

Surface edge width

width: FLOAT

外周線幅属性 (*surface_edge_width*) を変更する．指定された外周線幅は標準外周線幅 (**PEXGetImpDepConstants** を参照) に掛け合わされる．計算結果を PEX サーバによりサポートされている外周線幅のうち最も近いものにマッピングする．

Surface edge color index

color: TABLE_INDEX

外周線色属性 (*surface_edge_color*) を変更する．色型を *Indexed* に設定し， *color* で指定される色指標位置に色値を設定する．指定された色指標が未定義であったときには，色指標 1 が使用される．色指標 1 が未定義であったときには Appendix D に示された省略値の色値が用いられる．

Surface edge color

color: COLOR_SPECIFIER

外周線色属性 (*surface_edge_color*) を変更する．色型と色値を指定にしたがって設定する．色型が *Indexed* で，指定された色指標値が定義されていなかったときには，色指標 1 が使用される．色指標 1 が未定義であったときには Appendix D に示された省略値の色値が用いられる．

Edge bundle index

index: TABLE_INDEX

外周線束指標属性 (*edge_bundle_index*) を変更する．未定義な束指標が指定されたときには，束指標 1 が使用される．束指標 1 が未定義であったならば Appendix D に示された省略値の外周線属性値が使用される．

指定された束指標値が 0 のとき， *OutputCommand* エラーが返される． *c2

Set individual ASF

attribute: ASF_ATTRIBUTE

source: ASF_VALUE

指定された ASF(aspect source flag) 属性を変更する． *attribute* の値により，以下に示すパイプライン属性のうちの一つが変更される． *c1

marker_type_asf
marker_scale_asf
marker_color_asf
text_font_index_asf
text_prec_asf
char_expansion_asf
char_spacing_asf
text_color_asf
line_type_asf
line_width_asf
line_color_asf
curve_approx_asf
polyline_interp_asf
interior_style_asf
interior_style_index_asf
surface_color_asf
surface_interp_asf
reflection_model_asf
reflection_attr_asf
bf_interior_style_asf
bf_interior_style_index_asf
bf_surface_color_asf
bf_surface_interp_asf
bf_reflection_model_asf
bf_reflection_attr_asf
surface_approx_asf
surface_edges_asf
surface_edge_type_asf
surface_edge_width_asf
surface_edge_color_asf

attribute が正しい ASF を指していない，あるいは *source* が *Bundled* にも *Individual* にも設定されていない場合， *OutputCommand* が返される． *c2

Local transform 3D

comp_type: COMPOSITION
matrix: MATRIX
3 次元ローカル変換行列属性 (*local_transform*)を変更する． *comp_type* が *PreConcatenate* であったときには，指定された行列はローカル・モデル変換行列 (local model transformation matrix)の前方から掛け合わされる． *comp_type* が *PostConcatenate* であったときには，指定された行列はローカル・モデル変換行列の後方から掛け合わされる． *comp_type* が *Replace* であったときには，指定された行列はローカル・モデル変換行列に置き換えられる．

comp_type が *PreConcatenate*, *PostConcatenate*, *Replace* 以外であったときには *OutputCommand* エラーが返される．

Local transform 2D

comp_type: COMPOSITION
matrix: MATRIX_3X3
2 次元ローカル変換行列属性 (*local_transform*) を変更する．本コマンドは，変換行列が 4×4 ではなく 3×3 であることを除けば， *local transform 3D* と同じである．結合（掛け合わせ）がおこなわれる前に，以下の形で表現された 3×3 の行列は，

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & j \end{bmatrix}$$

以下のような 4×4 の行列に拡張される．

$$\begin{bmatrix} a & b & 0 & c \\ d & e & 0 & f \\ 0 & 0 & 1 & 0 \\ g & h & 0 & j \end{bmatrix}$$

comp_type が *PreConcatenate*, *PostConcatenate*, *Replace* 以外であったときには *OutputCommand* エラーが返される．

Global transform 3D

matrix: MATRIX

3次元グローバル変換行列属性 (*global_transform*)を変更する.

エラーおよび省略値は定義されていない.

Global transform 2D

matrix: MATRIX_3X3

2次元グローバル変換行列属性 (*global_transform*) を変更する. 本コマンドは, 変換行列が 4 × 4 ではなく 3 × 3 であることを除けば, "Global transform 3D" と同じである. 結合 (掛け合わせ) がおこなわれる前に, 以下の形で表現された 3 × 3 の行列は,

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & j \end{bmatrix}$$

以下のような 4 × 4 の行列に拡張される.

$$\begin{bmatrix} a & b & 0 & c \\ d & e & 0 & f \\ 0 & 0 & 1 & 0 \\ g & h & 0 & j \end{bmatrix}$$

エラーおよび省略値は定義されていない.

Model clip

clip_switch: CLIP_INDICATOR

モデル・クリップ属性 (*model_clip*)を変更する.

モデル・クリップ識別子 (model clip indicator) が *Clip, NoClip* 以外であったときには, *OutputCommand* エラーが返される.

Set model clip volume 3D *c1

operator: OPERATOR

halfspaces: LIST_of_HALFSPACE

3元モデル・クリップ・ボリューム属性 (*model_clip_volume*)モデル・クリップ・ボリューム属性 (*model_clip_volume*) を変更する. おのおのの半空間 (halfspace)は, モデリング空間において一点の座標と法線により定義されている. 法線ベクトルは半空間を与える境界面の法線ベクトルであり, 面のどちら側が有効領域なのかを示している. 点座標は境界面上の点を与える.

それぞれの半空間は現在のモデリング変換 (modeling transformation)によって変換され, *operator* によって指定された操作を用いて, 現在のモデル・クリッピング・ボリュームに結合される. 指定された処理 (operator) が PEX サーバにおいて未サポート, もしくは半空間が縮退したときには, 出力コマンドは無視される.

Set model clip volume 2D *c1

operator: OPERATOR

halfspaces: LIST_of_HALFSPACE_2D

2次元モデル・クリップ・ボリューム属性 (*model_clip_volume*) を変更する. おのおのの半空間 (halfspace)半空間 (halfspace) は, モデリング空間において一点の座標と法線により定義されている. 法線ベクトルは半空間を与える境界面の法線ベクトルであり, 面のどちら側が有効領域なのかを示している. 点座標は境界面上の点を与える. 各点および各ベクトルの z 座標は 0 とみなされる.

それぞれの半空間は現在のモデリング変換 (modeling transformation)によって変換され, *operator* によって指定された操作を用いて, 現在のモデル・クリッピング・ボリュームに結合される. 指定された処理 (operator) が PEX サーバにおいて未サポート, もしくは半空間が縮退したときには, 出力コマンドは無視される. *c2

Restore model clip volume

モデル・クリップ・ボリューム属性 (*model_clip_volume*)を変更する．最後のストラクチャ呼び出し (structure invocation)に対するモデル・クリップ・ボリュームを復元 (restore)する．本コマンド以前のストラクチャがモデル・クリップ・ボリュームを指定していなかったときには，省略値のモデル・クリップ・ボリュームが使用される．

エラーおよび省略値は定義されていない．

View index

index: TABLE_INDEX

ビュー指標属性 (*view_index*)を変更する．指定された指標が未定義であったときには，指標 0 が使用される．ビュー指標 0 が未定義であったならば Appendix D に示された省略値が用いられる．

Light source state

enable: LIST_of_CARD16

disable: LIST_of_CARD16

光源状態属性 (*light_state*)を変更する．現在の光源状態属性は，有効リスト (enable list)に記述された指標に対応する光源をアクティブ (「点灯」) にすることで変化する．また，無効リスト (disable list)に記述された指標に対応する光源を非アクティブ (「消灯」) にすることで変化する．有効／無効リストに登録された光源が光源表 (light table)において未定義であったならば，その光源は無視される．

有効／無効リストに登録されている光源が 0 である，または，同一の光源が両方のリストに同時に登録されたときには， *OutputCommand* エラーが返される． *c2

Depth cue index

index: TABLE_INDEX

デプス・キュー指標属性 (*depth_cue_index*)を変更する．指定されたデプス・キュー指標が未定義であったときには，指標 0 が使用される．指標 0 が未定義であったならば Appendix D に示された省略値が使用される．

Pick ID

pickid: CARD32

ピック識別子属性 (*pick_id*)を変更する．

エラーおよび省略値は定義されていない．

HLHSR identifier *c1

id: CARD32

HLHSR- 識別子属性 (*HLHSR_identifier*)を変更する．本出力コマンドは，現在登録されている HLHSR- モードに関して実行される限りにおいて no-op である（ただし *ZBufferId* を除く）．もし *HLHSR_mode* が *ZBufferId* に設定されているならば， 0 は z- バッファリング無効， 1 は有効を意味する．登録されていない HLHSR- モードでは，本コマンドの効果はインプリメンテーション依存である．指定された HLHSR- 識別子が PEX サーバによりサポートされていなかったときは， HLHSR- 識別子の使用方法はインプリメンテーション依存である．

Color approximation index

index: TABLE_INDEX

色近似指標属性 (*color_approx_index*)を変更する．指定された指標値が未定義であったときには，指標 0 が使用される．指標 0 が未定義であったときには Appendix D に示された省略値が使用される．

Rendering color model

model: COLOR_MODEL

レンダリング色モデル属性 (*rdr_color_model*)を変更する．指定されたモデルが PEX サーバにより未定義であったときには，モデル 0 (インプリメンテーション依存) が使用される．

色モデルとして任意の整数値が指定可能である．

Parametric surface characteristics

psr: PSURF_CHAR

パラメトリック (NURBS) 曲面特性属性 (*psurf_char*)を変更する．指定されたタイプが PEX サーバによりサポートされていなかったときには，型 1 (*None*) が使用され，データ・レコードの内容はインプリメンテーション依存となる．

特性型として多くの整数値が指定される． 矛盾した値が指定されたときには， *OutputCommand* エラーが通知される． （たとえば，型が *IsoparametricCurves* であるときに，配置に *Uniform* でも *Nonuniform* でもない設定がされた．もしくは *MC_LevelCurves* または *WC_LevelCurves* のときに，方向ベクトルが長さ 0 であったときなど．）

Add names to name set

names: LIST_of_NAME

レンダラ・リソースの名前セット属性に指定された名前を追加する．指定された名前がサポートされた範囲（**PEXGetImpDepConstants** 参照）を越えたときには，その名前は無視される．[10]

Remove names from name set

names: LIST_of_NAME

レンダラ・リソースの名前セット属性から指定された名前を削除する．指定された名前がサポートされた範囲（**PEXGetImpDepConstants** 参照）を越えたときにはその名前は無視される．[11]

Execute structure

s_id: STRUCTURE_ID

指定されたストラクチャに探訪の制御を移す．出力コマンドの実行は， *s_id* で指定されるストラクチャの先頭要素から始められる．呼び出されたストラクチャ内の要素がすべて処理されたら，探訪制御は元に戻される．本コマンドが実行されたとき，レンダラ・リソース内のすべてのパイプライン属性値が退避 (save)される．その時点でのグローバルなモデリング変換行列は現結合変換行列として設定され，ローカルなモデリング変換行列は単位行列に設定される．制御が戻されたら，その時点で退避されたレンダラ・リソースのパイプライン属性値が復元 (restore)される．

s_id が存在しないストラクチャを指示していたときには， **OutputCommand** エラーが返される．[12] 存在しないストラクチャ参照によるエラーは起こり得ない．なぜならば **PEXDestroyStructure** により，削除されるストラクチャへの参照はすべて取り除かれるからである．

Label

label: INT32

出力コマンドとしての機能はない（"no-op" 扱い）．ストラクチャを編集するときの指示子となるラベルを管理する．

エラーおよび省略値は定義されていない．

---

[10] PHIGS は Add Names To Set に対してエラーを定義していない．

[11] PHIGS は Remove Names From Set に対してエラーを定義していない．

[12] PHIGS はもし設定されたストラクチャが存在しなければ，新しく空 (empty)のストラクチャを生成する．PEX では存在しないストラクチャ・リソース識別子への参照が起きないようにするのはクライアント側の責任となる．

Application data

data: LISTofCARD8

出力コマンドとしての機能はない（"no-op" 扱い）．指定されたクライアント・アプリケーションのデータを管理する．

エラーおよび省略値は定義されていない．バイト交換 (byte-swapping)や浮動小数点データ変換は本コマンドにおいてはおこなわれない．

GSE

id: INT32

data: LISTofCARD8

インプリメンテーション依存なコマンドである．ネットワーク上で異なる浮動小数点データや色型をサポートするために，GSE が便利な拡張機能として提供されているわけではなく，PHIGS 互換 (PHIGS compatibility)のために提供されている．指定された GSE の ID が PEX サーバによりサポートされていなかったときには，出力コマンドは無視される．バイト交換や，浮動小数点データの変換はこの種の出力コマンドではおこなわれない．

Marker 3D

points: LISTofCOORD_3D

マーカ要素のレンダリングを指示する．マーカは幾何属性として「位置」のみを持つ幾何要素 (geometric primitive)である． *points* は 3D 座標のリストであり，おのおのはモデリング座標におけるマーカの位置を与えている．

レンダリング処理中，マーカの位置情報は装置座標系における位置情報へ変換される．マーカは幾何学的な大きさを持たないため，幾何変換によりマーカの表示サイズは影響を受けない．マーカ色はデプス・キューイング計算によってのみ変化し（光源によるシェーディング計算は領域要素に対してのみ影響する），装置のサポートする色にマッピングされる．マーカの位置はクリップ領域内にあるが，マーカの表示範囲がクリップ領域外にある場合の処理はインプリメンテーション依存である．

マーカ属性の ASF 値を設定することによって， マーカ色 (*marker_color*)，マーカ種 (*marker_type*)，マーカ寸法 (*marker_scale*) を現在のマーカ属性属性テーブルから得るか，もしくはマーカ束指標（ *marker_bundle_index*) によって指示されるマーカ束テーブル (*marker_bundle*) 内のエントリから得るかが選択される．

エラーまたは省略値は定義されていない．

Marker 2D

points: LISTofCOORD_2D

マーカ要素を表示する．本要素はモデリング座標で与えられるマーカ位置がX座標とY座標で与えられ，Z座標は常に 0 と見なされることを除けば，3D マーカ要素と全く同じである．

エラーまたは省略値は定義されていない．

Text 3D

origin: COORD_3D

vector1: VECTOR_3D

vector2: VECTOR_3D

string: ISTRING

文字列要素のレンダリングを指示する． *string*にはレンダリングの対象となる文字列が含まれている． *string* はデータ・レコードの並びであり，各レコードは文字列を構成する文字データと *char_set, char_set_width, encoding_state* のフィールドを持つ． *char_set* フィールドは現フォント・グループに対する指標である．フォント・グループは，1 から始まる整数で番号付けされた，それぞれに独立したフォントを持つ．たとえば番号 3 はその時点で使用されているフォント・グループの 3 番目のフォントを示す．指定された *char_set* の値が現在のフォント・グループにおいてサポートされていなかった場合，文字列は省略値のフォント・グループによっ

て表示される．指定された *char_set* の値が省略値のフォント・グループにおいてサポートされていなかった場合，サポートされていない *char_set* 値が指定された部分に対する表示はインプリメンテーション依存である．*char_set_width* は文字列に含まれる文字データが，8-, 16-, 32- ビットのいずれであるかを示す．必要であれば，文字列中の文字データは PEX サーバによってバイト交換がおこなわれる． *encoding_state* フィールドはクライアントによって使用されるフィールドであり，サーバは処理の対象としない．

文字列は位置と方向ベクトルにより平面上に配置される． 文字列の基点 (origin) は *origin* によりモデリング座標において与えられる．また，文字列の方向は，方向ベクトル *vector1*, *vector2* により規定される． *vector1* は文字列ローカル座標のX軸を規定し， *vector2* はY軸を規定する．

文字列属性 ASF の設定値にしたがって， *text_color*, *text_precision*, *char_expansion*, *char_spacing*, *text_font_index* を個別のテーブルから得るか，文字列束テーブルの *text_bundle_index* が指示するエントリから得るかが決定される． 同じ *char_height*, *char_up_vector*, *text_path*, *text_alignment* 属性も文字列要素表示に使用される． 文字列の表示では，現在の *text_font_index* により指定されるフォント・グループをできる限りの精度で表示する． *char_up_vector*, *text_path* によって規定される文字列方向は文字列ローカル座標系におけるものである．

レンダリング処理中，文字列位置は装置座標系へ変換される．文字列の色はデプス・キューイング計算によってのみ変化し（光源によるシェーディングは領域要素にのみ影響を与える），装置がサポートする色に変換される．文字列のクリップはその時点の文字列表示精度属性 (text precision) にしたがっておこなわれる．文字列表示精度が *String* であるとき，文字列クリップはインプリメント依存である．文字列表示精度が *Char* であるとき，文字列クリップは少なくとも一文字単位でおこなわれる．文字列表示精度が *Stroke* であるとき文字列クリップは文字に対しておこなわれる．

文字列方向ベクトルの少なくともどちらか一方の長さが 0 であるか，ベクトルが平行であったときには，省略値として＜1， 0， 0＞，＜0， 1， 0＞が使用される．

Text 2D

origin: COORD_2D
string: ISTRING
文字列要素のレンダリングを指示する．本要素は方向ベクトルを持たないこと，位置がモデリング座標においてX，Y座標が与えられ， Z座標は常に 0 と仮定されていることを除けば， 3 次元文字列要素と変わりはない．

エラーおよび省略値は定義されていない．

Annotation text 3D

origin: COORD_3D
offset: COORD_3D
string: ISTRING
注釈文字列要素のレンダリングを指示する． *string* には実際にレンダリングされる文字列が含まれている．*string* はデータ・レコードの並びで，おのおののレコードは文字列を構成する文字データ・フィールドと共に *char_set*, *char_set_width*, *encoding_state* のフィールドを含んでいる． *char_set* フィールドは現在のフォント・グループへの指標である．フォント・グループは， 1 から始まる整数で番号付けされた，それぞれに独立したフォントを持つ．たとえば番号 3 はその時点で使用されているフォント・グループの 3 番目のフォントを示す．指定された *char_set* の値が現在のフォント・グループにおいてサポートされていなかった場合，文字列は省略値のフォント・グループによって表示される．指定された *char_set* の値が省略値のフォント・グループにおいてサポートされていなかった場合，サポートされていない *char_set* 値が指定された部分に対する表示はインプリメンテーション依存である． *char_set_width* は文字列に含まれる文字データが, 8-, 16-, 32- ビットのいずれであるかを示す．必要であれば，文字列中の文字データは PEX サーバによってバイト交換がおこなわれる． *encoding_state* フィールドはクライアントによって使用されるフィールドであり，サーバは処理の対象としない．

文字列の基点 (origin)は *origin* によりモデリング座標において与えられる． NPC 空間におけるオフセット値は *offset* により与えられる． 注釈文字列が配置される位置は注釈位置 (annotation point) と呼ばれ，変換後の *origin* に対して *offset* を加えることで得られる．注釈位置のZ座標は，注釈文字列が置かれる X-Y 平面を NPC 空間で指定する．

文字列属性 ASF の設定値にしたがって， *text_color, text_precision, char_expansion, char_spacing, text_font_index* を個別のテーブルから得るか，文字列束テーブルの *text_bundle_index* が指示するエントリから得るかが決定される．注釈文字列をレンダリングする際， *atext_height, atext_path, atext_alignment, atext_up_vector, atext_style* 属性が参照される．

注釈文字列の色はデプス・キューイング計算によってのみ変化し（光源によるシェーディングは領域要素にのみ影響を与える），装置がサポートする色に変換される． *origin* がクリップされたときには，注釈文字列全体がクリップされる． *origin* がモデリング (modeling)，ビュー (view)，ワークステーションのいずれのクリップ枠によってもクリップ (clip)されなかったときには，注釈文字列は通常の文字列クリップと同様のルールでクリップされる．さらに引き出し線 (connection line)が与えられているときには，折線のクリップと同じ方法でクリップされる．ただしモデリング・クリップは無視される．引き出し線が表示されるときには，現在の折線表示属性が参照される．

エラーおよび省略値は定義されていない．

Annotation text 2D

origin: COORD_2D
offset: COORD_2D
string: ISTRING
注釈文字列要素のレンダリングを指示する．本要素は基点，オフセットがXおよびY座標のみで与えされ， Z 座標は常に 0 であると仮定されている他は， 3 次元注釈文字列と同じである．

エラーおよび省略値は定義されていない．

Polyline 3D

vertices: LISTofCOORD_3D
折線要素のレンダリングを指示する． *vertices* 内に指定される頂点 (vertices)データにより折線を定義する．頂点データはモデリング座標において与えられる．頂点は線分により結合される．第一の頂点は第二の頂点と，第二の頂点は第三の頂点と結合され，以下同様に繰り返される．最終頂点は第一頂点と結合されない．

折線属性 ASF の設定により， *line_color, line_type, line_width* が現在の線属性から得られるか，線束テーブル内の *line_bundle_index* により指示されるエントリから得るかが決められる．

折線は幾何学的な幅を持たず，長さの概念しか持たないため，座標変換は折線の表示長にしか影響を及ぼさない．折線の色はデプス・キューイング計算によってのみ変化し（光源によるシェーディングは領域要素にのみ影響を与える），装置がサポートする色に変換される．折線はクリップ境界の外側となる部分については表示されない．クリップ領域と交差する折線はクリップされ，領域の内部に存在する部分のみが表示される．

ストラクチャ構築時，折線を定義する頂点数が 2 より少なかったときには，ストラクチャ内に格納はされるが，この出力コマンドが実行される際，視覚的な効果は現れない．即時 (immediate)モードにおいては，上記のような要素は無視される．

Polyline 2D

vertices: LISTofCOORD_2D
折線要素のレンダリングを指示する．本要素はモデリング座標で与えられる位置がX，Y座標のみで与えられ，Z座標は常に 0 と仮定されていることを除けば， 3 次元折線要素と機能の上で何の変わりもない．

ストラクチャ構築時，折線を定義する頂点数が 2 より少なかったときには，ストラクチャ内に格納はされるが，この出力コマンドが実行される際，視覚的な効果は現れない．即時モードにおいては，上記のような要素は無視される．

Polyline set 3D with data

color_type: COLOR_TYPE
vert_attributes: BITMASK_SHORT *c2

vertices: LISTofLISTofVERTEX
複数の折線要素の組のレンダリングを指示する．複数の折線が表示できること，頂点において付加的な情報が指定できることを除けば，3次元折線要素の動作と変わりはない．色値は *color_type* で指定される型で渡される． *vert_attributes* は折線の各頂点で指定される属性を規定する．頂点属性要素へのビット・マスクは以下の並びをしている．

color　　COLOR

属性ビット・マスクのいずれのビットがセットされていたならば，対応する属性は頂点ごとに渡されなければならず，頂点座標データに続けて上記の順で与えられる．

頂点ごとに色値が与えられたとき，その色は要素の一部と見なされ, *line_color* 属性の代わりに用いられる．加えて，頂点ごとの色値データは *polyline_interp* 属性の影響を受ける． *polyline_interp* 属性は， *polyline_interp_asf* 属性が *Individual* に設定されていたときには， *polyline_interp* 属性テーブルから直接得られ，さもなければ，線束テーブルの *line_bundle_index* で指示されるエントリから得られる． *polyline_interp*属性は頂点間の色のシェーディングをいかにおこなうかを規定する．

ストラクチャ作成時，集合内の任意の折線に2未満の頂点しか与えられなかったとき，ストラクチャには格納されるが実行されても何も表示されない．即時モードでは上記の要素は無視される．

Non-uniform B-spline curve

order: CARD16
type: COORD_TYPE
tmin: FLOAT
tmax: FLOAT
knots: LISTofFLOAT
points: LISTofCOORD
NURBS-曲線のレンダリングを指示する． *order*は正の整数が指定される．曲線の形はパラメトリック (u, v) 空間において指定された節点 (knot)，およびモデリング空間において指定される制御点 (control point)により決定される．一般的に，制御点数は次数 (*order*)以上の大きさを持つ． 制御点数に曲線の次数を加えたものは節点数と等しくなければならない．節点の並びは非減少な整数値の並びとならなければならない．

type は曲線が有理 (*Rational*)か非有理 (*NonRational*)かを指定する． *type* が有理であったときには，制御は同次モデリング座標系 (homogeneous modeling coordinates) (COORD_4D) で与えられなければならない．非有理であったときには，非同次モデリング座標 (non-homogenous modeling coordinates)(COORD_3D) で与えられなければならない．

tmin, *tmax* は, NURBS-曲線の有効範囲に相当するパラメタの範囲を指定する． *tmin* は *tmax* より小さくなくてはいけない． *tmin* は次数番目の節点値以上の値を持ち， *tmax* は (k+1-次数) 番目の節点値以下の値を持たなければならない (k：節数).

線属性 ASF の設定にしたがって， *line_color*, *line_type*, *line_width*, *curve_approx* を現在の線属性から得るか， 線束テーブル *line_bundle* の *line_bundle_index* で指示されるエントリから得るかが決定される．

この出力コマンドの発行で，曲線の次数が PEX サーバによりサポートされていなかった場合，表示上は何の効果も持たない．即時モードのとき本要素は無視される．

以下の場合， *OutputCommand* エラーが通知される． *type* が *Rational*/*NonRational* のいずれでもなかった． *order* に1より小さい数値が指定された． 制御点数が *order* より少なかった． order が節点および制御点の数と異なっていた．節点に与えられた数値が非減少ではなかった． *tmin* と *tmax* の値が節点値と矛盾した． *c2 有理制御点の w 座標に0以下の値が指定された．

Fill area 3D

shape: SHAPE
ignore_edges: BOOLEAN

vertices: LISTofCOORD_3D

多角形要素のレンダリングを指示する．多角形要素は頂点の並びによって定義された頂点が順に線分により結合され，ひとつの閉じた平面領域を表現する（多角形は必ずしも平面上にある必要はないが，平面上にないときには違和感のあるシェーディング結果が現れる）．第一頂点は第二頂点と結合され，第二頂点は第三頂点と結合され，以下同様に繰り返される．最終頂点は暗黙のうちに第一頂点と結合される．

レンダリング処理中，頂点座標は装置座標へ変換される．表面の色はシェーディング計算（内部様式および反射モデルを用いる）により求められ，さらにデプス・キューイングにより変換された後，装置がサポートする色にマッピングする．指定されたクリップ領域から外れる部分は表示されない．クリップ境界と交差する多角形は境界線によりクリップされ，境界線の内側にある部分は表示されるが，外側にある部分については表示されない．

多角形は複雑な形状を表現するため，自己交差することが許されている．多角形の内部判定には odd-even-ルールが適用される．*shape* は *vertices* によって定義される多角形の型を指定する．*shape* に指定された型が *Unknown* であったとき，多角形の形状についての情報が提供されないことを意味している．*Complex*が指定されたときには，多角形の外周線が交差していることを示す．*Nonconvex* の指定は，交差はしないが凸型多角形 (convex)ではないことを示す．*Convex* の指定は多角形が凸型であることを示す．*shape* で指定されたものより，さらに複雑な多角形はインプリメンテーション依存な方法で表示される．PEX サーバ拡張部では，*shape* を無視し，すべての多角形要素を *Unknown* で扱うことが許されている．*ignore_edges* は外周線属性を多角形表示に適用するか否かを指定する．*True* であったときには，外周線は表示されない．*False* であったときには，外周線フラグが *On* であるときに限り，その時点での外周線属性を参照して外周線が表示される．外周線属性 ASF の設定にしたがい，*surface_edge, surface_edge_color, surface_edge_type, surface_edge_width* の各属性が個別のテーブルから得られるか，外周線束テーブルの *edge_bundle_index* で指示されるエントリから得られるかが決められる．

領域属性 ASF の設定にしたがって，*surface_color, interior_style, interior_style_index, surface_interp, reflection_model* の各属性を個別のテーブルから得るか，または領域束テーブルの *interior_bundle_index* で指示されるエントリから得るかが決定される．レンダリングされるとき，多角形が視点に対して表向きであると判断されたならば，*surface_color, reflection_attr* 属性（上述の方法で得られる）は多角形の領域色を計算するために使用される．多角形が視点に対して裏向きであると判断されたならば *bf_surface_color, bf_reflection_attr* 属性が使用される．多角形の面方向によらず，*interior_style* が *Pattern* であったときには，*pattern_size, pattern_ref_pt, pattern_ref_vec1, pattern_ref_vec2* の各属性が内部領域表示のために使用される．

ストラクチャ作成において，多角形に 3 未満の頂点しか与えられなかったとき，ストラクチャ内に格納はされるが，出力コマンドが実行されても視覚的な効果はもたらさない．即時モードでは上記のようなコマンドは無視される．

shape および *ignore_edges* に対して無効な値が設定されたときには，*OutputCommand* エラーが通知される．

Fill area 2D

shape: SHAPE

ignore_edges: BOOLEAN

vertices: LISTofCOORD_2D

多角形要素のレンダリングを指示する．モデリング座標で与えられる頂点がX，Y座標のみで与えられ，Z座標は常に 0 と仮定されていることを除けば，3 次元多角形要素と機能の上で何の変わりもない．

ストラクチャ作成において，多角形に 3 未満の頂点しか与えられなかったとき，ストラクチャ内に格納はされるが，出力コマンドが実行されても視覚的な効果はもたらさない．即時モードでは上記のようなコマンドは無視される．

shape および *ignore_edges* に対して無効な値が設定されたときには，*OutputCommand* エラーが通知される．

Fill area 3D with data

shape: SHAPE

ignore_edges: BOOLEAN
color_type: COLOR_TYPE
facet_attributes: BITMASK_SHORT *c2
vert_attributes: BITMASK_SHORT *c2
facet: FACET
多角形要素のレンダリングを指示する．多角形の各面または各頂点に付加情報 (additional information)が記述できることを除けば 3 次元多角形と同じである．与えられる色値は *color_type* に指定される色型をしている．

facet_attributes は多角形小面に対して指定される属性である．小面 (facet)属性要素へのビット・マスクは以下の並びで与えられる．

color	COLOR
normal	VECTOR_3D

属性ビット・マスクのいずれかのビットがセットされていたならば，対応する属性は小面表現を定義するデータに含まれなければならず，上記の順で与えられる．小面ごとに色値が与えられたとき，その色は小面に本来備わっている色として扱われる．小面に対して法線が与えられたときには，小面の法線として処理される．（法線は単位ベクトルであると仮定される．）

vert_attributes は多角形頂点に対して指定される属性である．頂点属性要素へのビット・マスクは以下の並びで与えられる．

color	COLOR
normal	VECTOR_3D

属性ビット・マスクのいずれかのビットがセットされていたならば，対応する属性は頂点ごとに含まれなければならず，頂点に対する座標データに続けて上記の順で与えられる．

色値が各頂点ごとに与えられた場合，色値は要素の一部と考えられ, *surface_color* 属性の代わりに使用される．頂点色は小面色に優先して使用される．頂点ごとに法線が与えられたときには，多角形の頂点における法線として処理される．加えて，頂点ごとの色値は *surface_interp* 属性により影響を受ける． *surface_interp* 属性は， *surface_interp_asf* が *Individual* に設定されていたときには個別テーブルから得られ，そうでないならば内部束テーブルの *interior_bundle_index* で指定されるエントリから得られる． *surface_interp* 属性は頂点間で色値をいかにシェーディングするかを規定している．

shape は *vertices* によって定義される多角形の型を指定する． *shape* に指定された型が *Unknown* であったとき， 多角形の形状についての情報が提供されないことを意味している． *Complex*が指定されたときには，多角形の外周線が交差していることを示す． *Nonconvex* の指定は，交差はしないが凸型多角形ではないことを示す． *Convex* の指定は多角形が凸型であることを示す． *shape* で指定されたものよりさらに複雑な多角形はインプリメンテーション依存な方法で表示される． PEX サーバ拡張部では， *shape* を無視し，すべての多角形要素を *Unknown* で扱うことが許されている． *ignore_edges* は外周線属性を多角形表示に適用するか否かを指定する． *True* であったときには， *fill area* について外周線は表示されない． *False* であったときには，外周線フラグが *On*であるときに限り，その時点での外周線属性を参照して外周線が表示される．

ストラクチャ作成において，多角形に 3 未満の頂点しか与えられなかったとき，ストラクチャ内に格納はされるが，出力コマンドが実行されても視覚的な効果はもたらさない．即時モードでは上記のようなコマンドは無視される．

shape および *ignore_edges* に対して無効な値が設定されたときには， *OutputCommand* エラーが通知される．

Fill area set 3D
shape: SHAPE
ignore_edges: BOOLEAN
contour_hint: CONTOUR
vertices: LISTofLISTofCOORD_3D

多角形集合要素 (fill area set)の表示を指示する．基本的には 3 次元多角形の集合であると同時に，本要素は内包多角形 (polygon with islands)または穴空き多角形 (polygon with holes)を表示する．結果の多角形が平面上にあるかどうかは保証されないが，平面でない多角形集合の表示はインプリメンテーション依存である．
ignore_edges は多角形集合を構成するすべての多角形に適用される． *c2

shape は多角形集合を構成する外周 (contour)の型を指定する． *shape* が *Unknown* であったときには，多角形集合の外周について情報がないことを意味している． *Complex* であったときには，いくつかの多角形の外周が自身で交差していることを意味している． *Nonconvex* であったときには，外周が自己交差している多角形は含まれていないことを意味する．しかしすべての多角形が凸型である必要はない． *Convex*であったときには，すべての多角形が凸型であることを意味する． *shape* で指定されたものよりさらに複雑な多角形集合はインプリメンテーション依存な方法で表示される． PEX サーバ拡張部では， *shape* を無視し，すべての多角形集合要素を *Unknown* で扱うことが許されている．

contour_hint はさらに集合要素に含まれる多角形外周間の関係を規定する． *contour_hint* が *Disjoint* であったときには， 個々の多角形は空間的に分離されている．重なったり，交わったりすることはない． *Nested* であったときには，孤立しているか，多角形のあるものは完全に他の多角形に包含されている．他の多角形の外周と交差したり，外周が重なることはない． *Intersecting* が指定されたときには，別々の多角形の外周が交わったり，重なったりしている． *Unknown* であったときには多角形外周間の関係がいっさい知られていないことを意味している． *contour_hint* により指定可能なものよりさらに複雑な多角形外周間の関係が与えられたときには，インプリメンテーション依存な方法で表示される． PEX サーバ拡張部は *contour_hint* の設定を無視し， *Unknown* として動作することが可能である．

ストラクチャ作成において，集合を構成する多角形に 3 未満の頂点しか与えられなかったとき，または集合を構成する多角形がひとつも与えられなかったとき，ストラクチャ内に格納はされるが，出力コマンドが実行されても視覚的な効果はもたらさない．即時モードでは上記のようなコマンドは無視される．

shape および *ignore_edges* に対して無効な値が設定されたときには， *OutputCommand* エラーが通知される．

Fill area set 2D

shape: SHAPE
ignore_edges: BOOLEAN
contour_hint: CONTOUR
vertices: LISTofLISTofCOORD_2D
多角形集合要素の表示を指示する．モデリング座標で与えられる頂点が X，Y 座標のみで与えられ，Z 座標は常に 0 と仮定されていることを除けば，3 次元多角形要素と機能の上で変わりはない．

ストラクチャ作成において，集合を構成する多角形に 3 未満の頂点しか与えられなかったとき，または集合を構成する多角形がひとつも与えられなかったとき，ストラクチャ内に格納はされるが，出力コマンドが実行されても視覚的な効果はもたらさない．即時モードでは上記のようなコマンドは無視される．

shape および *ignore_edges* に対して無効な値が設定されたときには， *OutputCommand* エラーが通知される．

Fill area set 3D with data

shape: SHAPE
ignore_edges: BOOLEAN
contour_hint: CONTOUR
color_type: COLOR_TYPE
facet_attributes: BITMASK_SHORT *c2
vert_attributes: BITMASK_SHORT *c2
facet_data: OPT_DATA
vertices: LISTofLISTofVERTEX
多角形集合要素の表示を指示する．多角形の各辺または各頂点に付加情報が記述できることを除けば 3 次元多角形集合と同じである．与えられる色値は *color_type* に指定される色型をしている． *c1

facet_attributes は *facet_data* に指定される属性である．小面属性要素へのビット・マスクは以下の並びで与えられる．

color	COLOR
normal	VECTOR_3D

属性ビット・マスクのいずれかのビットがセットされていたならば，対応する属性は小面表現を定義するデータに含まれなければならず，上記の順で与えられる．小面ごとに色値が与えられたとき，その色は小面に本来備わっている色として扱われる．

vert_attributes は多角形頂点に対して指定される属性である．頂点属性要素へのビット・マスクは以下の並びで与えられる．

color	COLOR
normal	VECTOR_3D
edges	SWITCH

属性ビット・マスクのいずれかのビットがセットされていたならば，対応する属性は頂点ごとに含まれなければならず，頂点に対する座標データに続けて上記の順で与えられる．

色値が *facet_data* の一部，または頂点データの一部として与えられた場合，色値は要素の一部と考えられ，*surface_color* 属性の代わりに使用される．頂点色は小面色に優先して使用される．頂点ごとに法線が与えられたときには，多角形の頂点における法線として処理される．外周線フラグが頂点ごとに指定されたときには，各フラグはそのフラグが付加された頂点から次の頂点までを結ぶ外周線の表示の有無を指定する（つまりは，4 頂点からなる小面において，頂点 #1 に付記された外周フラグは #1 と #2 を結ぶ外周線を表示するか否かを指示し，頂点 #2 に付記された外周フラグは #2 と #3 を結ぶ外周線の，頂点 #3 に付記された外周フラグは #3 と #4 を結ぶ外周線の，頂点 #4 に付記された外周フラグは #4 と #1 を結ぶ外周線の表示を制御する）．外周線は常に外周線色で表示され，小面色または頂点色で表示されることはない．

加えて，頂点ごとの色値は *surface_interp* 属性により影響を受ける． *surface_interp* 属性は， *surface_interp_asf* が *Individual* に設定されていたときには個別テーブルから得られ，そうでないならば内部束テーブルの *interior_bundle_index* で指定されるエントリから得られる． *surface_interp* 属性は頂点間で色値をいかにシェーディングするかを規定している．

ストラクチャ作成において，多角形に 3 未満の頂点しか与えられなかったとき，ストラクチャ内に格納はされるが，出力コマンドが実行されても視覚的な効果はもたらさない．即時モードでは上記のようなコマンドは無視される．

shape， *ignore_edges*， *contour_hint* に対して無効な値が設定されたときには， *OutputCommand* エラーが通知される． *c1

Triangle strip

color_type: COLOR_TYPE
facet_attributes: BITMASK_SHORT *c2
vert_attributes: BITMASK_SHORT *c2
facet_data: LISTofOPT_DATA
vertices: LISTofVERTEX
三角形ストリップ要素 (triangle strip primitive)の表示を指示する．渡される色値は *color_type* で指定される色型をしている． *facet_attributes* は三角形ストリップの各小面に対する属性を指定している．小面属性要素へのビット・マスクは以下の並びとなっている．

color	COLOR
normal	VECTOR_3D

いずれかのビットが設定されていた場合，対応する属性は，各三角形の小面表示を規定する *facet_data* に含まれていなければならない．この方法で渡される属性は上記の順をしていなければならない．色値が小面ごと

に渡されたとき，その色値は小面の表面に本来備わっている色として扱われる．法線が小面ごとに渡されたとき，その小面の法線として扱われる．（法線は単位長ベクトルであると仮定される．） *facet* 内には (n-2) のエントリが含まれていて， n は *vertices* に含まれるエントリ数である． *c1

vert_attributes は三角形ストリップを構成する．各頂点ごとに指定される属性である頂点属性要素へのビット・マスクは以下の並びとなっている．

color	COLOR
normal	VECTOR_3D

いずれかのビットが設定されていた場合，対応する属性は，各頂点に与えられなければならず，また頂点座標データの後に渡されなければならない．この方法で渡される属性は上記の順をしていなければならない．

色値が各頂点ごとに与えられた場合，色値は要素の一部と考えられ, *surface_color* 属性の代わりに使用される．頂点色は小面色に優先して使用される．頂点ごとに法線が与えられたときには，小面の頂点における法線として処理される．

三角形ストリップは頂点列から生成される． n を頂点数とすると， (n-2) の三角形ストリップが生成される．第一の三角形は最初の 3 頂点から生成される．第二の三角形は第二頂点から第四頂点を用いて表現される．以下同様に，最後の 3 頂点を用いて最終三角形が生成されるまで繰り返される．

多角形を表現する際に影響を及ぼす属性は，すべて三角形ストリップを表現する際にも影響を持つ．

ストラクチャを生成するとき，三角形ストリップが 3 未満の頂点しか与えられなかった場合，ストラクチャ内に格納はされるが，出力コマンドを実行する際，視覚的な効果はもたらさない．即時モードでは，このような要素は無視される．

Quadrilateral mesh

shape: SHAPE
color_type: COLOR_TYPE
m_pts: CARD16
n_pts: CARD16
facet_attributes: BITMASK_SHORT *c2
vert_attributes: BITMASK_SHORT *c2
facet_data: LISTofOPT_DATA
vertices: LISTofVERTEX

四辺形メッシュ要素 (quadrilateral mesh primitive)の表示を指示する． *shape* はメッシュを構成する四辺形の形を指定する． *shape* が *Unknown* であったときには，四辺形の形状について情報がないことを意味している． *Complex* であったときには，いくつかの四辺形の外周が自身で交差していることを意味している． *Non-convex* であったときには，外周が自己交差している四辺形は含まれていないことを意味する．しかしすべての四辺形が凸型である必要はない． *Convex* であったときには，すべての四辺形が凸型であることを意味する． *shape* で指定されたものよりさらに複雑な四辺形はインプリメンテーション依存な方法で表示される．色型は *color_type* で渡される型をしている．

facet_attributes は四辺形メッシュに現れる各小面に対して指定される属性である．小面属性要素へのビット・マスクは以下の並びとなっている．

color	COLOR
normal	VECTOR_3D

いずれかのビットが設定されていた場合，対応する属性は *facet_data* に与えられなければならず，この属性は四辺形メッシュの各小面の表現を規定する．この方法で渡される属性は上記の順をしていなければならない．小面ごとに色値が与えられたときには，その色値は小面の表面に本来備わっている色として扱われる．法線が小面ごとに渡されたとき，その小面の法線として扱われる．（法線は単位長ベクトルであると仮定される．）

vert_attributes は四辺形メッシュを構成する各頂点ごとに指定される属性である．頂点属性要素へのビット・マスクは以下の並びとなっている．

color COLOR
normal VECTOR_3D

いずれかのビットが設定されていた場合，対応する属性は，各頂点に与えられなければならず，また頂点座標データの後に渡されなければならない．この方法で渡される属性は上記の順をしていなければならない．

色値が各頂点ごとに与えられた場合，色値は要素の一部と考えられ，*surface_color* 属性の代わりに使用される．頂点色は小面色に優先して使用される．頂点ごとに法線が与えられたときには，小面の頂点における法線として処理される．

曲面は頂点の配列により定義され，頂点の配列は行方向主導（配列内に頂点を格納する際，桁数の方が早く変化する）．(i, j), (i+1,j), (i+1, j+1), (i, j+1) で与えられる頂点がひとつのメッシュを構成する．全メッシュが作成されるまで隣り合う頂点が結合される．頂点配列には *m_pts* × *n_pts* 個のエントリが含まれていて，面ごとのデータが与えられているときには， *facet_data* には (*m_pts*-1) × (*n_pts*-1) 個のエントリが含まれることになる． *m_pts* は頂点配列中の行数， *n_pts* は列数を示す． *c1 *c1

ひとつの小面の境界が一平面にのっている必要はないが，この場合の頂点属性の取り扱いはインプリメンテーション依存となる．

多角形の表現に影響を持つ属性はすべて四辺形メッシュの表現においても影響を及ぼす．

ストラクチャ作成時， *m_pts*, *n_pts* に 2 未満の数値が指定された場合，ストラクチャ内に格納されるが， 本出力コマンドが実行されても表示はおこなわれない．即時モードでは，上記要素は無視される．

shape に無効な数値が指定されたときには， *OutputCommand* エラーが通知される

Set of fill area sets

shape: SHAPE
color_type: COLOR_TYPE
fas_attributes: BITMASK_SHORT *c2
vert_attributes: BITMASK_SHORT *c2
edge_attributes: BITMASK_SHORT *c2
contour_hint: CONTOUR
contours_all_1: BOOLEAN
num_fas: CARD16
num_verts: CARD16
num_edges: CARD16
num_contours: CARD16
per_fas_data: LISTofOPT_DATA
per_vert_data: LISTofVERTEX
per_edge_data: LISTofSWITCH
contours: LISTofLISTofLISTofCARD16

多角形集合の集合要素の表示を指示する．（個々の多角形集合において，集合に含まれる多角形は互いに幾何情報および属性情報を頂点において共有する．）シェーディング計算，座標変換は各多角形について別々におこなう必要はなく，共有された頂点において一度おこなえばよい．同様に，ネットワーク上での頂点データ転送も各多角形について，すべての頂点を転送する必要はなく，ユニークな頂点に関して一度転送をおこなえばよい．

shape は各多角形集合を構成する多角形の形状を指定する． *shape* が *Unknown* であったときには，多角形の形状について情報がないことを意味している． *Complex* であったときには，いくつかの多角形の外周が自身で交差していることを意味している． *Nonconvex* であったときには，外周が自己交差している多角形は含まれていないことを意味する．しかしすべての四辺形が凸型である必要はない． *Convex* であったときにはすべての四辺形が凸型であることを意味する．（複数の多角形を含む多角形集合は外周が交差することが許されている． 多角形集合に含まれる多角形がひとつであるか， *contours_all_1* が *True* であり， *shape* が *Convex* であったときには，高速処理コード (fast rendering code path)の使用が許される．） *shape* で指定されたものよりさらに複雑な四辺形はインプリメンテーション依存な方法で表示される． PEX サーバ拡張部は *shape* の設定を無視し, *Unknown* として動作することが可能である．

contour_hint はさらに多角形集合要素に含まれる多角形外周間の関係を規定する．*contour_hint* が *Disjoint* であったときには，個々の多角形は空間的に分離されており，重なったり，交わったりすることはない．*Nested* であったときには，孤立しているか，多角形のあるものは完全に他の多角形に包含されている．他の多角形の外周と交差したり，外周が重なることはない．*Intersecting* が指定されたときには，別々の多角形の外周が交わったり，重なったりしている．*Unknown* であったときには多角形外周間の関係がいっさい知られていないことを意味している．*contour_hint* により指定可能なものより，さらに複雑な多角形外周間の関係が与えられたときには，インプリメンテーション依存な方法で表示される．PEX サーバ拡張部は *contour_hint* の設定を無視し，*Unknown* として動作することが可能である．

渡される色型は *color_type* で指定される型を持つ．

fas_attributes は各多角形集合に対して指定される属性を与える．*fas_attributes* 属性へのビット・マスクは以下の並びをしている．

color	COLOR
normal	VECTOR_3D

いずれかのビットが設定されていた場合，対応する属性は *fas_data* ごとに含まれていなければならず，*fas_data* は各多角形集合に対するデータ・レコードを保持している．この方法で渡される属性は上記の順をしていなければならない．色値が渡されると，その色値は小面の表面に本来備わっている色として扱われる．法線が小面ごとに渡されたとき，その小面の法線として扱われる．（法線は単位長ベクタであると仮定される．非単位長の法線ベクトルの表示についてはインプリメンテーション依存である．）*fas_attributes* が Null 設定であった ならば，*per_fas_data* も同様 Null 設定となる．*fas_attributes* が Null 設定でなかったならば，*num_fas* 分のエントリが *per_facet_data* 内に作成される． *c2

vert_attributes は，*per_vert_data* により指定されるユニークな頂点リストに含まれる頂点に対して与えられる属性を示している．頂点属性へのビット・マスクの並びは以下のようになる．

color	COLOR
normal	VECTOR_3D

いずれかのビットが設定されていた場合，対応する属性は *per_vert_data* 内の頂点ごとに与えられなければならず，各頂点ごとの座標データの後に渡されなければならない．この方法で渡される属性は上記の順をしていなければならない．頂点ごとに色値が渡されると，その色値は要素の一部と見なされ，*surface_color* の代わりに使用される．法線が頂点ごとに渡されたとき，その頂点の法線と見なされ，同一頂点を共有するすべての頂点に対して適用される．*per_vert_data* には *num_verts* 数だけのエントリが作成される．

edge_attributes は各頂点に与えられる属性を保持している．外周線属性へのビット・マスクは以下の並びをしている．

edges	SWITCH

いずれかのビットが設定されていた場合，対応する属性は *per_edge_data* 内の各外周線に与えられなければならない．もし，いかなる属性も指定されなかったときには *per_edge_data* は空なデータ・リストとなる．なんらかの属性が指定されたときには *num_edges* 分のエントリが作成される．各エントリは対応する外周線が表示されるか否かを決定するスイッチを含んでいる．本リストにはカウンタが存在しないため，リストが空でなかったときには，*contours* に保持されるデータを処理する際，リスト中の正しい位置を保証することは PEX サーバの責任である．外周線スイッチ (edge switch)が付加されたとき，スイッチが付加された頂点から次の頂点を結ぶ外周線を表示するか否かは，付加されたスイッチにより決定される．（つまりは，4 頂点からなる小面において頂点 #1 に付記された外周フラグは #1 と #2 を結ぶ外周線を表示するか否かを指示し，頂点 #2 に付記された外周フラグは #2 と #3 を結ぶ外周線の，頂点 #3 に付記された外周フラグは #3 と #4 を結ぶ外周線の，頂点 #4 に付記された外周フラグは #4 と #1 を結ぶ外周線の表示を制御する）．外周線は常に外周線色で表示され，小面色または頂点色で表示されることはない．

要素の結合規則は *contours* 配列により定義される．第一の多角形集合に含まれる外周（多角形）の数は *contours* 配列の第一エントリ内に記述されている．多角形数には多角形データ・レコードが続き，多角形集合に含

まれる多角形群に対し，ひとつ与えられる．多角形数が n であったとき， *contours* 配列の n データ・レコードが最初の多角形集合を定義するために使用される．多角形数に続く数値は，最初の多角形集合における最初の多角形の頂点数を保持している．頂点数が m であったとき，配列内の以下の m 個の数値が最初の多角形のデータ・レコードである．このデータ・レコードは，最初の多角形の頂点データへの指標を含んでいる． n の値によって，その次に続く数値は第一の多角形集合における二番目の多角形の頂点数，または二番目の多角形集合内の多角形数となる．特別な場合として， *contours_all_1* が *True* であったならば，各多角形集合に含まれる多角形数は常に 1 であることが保証される．

指標により番号付けられた頂点は 0 から始まる．（つまり最初の頂点は頂点 0 として参照される．）

多角形要素の表示に影響を持つ属性は，すべて多角形集合の集合の表示にも影響を及ぼす．

per_edge_data 配列における任意の外周線フラグが *On/Off* のいずれでもなかったならば， *OutputCommand* エラーが通知される． *shape, contour_hint, contours_all_1* のどれかが無効であった，または *contours* 配列内の頂点指標のどれかが有効範囲外であった，または *contours* 配列内の頂点指標数と同じ数の外周線フラグが存在しなかったときにも， *OutputCommand* エラーが通知される．

Non-uniform B-spline surface

type: COORD_TYPE
u_order: CARD16
v_order: CARD16
u_knots: LISTofFLOAT
v_knots: LISTofFLOAT
mpts: CARD16
npts: CARD16
points: LISTofCOORD
trim_curves: LISTofLISTofTRIM_CURVE
NURBS-曲面のレンダリングを指示する．パラメタ変数 u, v の関数として曲面を表示する． *u_order, v_order* はパラメタ変数の次数を正の整数で指定する．曲面の形はパラメトリック空間における節点の並びとモデリング空間における制御点の並びとによって規定される． *u_knots* と *v_knots* は共に非減少な数値並びとなっていなければならない． *mpts* は u- 方向の点数， *npts* は v- 方向の点数を示してる．点は頂点配列データに列優先順で格納される（つまり桁の方が早く変化する）．行は v が増加する方向に増加する． *c2

u_knots 内節点の最小／最大値は， u- 軸方向の NURBS 面有効範囲を定義し， *v_knots* 内節点の最小／最大値は， v- 軸方向の NURBS 面有効範囲を定義する．

type は曲面が有理型か非有理型かを指定する． *surface_type* が *Rational* であったならば，制御点は同次モデリング座標 (COORD_4D) で与えられなければならない． *NonRational* であったときには， 非同次モデリング座標 (COORD_3D) で与えられなければならない．

パラメタ変数の制限に加え，トリミング・ループ (trimming loop)のリストが与えられる．トリミング・ループは， NURBS 面の有効範囲をパラメトリック空間においてさらに制限するものである．各トリミング・ループは一つの NURBS 曲線ループ，または複数の NURBS 曲線セグメントの先頭と末尾を結ぶことによりできるループにより定義される．トリム曲線は有理／非有理のいずれでもよく，また異なった次数を持つことも許されている．トリム曲線は明確に閉じていなければならず，おのおののトリミング・ループにおいて最終セグメントの末尾と第一セグメントの先頭は必ず結合される．各トリム曲線は個々にパラメトリック表現される．セグメントの先頭と末尾の連結において，浮動少数点データ上の誤差があった場合でも，連結処理は保証されなければならない．トリミング・ループを構成する NURBS 曲線セグメントは， 曲面を定義するパラメトリック空間内で定義され，同空間を越えることは許されない． NURBS 面に対して一つもトリミング・ループが定義されなかったときには，外周線フラグ属性にしたがって矩形の境界が曲面の外周として表示される．

トリミング・ループは曲面のレンダリングの対象となる範囲を規定する．レンダリングは以下の 2 つのルールにしたがっておこなわれる． (1) 曲面上の点で，その点と無限遠点を結ぶ線分がトリミング・ループと奇数回交差する場合，その点をレンダリングする． (2) トリム曲線上をループの方向に移動したとき，進行方向右側の領域は対象外，左側の領域をレンダリングの対象とする．つまり，ループが反時計回りに定義されたときには，ループの内側がレンダリングされ，外側はクリップされる．逆に時計回りにループが定義されていると

きには，外側がレンダリングされ，内側はレンダリングされない．ループが入れ子になっていたときには，定義される方向は反対となっていなければならない．最も外側のループは必ず反時計回りに定義されなければならない．いかなる B-spline 曲線，トリミング・ループも交差してはならない． 上記のルールに合わないトリミング・ループはインプリメンテーション依存の結果となる． *c2

各 NURBS 曲線セグメントには， 曲面の外周の表示を制御するために可視性フラグ (visibility flag)が付加されている．レンダラ属性内の外周線属性および可視性フラグの設定によって，曲線セグメントは曲面の外周として表示される． *c2

領域属性はすべて NURBS 曲面の表示において影響を持つ．加えて *surface_approx* は NURBS 曲面をどのように近似するかを指示し， *psurf_char* は曲面をどのように視覚表現するかを決定する．

曲面の次数かトリム曲線の次数が PEX サーバによりサポートされていなかったときには，出力結果は視覚的効果を持たない．即時モードではこのような要素は無視される．

以下の場合， *OutputCommand* エラーが通知される． *type* が *Rational/Nonrational* のいずれでもなかった． *order* に 1 より小さい数値が指定された． u- 方向, v- 方向のいずれかで制御点数が *order* より少なかった． *order* が節点数と制御点数とに対して矛盾していた．節点に与えられた数値が非減少ではなかった． 有理制御点の w 要素に 0 以下の値が指定された．トリム曲線の *order* が 2 未満であった．トリム曲線がその *order* に対し十分な制御点数を持たなかった． トリム曲線の *order* が節点数と制点数とに対して矛盾していた．節点に与えられた数値が非減少ではなかった．トリム曲線のパラメータの範囲が節点値と矛盾していた．

Cell array 3D

point1: COORD_3D
point2: COORD_3D
point3: COORD_3D
dx: CARD32
dy: CARD32
colors: LISTofTABLE_INDEX

3 次元セル配列要素 (cell array primitive)の表示を指示する．色値は *Indexed* 型で渡される． セル配列は単一サイズの平行四辺形の並びで，各セルはひとつの色を持っている．各セルは以下の式で与えられる幅と高さを持っている．

$$width = \frac{\sqrt{(point1.x - point2.x)^2 + (point1.y - point2.y)^2 + (point1.z - point2.z)^2}}{dx}$$

$$height = \frac{\sqrt{(point1.x - point3.x)^2 + (point1.y - point3.y)^2 + (point1.z - point3.z)^2}}{dy}$$

色値は， $dx \times dy$ の大きさを持つ一次元配列に指定される．各セルの色値は *color* 配列内の対応する要素から得られる． 色値は本配列内に行方向優先で格納されている．つまり桁方向のカウンタが早く変化する． 配列内の最初の色値は *point1* の位置のセル色となり，以下に続く色値は *point2* の位置に向かう方向に順にセルに割付けされていく． *c2

未定義な色指標値が指定されたときには，色指標 1 が使用される．色指標 1 が未定義であったときには Appendix D に示された省略値が使用される．

dx または *dy* に 0 が指定されたときには， *OutputCommand* エラーが通知される．

Cell array 2D

point1: COORD_2D
point2: COORD_2D
dx: CARD32
dy: CARD32
colors: LISTofTABLE_INDEX

2 次元セル配列要素の表示を指示する．本要素はモデリング座標で与えられる頂点がX，Y座標のみで与えら

れ，Z座標は常に0と仮定されていることを除けば，3次元セル配列要素と機能の上で同じである．セルは，モデリング座標系に並べられる矩形を規定する2点により定義される．

未定義な色指標値が指定されたときには，色指標1が使用される．色指標1が未定義であったときには Appendix D に示された省略値が使用される．

dx, dy に0が指定されたときには，*OutputCommand* エラーが通知される．

Extended cell array 3D

color_type: COLOR_TYPE
point1: COORD_3D
point2: COORD_3D
point3: COORD_3D
dx: CARD32
dy: CARD32
colors: LISTofCOLOR
color_type の設定により，色値型が指標指定か直値指定かが選択されることを除き，3次元セル配列と同じである．

未定義な色指標値が指定されたときには，色指標1が使用される．色指標1が未定義であったときには Appendix D に示された省略値が使用される．

dx または *dy* に0が指定されたときには，*OutputCommand* エラーが通知される．

GDP 3D

gdp_id: INT32
points: LISTofCOORD_3D
data: LISTofCARD8
インプリメンテーション依存の要素を提供する．ネットワーク上の浮動小数点データ型，および色値型の不一致のため，3次元 GDP 要素は拡張機能としてそれほど有効ではない．PHIGS 互換のために用意されている．

指定された GDP-id がサポートされていなかったときには，出力コマンドは無視される．この型の出力コマンドでは，バイト交換や浮動小数点データ型変換はおこなわれない．

GDP 2D

gdp_id: INT32
points: LISTofCOORD_2D
data: LISTofCARD8
インプリメンテーション依存の要素を提供する．ネットワーク上の浮動小数点データ型，および色値型の不一致のため，2次元 GDP 要素は拡張機能としてそれほど有効ではない．PHIGS 互換のために用意されている．

指定された GDP-id がサポートされていなかったときには，出力コマンドは無視される．この型の出力コマンドでは，バイト交換や浮動小数点データ型変換はおこなわれない． *c1

Noop *c2

レンダラによって処理されるとき，このコマンドはカレント・パスの更新のみ実行する．これは何の出力も生成しないので，クライアント側での累積状態探訪 (accumulate state traversal)において有用である．

4 ルックアップ・テーブル

ルックアップ・テーブル (lookup table)は PEX のリソースであり，クライアントがいろいろな目的のために参照するデータ・テーブルのことである．ルックアップ・テーブルは，たとえば視野情報 (view information)や光源情報 (light source description)，またはデプス・キュー情報などを格納し，それらに間接的にアクセスする方法を提供する．

テーブルは疎 (sparse) かもしれないので，指標 (index)/ エントリ (entry) の組で構成される． *index* はエントリを参照するための数値である．指標は 16 ビットであるため，可能な指標値は [0..65535] の範囲内にある．ただし可能ならば 0 （いくつかのテーブルは 0 指標を許し，他はそうでない）と， 65535 （テーブルは最大でも $2^{16}-1$ 個のエントリしか持てないため，もしテーブルが 0 から始まれば，最大の指標値は 65534 となる）を除く．一つの *entry* はテーブルの各型について定義された情報（およびデータレコード）からなる．ある指標がどのテーブルのエントリにも関連しないとき，あるいは関連するテーブルのエントリが削除されているときは， *undefined entry* を指すことになる．一つのテーブル・エントリはそのテーブルの型に応じて一つ以上のデータ項目を含む．たとえば *MarkerBundle* テーブル型については，各エントリは一つのマーカ型値，マーカ・スケール値，マーカ色値，からなる．

一つのルックアップ・テーブルは *predefined entries* を持つかもしれない．これらはテーブルが作成されるとき，サーバによって自動的に値を設定された指標値に対応するものである． **PEXGetTableInfo** が事前設定のエントリ範囲を返す．事前設定のテーブル・エントリは削除されたり，上書きされたりする可能性もある． *c2

テーブルの各型は *default table entry* を示すように設定された値を持つ．もしテーブルの指標が未定義のテーブル・エントリを参照するならば，省略値のテーブル・エントリの内容が使用される．もし省略値のテーブル・エントリが未定義ならば，省略値の属性値（Appendix D に掲載）が省略値エントリとして使用される．すべてのテーブルに対する省略値エントリは 1 である．ただし視野テーブル (view table)，デプス・キュー・テーブル (depth cue table)，色近似テーブル (color approximation table)は 0 である．

PEX のルックアップ・テーブルは PHIGS の set/realized をサポートしている．一つのルックアップ・テーブルのエントリは現実に表現不可能な値に設定されるかもしれない．そのような値はドローアブルに表示されるとき，暗黙のうちに相応の省略値にマップされる．ルックアップ・テーブルからエントリ値を引き出す際，クライアントはクライアントによって設定された元の値（ *value_type* = *Set*），あるいは表示の際に実際に使用される値，すなわち実際に設定された値あるいは近似値（ *value_type* = *Realized*）を要求するかもしれない．すべての事前設定エントリ *c2
は定義によって実現可能であるため，事前設定されたルックアップ・テーブルのエントリは *Set* あるいは *Realized* がリクエストされていることにかかわらず，同じ値を返す．

ルックアップ・テーブルの型，指標値の範囲，省略値エントリ，エントリの形式などには以下のものがある．

線束 *LineBundle(1..65535, default entry = 1)*

このテーブル型は折線や曲線を表示する属性のものである． ASF フラグの設定により，折線や曲線属性は，このテーブルから獲得される．各エントリは以下の属性データから構成されている．

line_type	LINE_TYPE
polyline_interp	POLYLINE_INTERP
curve_approx	CURVE_APPROX
line_width	FLOAT
line_color	COLOR_SPECIFIER

このテーブルに格納される属性は同一名のパイプライン・コンテキスト属性と同じ方法で定義，使用される．

マーカ束 *MarkerBundle(1..65535, default entry = 1)*

このテーブル型はマーカ要素を表示する属性のものである． ASF フラグの設定により，マーカ属性は，このテーブルから獲得される．各エントリは以下の属性データから構成されている．

marker_type	MARKER_TYPE
marker_scale	FLOAT
marker_color	COLOR_SPECIFIER

このテーブルに格納される属性は同一名のパイプライン・コンテキスト属性と同じ方法で定義，使用される．

文字列束 *TextBundle(1..65535, default entry = 1)*

このテーブル型は文字列および注釈文字列要素を表示する属性のものである． ASF フラグの設定により，文字列および注釈文字列属性は，このテーブルから得られる．各エントリは以下の属性データから構成されている．

text_font_index	TABLE_INDEX
text_precision	TEXT_PRECISION
char_expansion	FLOAT
char_spacing	FLOAT
text_color	COLOR_SPECIFIER

このテーブルに格納される属性は同一名のパイプライン・コンテキスト属性と同じ方法で定義，使用される．

内部束 *InteriorBundle(1..65535, default entry = 1)*

このテーブル型は領域要素を表示する属性のものである． ASF フラグの設定により，領域表示属性は，本テーブルから獲得される．各エントリは以下の属性データから構成されている．

interior_style	INTERIOR_STYLE
interior_style_index	TYPE_OR_TABLE_INDEX
surface_color	COLOR_SPECIFIER
reflection_attr	REFLECTION_ATTR
reflection_model	REFLECTION_MODEL
surface_interp	SURFACE_INTERP
bf_interior_style	INTERIOR_STYLE
bf_interior_style_index	TYPE_OR_TABLE_INDEX
bf_surface_color	COLOR_SPECIFIER
bf_reflection_attr	REFLECTION_ATTR
bf_reflection_model	REFLECTION_MODEL
bf_surface_interp	SURFACE_INTERP
surface_approx	SURFACE_APPROX

このテーブルに格納される属性は同一名のパイプライン・コンテキスト属性と同じ方法で定義，使用される．

外周線束 *EdgeBundle(1..65535 , default entry = 1)*

このテーブル型は領域要素の外周線表示を表示する属性のものである． ASF フラグの設定により，外周線表示属性は，本テーブルから獲得される．各エントリは以下の属性データから構成されている．

surface_edges	SWITCH
surface_edge_type	SURFACE_EDGE_TYPE
surface_edge_width	FLOAT
surface_edge_color	COLOR_SPECIFIER

このテーブルに格納される属性は同一名のパイプライン・コンテキスト属性と同じ方法で定義，使用される．

パターン *Pattern(1..65535 , default entry = 1)*

このテーブル型は，領域表示属性の *interior_style* が *Pattern* であるときに，使用されるパターンのものである．

color_type	COLOR_TYPE
numx	CARD16
numy	CARD16
colors	LISTofCOLOR

一つのパターン矩形は $numx \times numy$ セルで構成される． *color_type* は色値が指標か直値かを示す．色値は配列の中で行方向優先順で配置される．パターン矩形の左上隅のセルは色値の並びの中で最初のものを指し，行方向に続いていく．その次に第 2 行目の色値が続く．

色 *Color(0..65534 , default entry = 1)*

このテーブル型は間接色値参照 (indirect color reference)機能を実現するものである．したがって，このテーブルにおいて使用される色値は必ず直値でなければならない．

color_type	COLOR_TYPE
color	DIRECT_COLOR

ColorType のエラーは *indexed* の *color_type* を持つエントリを作ろうとしたときに発生する．

文字列フォント *TextFont(1..65535 , default entry = 1)*

このテーブル型はフォント・グループのリストを管理するものである．フォント・グループは PEX フォントまたは X11 フォントに対するフォント・リソース ID に対応している．フォントに関するリソース ID はフォント・テーブルにおいて重複して使用されてもよい． PEX フォントのみは PEX のテキスト属性すべてを実現することを保証されている．特に文字列の拡大縮小や回転操作は X11 フォントを使用したときには保証されないが， PEX フォントを使うときは保証される．フォント値は出力コマンドが使われたとき，指標として設定される．フォント指標はこの型のテーブル（実際のフォント・グループを得るための）と共に使用できる．フォント・グループ中におけるフォントの切り換え (switching)は，表示される文字列の中に埋め込まれた切り換え機構によって実現される．クライアントに対してはこのテーブル中で使用可能なすべての文字セットは，省略値テーブルのエントリについても使用可能であることを保証する．

font	LISTofFONT_ID

もしあるフォントがオープン (open)され，フォント・テーブルと結合し，クローズ (close)したならば，フォントの内容は残る．なぜならば内容はフォント・テーブルにおいて参照されるからである．しかしフォント・テーブルが参照されたとき， *AlreadyFreed* の値がそのフォントに対して返るだろう．なぜならば参照できるリソース ID がもはや正しくないからである．

視野 *View(0..65534 , default entry = 0)*

このテーブル型は視野に関する情報を管理するものである．「視野」はルックアップ・テーブルにおいて適当な情報を指示する指標によって指定される． *c2

clip_flags	BITMASK_SHORT
clip_limits	NPC_SUBVOLUME
orientation	MATRIX
mapping	MATRIX

Orientation は世界座標系 (world coordinates)における図形データを視野参照座標系（View_reference_coordinates, Eye_coordinates, Viewing_coordinates, 視点座標系あるいは視野座標系ともよばれる右手系）へマッピングするための変換行列である. *Mapping* は視野座標系における図形データを正規化投影座標系 (normalized projection coordinates)（右手系）へマッピングするための変換行列である. この座標系は典型的には平行投影 (parallel projection)と透視投影 (perspective projection)の 2 つがある. *clip_limits* は正規化投影座標系における直方体の最大値と最小値を指定する. *clip_flags* は 3 ビットからなり, *clip_limits* により規定された領域の x, y, z のそれぞれの境界面を機能させるか否かを指定する. NPC サブボリュームは *clip_flags* と *clip_limits* と組み合わされて, 実際のクリップ領域を決定する. *c2

光源 *Light(1..65535 , default entry = 1)*

このテーブル型はシェーディング計算における光源の定義情報を管理する.

light_type	LIGHT_TYPE
direction	VECTOR_3D
point	COORD_3D
concentration	FLOAT
spread_angle	FLOAT
attenuation	[factor1, factor2: FLOAT]
color	COLOR_SPECIFIER

*c1

現在 PEX がサポートしている光源は, *ambient*, *WCS vector*, *WCS point*, *WCS spot* の 4 種である. 光源の型によっては, エントリを構成する要素のうちいくつかのものは無視される.

環境光源 (ambient light source)は *color* によって定義される. これはすべての物体の表面の色にその方向と無関係に影響を与える無指向性の光である.

WCS ベクトル光源 (WCS vector light source)（あるいは平行光源 (directional light source)）は *direction* と *color* で定義される. このタイプの光源は無限遠点に位置し, すべての光線は物体の表面では平行である.

WCS 点光源 (WCS point light source)（あるいは点光源 (positional light source)はすべての方向に均一に発光し, *point*, *attenuation*, *color* で設定される. *point* は世界座標系での光源位置を示し, *attenuation* は点光源の強さが距離の関数としてどのように減衰するかを示す.

WCS スポット光源 (WCS spot light source)はスポットライト光源のシミュレートであり, *direction*, *point*, *concentration*, *spread_angle*, *attenuation*, *color*によって設定される. *point* は世界座標系における光源の位置, *direction* と *spread_angle* は放射円錐 (cone of influence)を示す. *direction* ベクトルは指定された光源位置に対して相対的である. もし対象となる点が放射円錐の外にある場合には, スポットライトは無関係である.

デプス・キュー *DepthCue(0..65534 , default entry = 1)*

このテーブル型はデプス・キューに関する情報を管理する.

mode	SWITCH
front_plane	FLOAT
back_place	FLOAT
front_scaling	FLOAT
back_scaling	FLOAT
color	COLOR_SPECIFIER

*c1

デプス・キューイングの計算は視点からの距離にしたがって色を変えていくことである. *color* は *front_plane* から *back_plane* に向けて変化していく対象の色を示す. *front_scaling* と *back_scaling* は前方 / 後方デプス・

キューイング面における色のスケールを示す．デプス・キューイングは NPC 空間において実現される．すなわち *front_plane* は NPC 座標系における前方デプス・キューイング面 (front depth cueing plane)への距離，*back_plane* は後方デプス・キューイング面 (back depth cueing plane)への距離である．

色近似 *ColorApprox(0..65534 , default entry = 1)*

このテーブル型はレンダリング・パイプラインにおいて扱われる色値を表示画面上の画素値に変換するルールを指定する．各エントリは以下のデータから構成される．

type	COLOR_APPROX_TYPE
color_model	COLOR_APPROX_MODEL
max1	CARD16
max2	CARD16
max3	CARD16
mult1	CARD32
mult2	CARD32
mult3	CARD32
weight1	FLOAT
weight2	FLOAT
weight3	FLOAT
base_pixel	CARD32
dither	SWITCH

レンダラが輝度計算，デプス・キューイング，クリッピング処理を実行した後，レンダリング・パイプラインの色を表示可能な画素値に変換する必要がある．レンダラの *color_approx_index* の現在値は，この型のテーブルの中のどのエントリを使って，表示可能な画素値に変換するかを決定するために使用される．レンダリング・パイプラインから得られる値は，まず *color_model* によって指定された色空間 (color space)に変換される（レンダリング・パイプラインの色がすでに指定の色空間にある場合は何も処理しない）． *c2

もし *type* が *ColorSpace* ならば，変換後の色（ *c1*， *c2*， *c3*）の各要素は，対応する最大値（ *max1*， *max2*, *max3*）によって正規化される．（ *max1, max2, max3* はクライアントによって使用され，各色軸に対する色マップ (colormap)のエントリ数－１を示している．たとえば，3-3-2 RGB 色マップについて画素値を計算するためには，最大値を７，７，３にする．） 結果の色値は，３つの値の中の第１項と *multi1*，第２項と *multi2*，第３項と *multi3*をそれぞれ掛け合わせて，さらにこれら３つの値の総和に *base_pixel* を加えて，一つの画素値にまとめられる． *type* が *ColorSpace* ならば， *weight1*， *weight2*， *weight3* 値は使用されない． *c2

より具体的な例として， *color_model*が *RGB* に設定されているとする．色近似処理の段階では，レンダリング・パイプラインから得た各色値は一番初めに RGB に変換しなければならない．もしレンダリング・パイプラインが RGB 空間での色計算を実現できるならば，この変換は no-op である． ３要素のそれぞれは次のような整数値にマップされる．

red = 赤色の強さ，範囲 [0, *max1*]
green = 緑色の強さ，範囲 [0, *max2*]
blue = 青色の強さ，範囲 [0, *max3*]

単一の画素値は次のように計算される．

$$pixel = multi1 \times red + multi2 \times green + multi3 \times blue + base_pixel$$

*c2

もし *color_type* が *ColorRange* ならば，色近似処理の段階では違った方法で値が結合される．まず，レンダリング・パイプラインの計算した色は *color_model* によって設定された色空間の色に変換される．最初にレンダリング・パイプラインから得られた色の要素値（ *c1*， *c2*， *c3*）と正規化した重み値（ *weight1*， *weight2*, *weight3*）とを掛け合わせ，その和を単一の画素値とする．重み値は各要素をすべて等しくしたり，一つ以上の要素について最小化あるいは 0 とすることも許される．たとえば， 0.39， 0.59， 0.11 という重み値は， RGB の画素値を NTSC 色標準形式での単一画素値に変換するものである． *c2

次に計算された値は *max1* と掛け合わされる（*max2* と *max3* は用いない）．ここで *max1* は色マップのエントリ数－1を示すもので，クライアントによって設定されていなければならない．たとえば，クライアントが計算した画像をグレー・スケールを使って擬似色ディスプレイに表示する場合において，100 個の連続した色マップのエントリがあり，*max1* は 99 に設定されているならば，画素値は $[0, 99]$ にマッピングされる．この値は *multi1*，*multi2*，*multi3* をそれぞれ掛けることによって，色の3要素に逆変換される．これらの値は画素値となるために，*base_pixel* を加えられる．本来，3つのチャネルを持たないようなドローアブルに対して，妥当な乗数をクライアントは提供しなければならない．たとえば，visual type が *PseudoColor* であるようなドローアブルが目標ならば，乗数（1.0，0.0，0.0）が使用できる．PEX のインプリメンテーションでは，1つ以上の乗数が0であるような場合の最適化については自由である．

画素値の構成については，おおもとの構成が明確なビット・プレーンになっていないような場合に，論理 OR よりも加算の使用が許されている．いくつかのハードウェアは乗数（*multi1*，*multi2*，*multi3*）が2の巾乗であるときに性能向上ができるので，どのくらいの2の巾乗が良いかを知るために，**PEXGetImpDepConstants** リクエストによって定数 *BestColorApproxValues* を得ることができる．

Dither はレンダラに対して，どのようなディザ (dithering)法が良いかを示すヒントである．ディザをサポートするかどうか，また使用するディザのアルゴリズムについてはインプリメンテーション依存である（**PEXGetImpDepConstants** を参照）．

4.1 ルックアップ・テーブル・リソースの管理

ルックアップ・テーブルは X11 のリソースであり，X11 リソースへの管理責任やアクセス権はすべて持ち合わせている．以下のリクエストはルックアップ・テーブル・リソースの作成／解放／複写をおこなう．

4.1.1 ルックアップ・テーブルの作成

名前:

PEXCreateLookupTable

リクエスト:

drawable_example: DRAWABLE_ID
lut_id: LOOKUP_TABLE_ID
table_type: TABLE_TYPE

エラー:

IDChoice, Drawable, Match, Value, Alloc, LookupTable *c2

説明:

本リクエストは *lut_id* で指定されるルックアップ・テーブルを作成する．作成されたルックアップ・テーブルは *drawable_example* で指定されるドローアブルと同じルート・ウィンドウ (root window)と深さ (depth)を持つドローアブルへの表示に使用される． *table_type* パラメータは作成されるテーブルの型を指定する．テーブルのエントリのいくつかは作成時に定義されている．あらかじめ定義されるエントリの数と内容はテーブルの型，およびインプリメンテーションに依存する．

4.1.2 ルックアップ・テーブルの複写

名前:

PEXCopyLookupTable

リクエスト:

src_lut_id: LOOKUP_TABLE_ID
dest_lut_id: LOOKUP_TABLE_ID

エラー:

LookupTable, Match

説明:

本リクエストは *src_lut_id* で指定されるルックアップ・テーブルを， *dest_lut_id* で指定されるルックアップ・テーブルへ複写する． *dest_lut_id* で指定されるルックアップ・テーブルの内容は複写前に消去される． *dest_lut_id* で指定されるルックアップ・テーブルは複写時に有効なリソースとして存在していなければならない． *dest_lut_id* で指定されるルックアップ・テーブルは *src_lut_id* で指定されるルックアップ・テーブルと同じ型でなければならない．

4.1.3 ルックアップ・テーブルの解放

名前:

PEXFreeLookupTable

リクエスト:

lut_id: LOOKUP_TABLE_ID

エラー:

LookupTable

説明:

本リクエストはリソース ID とルックアップ・テーブル間の結合を解除する．解除されたルックアップ・テーブルを参照するものが他になかったならば，ルックアップ・テーブルに割り当てられていた領域は解放される．

4.2 ルックアップ・テーブルへの問い合わせ

以下のリクエストはルックアップ・テーブル属性への問い合わせ機能を提供する.

4.2.1 ルックアップ・テーブル情報を得る

名前:

PEXGetTableInfo

リクエスト:

drawable_example: DRAWABLE_ID
table_type: TABLE_TYPE

リプライ:

definable_entries: CARD16
num_predefined: CARD16
predefined_min: TABLE_INDEX
predefined_max: TABLE_INDEX

エラー:

Drawable, Match, Value, LookupTable *c2

説明:

本リクエストは *table_type* で指定された型のルックアップ・テーブルに関する情報を返す. ただしそれらが *drawable_example* と同じ root と depth を持つドローアブルで使用されていなければならない. *definable_entries* は指定された型のテーブルに定義可能なエントリの最大数を示している. 最大数はあらかじめ定義されるエントリの数を含んだ値である. あらかじめ定義されたエントリをクライアントが変更することが許されている. *predefined_min* と *predefined_max* で指定されるエントリの間に位置するエントリは, あらかじめ定義されることが保証されている. (つまり, あらかじめ定義されるエントリは連続した指標を持つ.) もし *num_predefined* が 0 のときには, *predefined_min* と *predefined_max* に対する指定は意味をなさない.

4.2.2 事前定義されたエントリを得る

名前:

PEXGetPredefinedEntries

リクエスト:

fp_format: FLOAT_FORMAT
drawable_example: DRAWABLE ID
table_type: TABLE_TYPE
start: TABLE_INDEX
count: CARD16

リプライ:

entries: LISTofTABLE_ENTRY

エラー:

Drawable, Match, Value, FloatingPointFormat, LookupTable *c2

説明:

table_type で指定された事前設定されたテーブル・エントリ値を返す. (それらが *drawable_example* と同じ root と depth を持つドローアブルで使用されていなければならない.) 事前設定されていない範囲の各エントリに対しては, 省略値のエントリが返される. *start* で指定したエントリから, *count* の数だけのエントリが *entries* に返される. *entries* の中の値は *table_type* で定義された形式になっている. *entries* の中の浮動小数値は *fp_format* で指定された浮動小数形式になっている.

4.2.3 設定されている指標を得る

名前:

PEXGetDefinedIndices

リクエスト:

lut_id: LOOKUP_TABLE_ID

リプライ:

defined_indices: LISTofTABLE_INDEX

エラー:

LookupTable

説明:

ルックアップ・テーブル・リソース *lut_id* の中で，現在設定されているすべての指標のリストを *defined_indices* に返す．返されるエントリはサーバによって事前設定されたものと，クライアントによって設定されたものの両方を含む．

4.2.4 テーブル・エントリを得る

名前:

PEXGetTableEntry

リクエスト:

fp_format: FLOAT_FORMAT
lut_id: LOOKUP_TABLE_ID
index: TABLE_INDEX
value_type: {*Set*, *Realized*}

リプライ:

status: {*Defined*, *Default*}
table_type: TABLE_TYPE
entry: TABLE_ENTRY

エラー:

LookupTable, FloatingPointFormat, Value

説明:

ルックアップ・テーブルの型と， *index* で指定されたルックアップ・テーブルのエントリを返す．このエントリは *lut_id* で指定されたルックアップ・テーブルから得られ， *lut_id* のテーブル型に示された形式となっている．もしルックアップ・テーブル内で指定されたエントリが未定義ならば，そのテーブル型の省略エントリ値が *entry* に返され， *status* は *Default* に設定される．もし指定のエントリが定義済ならば，その内容は *entry* に返され， *status* は *Defined* に設定される．もし *value_type* が *Set* ならば，返ってくる値はクライアントによって設定された元のものである．もし *value_type* が *Realized* ならば，値はレンダリングの間に実際に使われる値である．（すなわち設定された値がそのドローアブルでサポートされないときに使われる省略値である．） *entry* の中の浮動小数値は *fp_format* で指定される浮動小数形式となっている．

4.2.5 複数のテーブル・エントリを得る

名前:

PEXGetTableEntries

リクエスト:

fp_format: FLOAT_FORMAT
lut_id: LOOKUP_TABLE_ID
start: TABLE_INDEX
count: CARD16
value_type: {*Set*, *Realized*}

リプライ:

table_type: TABLE_TYPE
entries: LISTofTABLE_ENTRY

エラー:

LookupTable, Value, FloatintPointFormat

説明:

lut_id で指定されるルックアップ・テーブルから，*start* で指定されるエントリから始めて，*count* 個のエントリを返す．指定された範囲のエントリが未定義であるならば，省略値を返す．*value_type* が *Set* ならば，クライアントが設定した元の値が返る．*value_type* が *Realized* ならば，レンダリングの間に実際に使われた値が返る．（すなわち設定された値が，ドローアブルでサポートされていないときに使われる省略値である．）*entries* の中の浮動小数値は *fp_format* で指定された浮動小数形式となっている．

4.3 ルックアップ・テーブルの更新

この章ではルックアップ・テーブル・リソースを更新するためのリクエストについて述べる.

4.3.1 テーブル・エントリの代入

名前:

PEXSetTabelEntries

リクエスト:

fp_format: FLOAT_FORMAT
lut_id: LOOKUP_TABLE_ID
start: TABLE_INDEX
count: CARD16
entries: LISTofTABLE_ENTRY

エラー:

LookupTable, Value, FloatingPointFormat, ColorType, Alloc

説明:

lut_id で指定されたルックアップ・テーブル・リソース内の, *start* で指定されたエントリから, *count* 個分のエントリに値を設定する. ルックアップ・テーブルに設定する値は *entries* 内に *lut_id* の型で指定された形式で保持されている. *entries* の中の浮動小数値は *fp_format* で指定される形式となっている.

4.3.2 テーブル・エントリの削除

名前:

PEXDeleteTableEntries

リクエスト:

lut_id: LOOKUP_TABLE_ID
start: TABLE_INDEX
count: CARD16

エラー:

LookupTable, Value

説明:

lut_id で指定されたルックアップ・テーブル・リソース内の, *start* で指定されたエントリから, *start* + *count* − 1 までの連続したエントリを削除する.

5　パイプライン・コンテキスト

パイプライン・コンテキスト (*pipeline context*)は PEX のリソースであり, レンダリング・パイプラインでの表示をおこなう際の属性の実体である. **PEXBeginRendering** リクエストの実行にともない, パイプライン・コンテキストの属性がレンダラに複写される. 本章ではパイプライン・コンテキストの属性と, パイプライン・コンテキストにおいて実行可能な処理について記述している.

本章におけるリクエストのうちのいくつかのものは, パイプライン・コンテキストの属性に影響を持つ. *item_mask* と *item_list* は影響のおよぶ要素を指定する. *item_mask* の各ビットは, ビットに対応する属性が影響されるか否かを示している. パイプライン・コンテキストの属性が設定または問い合わせされるような場合, *item_mask* における各ビットに対応した *item_list* 内のエントリが存在する. そのため, ひとつのリクエストにより, ひとつまたは複数のパイプライン・コンテキストの属性に影響を与えることができる.

名前セット・リソース ID はパイプライン・コンテキストの属性のひとつである. 名前セットが新規に作成されたとき, パイプライン・コンテキストへ結合される. その後解放されても, 名前セットの内容はパイプライン・コンテキストによって参照されるため, そのまま残る. しかし, パイプライン・コンテキストの属性が問い合わせされたときには, 既に参照されるべきリソース ID が無効となっているため, *AlreadyFreed* が名前セット ID に対して返される.

パイプライン・コンテキストで保持する属性には以下のものがある.

属性名	データ型	省略値
(1) マーカ属性		
marker_type	MARKER_TYPE	*MarkerAsterisk*
marker_scale	FLOAT	1.0
marker_color	COLOR_SPECIFIER	{*indexed*,1}
marker_bundle_index	TABLE_INDEX	1
(2) 文字列属性		
text_font_index	TABLE_INDEX	1
text_precision	TEXT_PRECISION	*String*
char_expansion	FLOAT	1.0
char_spacing	FLOAT	0.0
text_color	COLOR_SPECIFIER	{*Indexed*,1}
char_height	FLOAT	0.01
char_up_vector	VECTOR_2D	< 0.0, 1.0 >
text_path	TEXT_PATH	*PathRight*
text_alignment	TEXT_ALIGNMENT	{*HalignNormal*, *ValianNormal*}
atext_height	FLOAT	0.01
atext_up_vector	VECTOR_2D	< 0.0, 1.0 >
atext_path	TEXT_PATH	*PathRight*
atext_alignment	TEXT_ALIGNMENT	{*HalignNormal*, *ValignNormal*}
atext_style	ATEXT_STYLE	*ATextNotConnected*
text_bundle_index	TABLE_INDEX	1
(3) 線属性		
line_type	LINE_TYPE	*LineTypeSolid*
line_width	FLOAT	1.0
line_color	COLOR_SPECIFIER	{*Indexed*,1}
curve_approx	CURVE_APPROX	{1, 1.0}
polyline_interp	POLYLINE_INTERP	*PolylineInterpNone*
line_bundle_index	TABLE_INDEX	1
(4) 領域属性		
interior_style	INTERIOR_STYLE	*InteriorStyleHollow*
interior_style_index	TYPE_OR_TABLE_INDEX	1
surface_color	COLOR_SPECIFIER	{*Indexed*,1}
reflection_attr	REFLECTION_ATTR	{1.0,1.0,1.0,0.0,0.0,{*Indexed*,1}}
reflection_model	REFLECTION_MODEL	*ReflectionNoShading*
surface_interp	SURFACE_INTERP	*SurfaceInterpNone*
bf_interior_style	INTERIOR_STYLE	*InteriorStyleHollow*
bf_interior_style_index	TYPE_OR_TABLE_INDEX	1
bf_surface_color	COLOR_SPECIFIER	{*Indexed*,1}
bf_reflection_attr	REFLECTION_ATTR	{1.0,1.0,1.0,0.0,0.0,{*Indexed*,1}}
bf_reflection_model	REFLECTION_MODEL	*ReflectionNoShading*
bf_surface_interp	SURFACE_INTERP	*SurfaceInterpNone*

属性名	データ型	省略値
(4) 領域属性 (続き)		
surface_approx	SURFACE_APPROX	{1,1.0,1.0}
culling_mode	CULL_MODE	*None*
distinguish	BOOLEAN	*False*
pattern_size	VECTOR_2D	{1.0,1.0}
pattern_ref_pt	COORD_3D	{0.0,0.0,0.0}
pattern_ref_vec1	VECTOR_3D	< 1.0,0.0,0.0 >
pattern_ref_vec2	VECTOR_3D	< 0.0,1.0,0.0 >
interior_bundle_index	TABLE_INDEX	1
(5) 外周線属性		
surface_edges	SWITCH	*Off*
surface_edge_type	SURFACE_EDGE_TYPE	*SurfaceEdgeSolid*
surface_edge_width	FLOAT	1.0
surface_edge_color	COLOR_SPECIFIER	{*Indexed*,1}
edge_bundle_index	TABLE_INDEX	1
(6) 幾何変換属性		
local_transform	MATRIX	identity
global_transform	MATRIX	identity
model_clip	CLIP_INDICATOR	*NoClip*
model_clip_volume	LISTofHALF_SPACE	*Null*
view_index	TABLE_INDEX	0
(7) 色変換属性		
light_state	LISTofTABLE_INDEX	*Null*
depth_cue_index	TABLE_INDEX	0
color_approx_index	TABLE_INDEX	0
rdr_color_model	COLOR_MODEL	0
psurf_char	PSURF_CHAR	{1,*NULL*}
(8)ASF 属性		
asfs	ASF_SPECIFIER	{0,all *Individual*}
(9) その他の属性		
pick_id	CARD32	0
HLHSR_identifier	CARD32	0
name_set [1]	NAME_SET_ID	*None*

[1] パイプライン・コンテキストの属性がレンダラにコピーされるとき (すなわち **PEXBeginRendering** リクエストが発生するとき), 名前セット・リソースの実際の中身がコピーされるが, 名前セットのリソース ID はコピーされない.

属性名	データ型	省略値
ASF_SPECIFIER の *asfs* フィールド内の各ビットは次のように決めてある.		
marker_type_asf	AFS_VALUE	*Individual*
marker_scale_asf	AFS_VALUE	*Individual*
marker_color_asf	AFS_VALUE	*Individual*
text_font_index_asf	AFS_VALUE	*Individual*
text_precision_asf	AFS_VALUE	*Individual*
char_expansion_asf	AFS_VALUE	*Individual*
char_spacing_asf	AFS_VALUE	*Individual*
text_color_asf	AFS_VALUE	*Individual*
line_type_asf	AFS_VALUE	*Individual*
line_width_asf	AFS_VALUE	*Individual*
line_color_asf	AFS_VALUE	*Individual*
curve_approx_asf	AFS_VALUE	*Individual*
polyline_interp_asf	AFS_VALUE	*Individual*
interior_style_asf	AFS_VALUE	*Individual*
interior_style_index_asf	AFS_VALUE	*Individual*
surface_color_asf	AFS_VALUE	*Individual*
reflection_model_asf	AFS_VALUE	*Individual*
surface_interp_asf	AFS_VALUE	*Individual*
reflection_attr_asf	AFS_VALUE	*Individual*
bf_interior_style_asf	AFS_VALUE	*Individual*
bf_interior_style_index_asf	AFS_VALUE	*Individual*
bf_surface_color_asf	AFS_VALUE	*Individual*
bf_reflection_model_asf	AFS_VALUE	*Individual*
bf_surface_interp_asf	AFS_VALUE	*Individual*
bf_reflection_attr_asf	AFS_VALUE	*Individual*
surface_approx_asf	AFS_VALUE	*Individual*
surface_edges_asf	AFS_VALUE	*Individual*
surface_edge_type_asf	AFS_VALUE	*Individual*
surface_edge_width_asf	AFS_VALUE	*Individual*
surface_edge_color_asf	AFS_VALUE	*Individual*

パイプライン・コンテキスト・リソースの属性は次のように定義されている.

marker_type
マーカ要素を表示するときに使用するマーカの型である. 登録されているマーカ型の詳細は「拡張情報」の章を参照.

marker_scale
マーカ要素を表示するときに使用するマーカのスケール・ファクタである.

marker_color
マーカ要素を表示するときに使用する色値である.

marker_bundle_index
マーカ束テーブルからバンドルされたマーカの属性を得るためのルックアップ・テーブルの指標である.

text_font_index
文字列と注釈文字列を表示するフォント ID を得るためのルックアップ・テーブルの指標である.

text_precision
文字列と注釈文字列を表示するときの精度である.

char_expansion
文字列を表示する際の拡大率 (character expansion)である. このファクタはフォント設計者が設定した幅／高さの比率からの偏差を意味する.

char_spacing
文字列を表示する際の文字間隔 (character spacing)である. 文字間隔は隣接する文字の間にどのくらいの空白を挿入するかを示し, 標準フォントの文字高に対する比率で表される.

text_color
文字列と注釈文字列の表示の際の色値である.

char_height
文字列を表示する際の文字高 (character height)である. 文字高はモデリング座標系で設定される.

char_up_vector
文字列を表示する際の文字上方向ベクトル (character up vector)である. このベクトルは文字列のローカル座標系で設定される. この座標系の軸は文字列に対して設定されている方向ベクトルによって決定される.

text_path
文字列を表示する際の文字列方向 (text path)である. (すなわち文字列の記入方向である.)

text_alignment
文字列を表示する際の水平／垂直配置 (text alignment)である.

atext_height
注釈文字列を表示する際の文字高である. 文字高は正規化投影座標系で設定される.

atext_up_vector
注釈文字列を表示する際の文字上向きベクトルである. このベクトルは文字列のローカル座標系（ディスプレイに対して平行である）で設定される.

atext_path
注釈文字列を表示する際の文字列方向である. (すなわち文字列の記入方向である.)

atext_alignment
注釈文字列を表示する際の水平／垂直配置である.

atext_style
注釈文字列を表示する際の文字列様式である. 登録された文字列様式については「拡張情報」の章を参照.

text_bundle_index

文字列束テーブルからのバンドルされた属性値を得るためのルックアップ・テーブルの指標である.

line_type

折線と曲線要素を表示する際の型である. 登録された線の型については「拡張情報」の章を参照.

line_width

折線と曲線要素を表示する際の線幅である. これは表示されるときの線幅に対応したスケール・ファクタである. 線幅は要素が 3D 空間から 2D ラスタ空間に変換された後の 2D 座標系で設定される.

line_color

折線と曲線要素を表示する際の色値である.

curve_approx

曲線要素を表示する際の曲線近似 (curve approximation)属性である. これは曲線近似法と描く曲線の誤差値を設定する. 登録されている曲線近似法と各手法に適用される誤差値については「拡張機能」の章を参照.

polyline_interp

折線要素を表示する際の折線シェーディング法 (polyline interpolation method)である. 登録されているシェーディング法については「拡張機能」の章を参照.

line_bundle_index

線束テーブルから折線と曲線のバンドルされた属性を得るためのルックアップ・テーブルの指標である.

interior_style

領域要素を表示する際の内部様式 (interior style)である. 登録されている内部様式については「拡張機能」の章を参照.

interior_style_index

この属性は内部様式が *Pattern* か *Hatch* のときに使う指標である. 内部様式指標 (interior style index)は, 内部様式が *Pattern* のときはパターン・テーブル・エントリの指標を示し, *Hatch* のときはハッチ・テーブルの列挙型指標を示す.

surface_color

領域要素 (surface primitive)を表示する際の色値である.

reflection_attr

領域 (面積が定義されている要素) を表示する際の光源シェーディングに関する, 環境係数 (ambient coefficient), 拡散係数 (diffuse coefficient), 鏡面係数 (specular coefficient), 鏡面集中指数 (specular concentration), 鏡面色 (specular color), 透過係数 (transmission coefficient)である. 鏡面色属性は, 鏡面反射計算において一次色に加算されていく係数であることを規定している. このことから光源色よりも何か他の色からハイライトが計算されることになる. 鏡面集中指数は鏡面のハイライト (あるいは表面の輝き) の鋭さを定義している. この値は通常は照明方程式の鏡面反射項において exp として用いられる. この値の範囲は 0 から最大の浮動小数値までである. もし *specular_conc* が 0 ならば, 鏡面ハイライトはたいへん広くなる. もし *specular_conc* が 0 よりきわめて大きいならば, あたかも表面が輝いているかのように, ハイライトはたいへん小さく, かつシャープとなる. 透過係数は表面を通過する光の量を示す. 透過係数が 0 のときは表面が不透明 (opaque)である. 透過係数が 1.0 のときは表面は全体的に見えない (すべての光が透過してしまう).

reflection_model

領域要素を表示する際の反射モデル (reflection model)である. 登録されている反射モデルについては「拡張機能」の章を参照.

surface_interp

領域要素を表示する際の領域シェーディング法 (surface interpolation method)である. 登録されているシェーディング法については「拡張機能」の章を参照.

bf_interior_style

裏面領域要素 (backfacing surface primitive)を表示する際の内部様式である．登録されている内部様式については「拡張機能」の章を参照.

bf_interior_style_index

内部様式が *Pattern* か *Hatch* のときの裏面領域要素に対する指標である．内部様式指標は内部様式が *Pattern* のときはパターン・テーブルへの指標となり，内部様式が *Hatch* のときはハッチ・テーブルの列挙型指標となる.

bf_surface_color

裏面領域要素を表示する際の色値である.

bf_reflection_attr

裏面領域要素を表示する際の反射値 (reflection value)である.

bf_reflection_model

裏面領域要素を表示する際の反射モデルである．登録されている反射モデルについては「拡張機能」の章を参照.

bf_surface_interp

裏面領域要素を表示する際の領域シェーディング法である．登録されているシェーディング法については「拡張機能」の章を参照.

surface_approx

領域要素を表示する際の曲面近似法 (surface approximation)と誤差値である．登録されている近似法とそれに対応した誤差許容値については「拡張情報」の章を参照.

culling_mode

裏面領域を処理する際の識別モード (culling mode)である． *BackFaces* ならば，すべての裏面領域が対象外となり，表面の領域のみが表示される． *FrontFaces* ならば，すべての表面領域が対象外となり，裏面領域のみが表示される． *None* ならば，表面と裏面の領域すべてが表示される.

distinguish

裏面領域の表示の際の識別フラグである．このフラグは裏面領域の表示で，裏面領域の属性か表面領域の属性のどちらを使うかを選択する． *distinguish* が *False* ならば表面属性を表示に用いる.

pattern_size

領域要素を表示するときに用いるパターンの寸法である．モデリング座標系で指定される．値$< x,\ 0 >$はパターンの幅ベクトルとして用いられ，値$< 0,\ y >$はパターンの高さベクトルとして用いられる．もし内部様式が *Pattern* ならば，レンダラは領域上のパターンの位置，スケール，回転角などを求めるために，これらの値と，パターン参照位置 (pattern reference point)，パターン参照ベクトル (pattern reference vector)を使用する. *c2

pattern_ref_pt

領域要素を表示する際のパターン参照ベクトルである．モデリング座標系で指定される．内部様式が *Pattern* ならば，レンダラは領域上のパターンの位置とスケールを知るためにパターン参照位置とパターン参照ベクトル，およびパターン寸法を用いる. *c2

pattern_ref_vec1

内部様式が *Pattern* のときに用いる 2 つのパターン参照ベクトルのうちの最初のものである．モデリング座標系で指定される．内部様式が *Pattern* のとき，レンダラは領域上のパターンの位置とスケールと回転角などを知るために，2 つのパターン参照ベクトルとパターン参照位置，およびパターン寸法を用いる. *c2

pattern_ref_vec2

内部様式が *Pattern* のときに用いる 2 つのパターン参照ベクトルのうちの 2 番目のものである．モデリング座標系で指定される．内部様式が *Pattern* のとき，レンダラは領域上のパターンの位置とスケールと回転角などを知るために，2 つのパターン参照ベクトルとパターン参照位置，およびパターン寸法を用いる. *c2

interior_bundle_index

内部様式束テーブルからバンドルされた内部様式属性を得るためのルックアップ・テーブル指標である.

surface_edges

外周線 (surface edge)の表示が可能／不可のいずれかを示す外周線フラグ属性である. もし *surface_edges* が *Off* ならば, 外周線表示は不可である. *surface_edges* が *On* ならば, 外周線表示は可能である. 外周線は外周線色 (surface edge color), 外周線型 (surface edge type), 外周線幅 (surface edge width)を使って表示される.

surface_edge_type

外周線を表示する際の外周線型である. 登録されている外周線型については「拡張機能」の章を参照.

surface_edge_width

外周線を表示する際の外周線幅である. これは外周線が表示されるときの外周線の幅に対するスケール・ファクタである. 一つの外周線要素が 3D 空間から 2D ラスタ空間に変換された後, 2D 座標系で適用される.

surface_edge_color

外周線を表示する際の色値である.

edge_bundle_index

外周線束テーブルからバンドルされた外周線属性を得るためのルックアップ・テーブル指標である.

local_transform

要素を表示する際のローカル・モデリング変換行列である.

global_transform

要素を表示する際のグローバル・モデリング変換行列である.

model_clip

要素を表示する際, モデル・クリッピングを実行するか否かを示すフラグである.

model_clip_volume

モデル・クリッピングが可能かどうかによって使用されるモデル・クリッピング・ボリュームである.

view_index

視野テーブルから視野属性を得るためのルックアップ・テーブル指標である.

light_state

使用可能な（「点灯している」）光源を示す指標のリストである. このリスト中にない光源は使用不可（「点灯していない」）とする.

depth_cue_index

デプス・キュー束テーブルからデプス・キュー属性を得るためのルックアップ・テーブル指標である.

color_approx_index

色近似テーブルから色近似パラメータを得るためのルックアップ・テーブル指標である.

rdr_color_model

要素のシェーディングをおこなう間の表示色モデルである. 登録される色モデルについては「拡張情報」の章を参照.

psurf_char

領域を表示する際のパラメトリック曲面特性 (parametric surface characteristics)である. *psc_type* はパラメトリック曲面特性の型を示し, *psc_data* は追加情報（もしあれば）である. 登録されているパラメトリック曲面特性については「拡張情報」の章を参照.

asfs

ルックアップ・テーブルから得られる各属性に対する ASF(aspect source flag) である．もし ASF が *Individual* に設定されていれば，対応する属性は表示中は現在値から直接得られる． ASF が *Bundled* に設定されていれば，属性値はルックアップ・テーブルの束から得られる． ASF の設定にあたっては，分離マスク（*enables*）がどの ASF が実際に変更されるかを示す．変更される ASF の値はパイプライン・コンテキストの属性を生成したり，変更するときのみ意味がある． ASF_SPECIFIER の *enable* フィールドはパイプライン・コンテキストの属性を生成したり，変更するときのみ意味を持つ．コピーについては，すべての ASF 値はコピーされる．問い合わせのときは，すべての ASF 値が返り，すべての定義された ASF の値は *enables* ビットマスクに返る．（このことは *enable* フィールドが表示の間に退避されたり，復元されたりする状態の一部として考える必要がないことを意味する．）

pick_id

ピッキング操作 (picking operation)に付随したピック ID である．

HLHSR_identifier

HLHSR 識別子を示す．これは， *ZBufferId* を除くすべての登録された HLHSR モードに対して，何もしない．レンダラの *HLHSR_mode* が *ZBufferId* に設定されているときは，0 値は z- バッファリングは無効となり，1 値は z- バッファリングが有効となる．登録されていない HLHSR モードに対する意味付けはインプリメンテーション依存である．概念的には，この属性は，すべての出力プリミティブがレンダリング・パイプラインに入るときに結合される． *c1

name_set

名前セット・リソースへの参照ポインタである．パイプライン・コンテキストの属性がレンダラにコピーされるとき，この名前セットの *contents* がレンダラにコピーされる．

5.1 パイプライン・コンテキスト・リソースの管理

パイプライン・コンテキストは X11 のリソースであり, X11 における管理責任や使用権のすべてを継承している. 以下に記述するリクエストは生成, 解放, 複写等の機能を提供する.

5.1.1 パイプライン・コンテキストの生成

名前:

PEXCreatePipelineContext

リクエスト:

fp_format: FLOAT_FORMAT
pc_id: PIPELINE_CONTEXT_ID
item_mask: PC_BITMASK
item_list: LISTofVALUE

エラー:

IDChoise, Value, FloatingPointFormat, ColorType, Alloc, NameSet *p

説明:

本リクエストは *pc_id* に指定されたパイプライン・コンテキスト・リソースを生成する. *item_mask* はリソースが作成されるとき, 明示的に設定されるパイプライン・コンテキスト属性を指定する. *item_list* に新設されるパイプライン・コンテキスト属性に設定される属性値が含まれている. *item_list* に登録される浮動小数型の属性値は *fp_format* で指定される浮動小数形式でなければならない. 同様に, *item_list* に含れる色値は *color_type* に指定される色型でなければならない.

5.1.2 パイプライン・コンテキストの複写

名前:

PEXCopyPipelineContext

リクエスト:

src_pc_id: PIPELINE_CONTEXT_ID
dest_pc_id: PIPELINE_CONTEXT_ID
item_mask: PC_BITMASK

エラー:

PipelineContext, Value

説明:

本リクエストは *src_pc_id* のパイプライン・コンテキストを, *dest_pc_id* のパイプライン・コンテキストへ複写する. *dest_pc_id* で指定されるパイプライン・コンテキストは複写の時点で既に有効なリソースとして存在していなければならない. *item_mask* はパイプライン・コンテキスト中のどの属性が複写されるかを指定する.

5.1.3 パイプライン・コンテキストの解放

名前:

PEXFreePipelineContext

リクエスト:

pc_id: PIPELINE_CONTEXT_ID

エラー:

PipelineContext

説明:

本リクエストはパイプライン・コンテキストの実体とリソース ID との関係を解消する. パイプライン・コンテキスト・リソースに割り当てられていた領域は, 他のリソースがこのパイプライン・コンテキスト・リソースを参照していなければ解放される.

5.2 パイプライン・コンテキスト・リソースへの問い合わせ

パイプライン・コンテキスト属性への問い合わせ機能を記述する.

5.2.1 パイプライン・コンテキスト属性を得る

名前:

PEXGetPipelineContext

リクエスト:

fp_format: FLOAT_FORMAT
pc_id: PIPELINE_CONTEXT_ID
item_mask: PC_BITMASK

リプライ:

item_list: LISTofVALUE

エラー:

PipelineContext, FloatingPointFormat, Value

説明:

本リクエストは *pc_id* で指定されるパイプライン・コンテキスト内の属性を通知する. *item_mask* はどの属性が問い合わされ, 通知するのかを決定する. 指定された属性は *item_list* 中に返される. *item_list* 内に返される属性で浮動小数型のものは *fp_format* で指定される形式をしていなければならない.

5.3 パイプライン・コンテキストの変更

パイプライン・コンテキスト属性を変更するためのリクエストを記述する.

5.3.1 パイプライン・コンテキスト属性の変更

名前:

PEXChangePipelineContext

リクエスト:

fp_format: FLOAT_FORMAT
pc_id: PIPELINE_CONTEXT_ID
item_mask: PC_BITMASK
item_list: LISTofVALUE

エラー:

PipelineContext, Value, FloatingPointFormat, ColorType, NameSet *p

説明:

本リクエストは *pc_id* で指定されるパイプライン・コンテキスト内の属性を変更する. *item_mask* はどの属性が変更されるのかを指定する. 変更される属性の変更値は *item_list* に指定される. *item_list* 内で浮動小数型のものは *fp_format* で指定される形式をしていなければならない. 同様に, *item_list* に含まれる色値は *color_type* に指定される色型でなければならない.

6 レンダラ

レンダラ (renderer)は 3 次元レンダリングのために作成される PEX リソースである．レンダラ・リソースは各種テーブルと名前セットの ID 群，レンダリング・パイプライン属性の初期値として複写されるパイプライン・コンテキスト・リソースの ID，およびその他の属性からなっている．レンダラは隠線 / 隠面処理において必要となる HLHSR バッファの管理もおこなう．レンダラは **PEXBeginRendering** リクエストによりレンダリング・モードとなる．（**PEXBeginRendering** リクエストは，**PEXRenderNetwork** リクエストの一部として暗黙的に動作する．） PEX のインプリメンテーションでは，特定の属性を **PEXBeginRendering** がリクエストされたとき，およびレンダラが有効である期間だけ，レンダラに結合することが可能である．ある PEX のインプリメンテーションにおいて，どのようにレンダラ属性が結合されるかを知るためには, **PEXGetRendererDynamics** リクエストを用いる．

いくつかのレンダラ・リソース・リクエストは *item_mask* パラメータを必要とする． *item_mask* の各ビットは，対応する属性が設定／問い合わせされるか否かを示す． *item_mask* の設定された各ビットに対応するエントリが *item_list* の中に存在する．したがって一回のリクエストにおいて複数個のレンダラ属性を設定／問い合わせすることができる．

レンダラ・リソースが生成されるとき，レンダラ属性のうち明示的には指定されないいくつかのリソース ID は *Null* 値に設定される．レンダリング処理中，レンダラが *NULL*-ID のルックアップ・テーブルから属性を読み出そうとすると，そのテーブルでの属性の省略値が使用される．（これらは Appendix D に掲載した．） *NULL*-ID の名前セット・リソースから読み出しをおこなおうとすると，結果は名前セットが空であったときと同じになる．レンダラ内のパイプライン・コンテキスト属性でリソース ID が *Null* であるリソースを持つ場合には，デフォルトのパイプライン・コンテキスト属性値が **PEXBeginRendering** リクエスト発行ごとにレンダラ内に写される．

パイプライン・コンテキスト，ルックアップ・テーブル，および名前セットの各リソースはレンダラ・リソースに結合され，また解放される．この一連の処理の終了後，上記の各リソースの内容はレンダラの参照のため，そのまま残される．しかしながら，レンダラ属性への問い合わせがなされたときには，既に解放され参照すべきリソース ID が存在しないことから *AlreadyFreed* が返される．

あるレンダラに関係したウィンドウが， *Rendering* 状態中に破壊 (destroy)，またはリサイズ (resize)されたときには，サーバにより *flush* が *False* に設定されるとともに， **PEXEndRendering**リクエストが暗黙的に発行され，レンダラはアイドル・モードとなる．以下に続く出力 / 探訪コマンド（たとえば， **PEXRenderOutputCommands**, **PEXBeginStructure**, **PEXEndStructure**, **PEXRenderElements**, **PEXAccumulateState**）は，最後の **PEXEndRendering** を含めすべて無視される．エラーは生じない．しかし， z- バッファの再割り当てができなかったとき，次の **PEXBeginRendering** において *Alloc* エラーが発生する． *c1

Rendering 状態において，ウィンドウの移動 (move)がなされ，ウィンドウが露出 (expose)または隠蔽 (occlude)されたときには，新たなウィンドウ階層の下で明示的 / 暗黙的な **PEXEndRendering** リクエストが発行されるまで，出力コマンドの処理は継続させなければならない． *c1

明示的 / 暗黙的な **PEXBeginRendering** リクエストが生じ，かつ *HLHSR_mode* が z- バッファの使用を示唆していたならば，指定されたドローアブルに合った z- バッファが用意され，レンダラに結合されなければならない．（レンダラに対応した z- バッファがなければ，あるいはレンダラに結合した z- バッファが現在のドローアブルに不適合ならば，新しい z- バッファがアロケートされるのみである．） z- バッファが用意できなかったときには, *Alloc* エラーが発生し，レンダリングは中止される．もし *clear_Z* が有効ならば，あるいは新しい z- バッファが領域確保されたならば， z- バッファはレンダラの現在のクリップ・リストによって設定された領域の中で無限大（最大値）にクリアされる．要素が同じ z 値を持つときは，どの要素が表示されるかはインプリメンテーション依存である．最大値がいくつになるかはインプリメンテーション依存である．

レンダリング処理中，図形要素はレンダラのクリップ・リストによりクリップされる． z- バッファリングが有効となっているときには，視点に近い要素が遠い要素に上書きされる．要素が同じ z 値を持っていたとき，いずれの要素が表示されるかはインプリメンテーション依存である．

明示的 / 暗黙的な **PEXEndRendering** リクエストが発行されたとき， z- バッファは解放されない． z- バッファはレンダラが解放されるまでは解放されない．続けて **PEXBeginRendering** リクエストが発行されたときには，同じ z- バッファが使用される．同一の z- バッファが使用できなかった（たとえば，前のドローアブルと現ドローアブルのサイズが異なっているとかの）ときには，現 z- バッファは解放され，新たなドローアブルに適応した z- バッファが用意される． *c1

以下のテーブルにレンダラ・リソースの構成要素を列記する．略記 "imp. dep. " は， "implementation-dependent" の省略値である． *c1

属性名	データ型	省略値
pipeline_context	PIPELINE_CONTEXT_ID	*Null*
current_path	LISTofELEMENT_REF	*Null*
renderer_state	RENDERER_STATE	*Idle*
marker_bundle	LOOKUP_TABLE_ID	*Null*
text_bundle	LOOKUP_TABLE_ID	*Null*
line_bundle	LOOKUP_TABLE_ID	*Null*
interior_bundle	LOOKUP_TABLE_ID	*Null*
edge_bundle	LOOKUP_TABLE_ID	*Null*
view_table	LOOKUP_TABLE_ID	*Null*
color_table	LOOKUP_TABLE_ID	*Null*
depth_cue_table	LOOKUP_TABLE_ID	*Null*
light_table	LOOKUP_TABLE_ID	*Null*
color_approx_table	LOOKUP_TABLE_ID	*Null*
pattern_table	LOOKUP_TABLE_ID	*Null*
text_font_table	LOOKUP_TABLE_ID	*Null*
highlight_inclusion	NAME_SET_ID	*Null*
highlight_exclusion	NAME_SET_ID	*Null*
invisibility_inclusion	NAME_SET_ID	*Null*
invisibility_exclusion	NAME_SET_ID	*Null*
pick_inclusion	NAME_SET_ID	*Null*
pick_exclusion	NAME_SET_ID	*Null*
pick_start_path	LISTofELEMENT_REF	*Null*
background_color	COLOR_SPECIFIER	{*Indexed*, 0}
clear_I	BOOLEAN	*False*
clear_Z	BOOLEAN	*True*
echo_mode	ECHO_MODE	*NoEcho*
HLHSR_mode	HLHSR_MODE	1
NPC_subvolume	NPC_SUBVOLUME	{(0.0,0.0,0.0),(1.0,1.0,1.0)}
viewport	VIEWPORT	{imp.dep., imp.dep., True}
clip_list	LISTofDEVICE_RECT	*Null*

レンダラ・リソースの各属性は次のように定義されている.

pipeline_context
PEXBeginRendreing リクエストが明示的／暗黙的に実行されるとき，レンダリング・パイプラインの初期属性としてコピーされるパイプライン・コンテキストのリソース ID である.

current_path
クライアント側探訪 (client-side traversal)のパス経路 (track of paths)を保持するための，要素リストである.

renderer_state
レンダラの現在状態である．明示的／暗黙的な **PEXBeginRendering** リクエストが処理されるときは， *renderer_state* は *Rendering* に設定され，明示的／暗黙的な **PEXBeginPickOne** あるいは **PEXBeginPickAll** リクエストが処理されるときは， *Picking* に設定される．また明示的／暗黙的な **PEXEndRendering**, **PEXEndPickOne**, **PEXEndPickAll** リクエストが処理されるときは， *Idle* に設定される. *c1

marker_bundle
レンダリングする際のマーカ束ルックアップ・テーブルのリソース ID である.

text_bundle
レンダリングする際の文字列束ルックアップ・テーブルのリソース ID である.

line_bundle
レンダリングする際の線束ルックアップ・テーブルのリソース ID である.

interior_bundle
レンダリングの際の内部様式束ルックアップ・テーブルのリソース ID である.

edge_bundle
レンダリングの際の外周線束ルックアップ・テーブルのリソース ID である.

view_table
レンダリングの際の視野ルックアップ・テーブルのリソース ID である.

color_table
指標色を参照するときの色ルックアップ・テーブルのリソース ID である.

depth_cue_table
レンダリングの際のデプス・キュー・ルックアップ・テーブルのリソース ID である.

light_table
レンダリングの際の光源ルックアップ・テーブルのリソース ID である.

color_approx_table
レンダリング・パイプラインの色近似処理に使われる色近似ルックアップ・テーブルのリソース ID である.

pattern_table
パターンを参照する際のルックアップ・テーブルのリソース ID である.

text_font_table
文字列フォントを参照する際のルックアップ・テーブルのリソース ID である.

highlight_inclusion
顕示包含集合 (highlight inclusion set)として使う名前セットのリソース ID である.

highlight_exclusion
顕示除外集合 (highlight exclusion set)として使う名前セットのリソース ID である.

invisibility_inclusion
不可視包含集合 (invisibility inclusion set)として使う名前セットのリソース ID である.

invisibility_exclusion

不可視除外集合 (invisibility exclusion set)として使う名前セットのリソース ID である.

pick_inclusion *c1

ピック包含フィルタ (pick inclusion filter)として使う名前セットのリソース ID である.

pick_exclusion *c1

ピック除外フィルタ (pick exclusion filter)として使う名前セットのリソース ID である.

pick_start_path *c1

次のサーバ側の「全ピック (pick all)」トラバーサルを開始するための階層パス (hierarchical path)を示す. **PEXPickAll**はこのパスで指定された箇所の次の要素から, ヒット・テスト (hit test)を開始する. **PEXPickAll** と **PEXEndPickAll** ともにこの属性を, 「全ピック」探訪の最後の要素のパスに設定する.

background_color *c1

画像バッファをクリアする際に使う色値である. *clear_I* が有効ならば, 画像バッファは **PEXBeginRendering** によってクリアされる. 初期パイプライン・コンテキスト中の *color_approx_index* が, 色を画素値にマップするために使われる.

clear_I *c1

PEXBeginRendering の際に画像バッファを背景色でクリアするか否かを示す属性である. レンダラのクリップ・リスト内の画素だけがクリアされる.

clear_Z *c1

レンダリング開始時に z- バッファを無限大（無限大に対応した値は platform 依存である）に設定するか否かを示す属性である. レンダラのクリップ・リスト中にある画素のみがクリアされる. この属性は *HLHSR_mode* が z- バッファを必要としないならば, 無視される.

echo_mode *c1

要素がエコーされるか, されないか, あるいは通常の表示かを示す属性である. 要素をエコーするため, あるいはエコーしないための方法はインプリメンテーション依存である. 採用した方法が一時的な優先度を保持することは保証されない. エコーは顕示に対して優位にある. エコーと顕示の方法は見かけの上で区別できることが望ましいが, 不可欠な条件ではない.

HLHSR_mode

オーバーラップした要素を表示する際の隠線／隠面処理を示す. *renderer_state* 属性が *Rendering* に設定されている間, *HLHSR_mode* の変更が変化をもたらすか否かは, **PEXGetRendererDynamics**リクエストが決定できる.

NPC_subvolume

ビューポート (viewport)にマップされる sub-volume 空間に対する正規化投影座標系 (normalized project coordinates)を示す.

viewport

3D グラフィックスの要素が表示されるドローアブル内の領域を示す. ビューポートは装置座標系内に設定される. これはドローアブルの左下隅を原点とし, x, y, offset（単位は画素）で指定される. ドローアブルの外側に境界をもつビューポートを設定してもよい. ドローアブルの外側の値にマッピングされた図形はクリップされる. ビューポートの z 値は [0-1.0] の範囲になければならない. ここで z = 0 は視点から最も遠い位置の装置 z 座標にマップされる. z = 1 は視点座標に最も近い位置の装置 z 座標にマップされる. *viewport* 属性（**PEXGetRendererDynamics**参照）のダイナミックス (dynamics)に依存して, ビューポートの定義値はいつも結合されているかもしれないし, また **PEXBeginRendering** のときのみ効果をもたらすかもしれない. ビューポート値が結合されているときは, ビューポートの *use_drawable* フラグは最初に確認される. それが *True* に設定されていれば, ビューポートの幅と高さはドローアブルの現在の幅と高さから得られる. ビューポートの z 値は VIEWPORT ストラクチャの中の設定値から得られる. もし *use_drawable* フラグが *False* ならば, ビューポートの幅, 高さ, z 値はすべて ビューポート・ストラクチャ内の値に設定される. このビューポートは左下隅を固定し, レンダラの *NPC_subvolume* に等方マッピング (isotropic map- *c2 *c2

ping)された最大の矩形に設定される．リサイズ (resize)の事象が発生した後のビューポートは，ドローアブルの左下隅に対して同じ位置に残る．

clip_list

レンダリングが有効なドローアブルの領域を定義した（装置座標系での）矩形のリストである．もしリストが *Null* ならば，ドローアブル内のすべての画素はレンダリングによって上書きされるかもしれない．この矩形は重なっていけない．さもなければ表示結果は定義されない． *clip_list* 内のすべての矩形の外側にある画素は，実質的に「書き込みプロテクト (write-proteced)」される．リスト内の矩形は装置座標系（0，0 はウィンドウの左下隅にある）で定義される．またリサイズが発生した後は，ドローアブルの左下隅に対して同じ位置に残る．（注意：もし z- バッファリングが使用されるならば，**PEXBeginRendering** や **PEXRenderNetwork** リクエストがあったときには，リストに登録されている矩形内の画素のみが初期化された z 値を持つ．）

6.1 レンダラ・リソースの管理

レンダラは X11 のリソースであり，X11 における管理責任や使用権のすべてを継承している．以下に記述するリクエストはレンダラ・リソースに対する生成／解放などを管理する．

6.1.1 レンダラ・リソースの生成

名前:

PEXCreateRenderer

リクエスト:

fp_format: FLOAT_FORMAT
rdr_id: RENDERER_ID
drawable_example: DRAWABLE_ID
item_mask: BITMASK
item_list: LISTofVALUE

エラー:

IDChoice, Drawable, PipelineContext, NameSet, LookupTable, FloatingPointFormat, Value, Alloc, Match

説明:

本リクエストは *rdr_id* で指定されるレンダラ・リソースを作成する．レンダラは *drawable_example* で指定されるドローアブルと同じ root と depth を持つドローアブルに対して，レンダリングをおこなうために使用される． *item_mask* はリソースが作成されるとき，明示的に設定されるレンダラ属性を指定する． *item_list* には，新設されるレンダラ属性に設定される属性値が含まれている． *item_list* に登録される浮動小数型の属性値は， *fp_format* に指定される浮動小数型でなければならない．

6.1.2 レンダラ・リソースの解放

名前:

PEXFreeRenderer

リクエスト:

rdr_id: RENDERER_ID

エラー:

Renderer

説明:

本リクエストは指定されたレンダラ・リソースを削除し，このレンダラに割り当てられていた領域を解放する．レンダラの *renderer_state* が *Rendering* に設定されている状態で本リクエストが発行されたときには，暗黙的に **PEXEndRendering**リクエストが， *flush* を *False* に設定した状態で，レンダラの解放前に発行される．もしこのリクエストが処理されるとき，レンダラの *renderer_state* が *Picking* に設定されていれば，ピック操作が中断される． *c1

6.2 レンダラ・リソースの変更

ここでのリクエストはレンダラ属性の変更をおこなう.

6.2.1 レンダラ・リソースの変更

名前:

PEXChangeRenderer

リクエスト:

fp_format: FLOAT_FORMAT
rdr_id: RENDERER_ID
item_mask: BITMASK
item_list: LISTofVALUE

エラー:

Renderer, Match, Value, FloatingPointFormat, NameSet, LookupTable, PipelineContext

説明:

本リクエストはレンダラの構成要素（属性）を変更する. *item_mask* と *item_list* はどの属性を変更するかを指定する. *item_mask* の各ビットは, 対応する属性エントリが *item_list* の中にあるか否かを示している. したがって, 一度の **PEXChangeRenderer** リクエストにおいて, ひとつまたは複数の属性変更の要求が可能である. レンダラの *current_path* と *renderer_state*属性は読込み専用である. それゆえ, 両者に対する変更要請は無視される. *item_list* の中の浮動小数型の属性値は, *fp_format* に指定される浮動小数型でなければならない.

6.3 レンダラ・リソース属性の問い合わせ

ここでのリクエストはレンダラ属性の問い合わせをおこなう.

6.3.1 レンダラ属性を得る

名前:

PEXGetRendererAttributes

リクエスト:

fp_format: FLOAT_FORMAT
rdr_id: RENDERER_ID
item_mask: BITMASK

リプライ:

item_list: LISTofVALUE

エラー:

Renderer, FloatingPointFormat, Value

説明:

本リクエストは *rdr_id* で指定されるレンダラ・リソースの属性を通知する. *item_mask* はどの属性が問い合わされ, 通知されるのかを決定する. 指定されたレンダラ属性は *item_list* 中に報告される. *item_list* 内に通知される属性で, 浮動小数型のものは *fp_format* で指定される型をしていなければならない. (上記によって返される値はクライアントにより設定された最後のものである. これらが現在使用中の値かどうかは, レンダラのダイナミックスに依存している. **PEXGetRendererDynamics** を参照.)

6.3.2 レンダラ属性ダイナミックスを得る

名前:

PEXGetRendererDynamics

リクエスト:

rdr_id: RENDERER_ID

リプライ:

tables: BITMASK
namesets: BITMASK
attributes: BITMASK

エラー:

Renderer

説明:

本リクエストは指定されたレンダラのすべての属性に対する dynamics（結合期間）を返す. *tables* に指定されるビットマスクの各ビットには以下の意味付けがなされている.

MarkerBundle	*MarkerBundleContents*
TextBundle	*TextBundleContents*
LineBundle	*LineBundleContents*
InteriorBundle	*InteriorBundleContents*
EdgeBundle	*EdgeBundleContents*
ViewTable	*ViewTableContents*
ColorTable	*ColorTableContents*
DepthCueTable	*DepthCueTableContents*
LightTable	*LightTableContents*
ColorApproxTable	*ColorApproxTableContents*
PatternTable	*PatternTableContents*
TextFontTable	*TextFontTableContents*

namesets に指定されるビットマスクの各ビットには以下の意味付けがなされている.

HighlightNameset	*HighlightNamesetContents*
InvisibilityNameset	*InvisibilityNamesetContents*
PickNameset	*PickNamesetContents*

attributes に指定されるビットマスクの各ビットには以下の意味付けがなされている.

HLHSRMode
NPCSubvolume
Viewport
ClipList
EchoMode

設定された各ビットとも，0 に対応する属性は *dynamic* であり，1 に対応する属性はそうでないことを示している. *dynamic* 属性への変更はただちに効果をもたらす. *dynamic* でない属性はいつでも変更可能である *p が，新しい値は次の明示的／暗黙的な **PEXBeginRendering** が発行されるまで効果が保留される. *HLHSRMode* のような属性を *dynamic* にすることを許すようなインプリメンテーションでは，属性に対して設定可能なすべての値の間での変更の意味を決めておかなければならない.

6.4 クライアント側探訪

以下のリクエストはクライアント側のストラクチャ探訪機能を提供する．クライアント側のピッキングをおこなうリクエストについては **Render Picking** の章を参照されたい．クライアント側でのサーチ探訪機能は提供していない．

6.4.1 レンダリングの開始

名前:

PEXBeginRendering

リクエスト:

rdr_id: RENDERER_ID
drawable_id: DRAWABLE_ID

エラー:

Renderer, Drawable, Match, Alloc, RendererState

説明:

本リクエストにより， *drawable_id* で指定されたドローアブルでのレンダリングが開始される． *rdr_id* で指定されるレンダラの出力は, **PEXEndRendering** リクエストが発行されるまで，上記のドローアブルに対しておこなわれる．初期パイプライン属性は，レンダラ・リソース内の *pipeline_context* により，指定されるパイプライン・コンテキストからレンダラにコピーされる． *pipeline_context* の指定が *Null*のときは，パイプライン・コンテキストにおけるすべての属性に対する省略値が，レンダラにコピーされる．レンダラ・リソースの *renderer_state* 属性は *Rendering* に設定され， *current_path* 属性は要素オフセット 0 とストラクチャ ID が 0 に初期化される． *clear_I*が有効ならば，画像バッファは *background_color* の値に初期化される． *clear_Z* が *c1
有効か， *HLHSR* モードが z- バッファリングの実行を示しているならば， z- バッファは最大値にクリアされる．画像と z- バッファは *clip_list* によって設定された領域内のみクリアされる． *c1

指定されたドローアブルが，レンダラを生成するために本リクエストに渡されたドローアブルと同じ root と depth を持っていなかったときには, あるいは指定されたドローアブルが **PEXMatchRendererTargets** によって返されるドローアブルの 1 つでないときには， *Match* error が発生する． *c1

レンダラの *renderer_stat* 属性が *Rendering*, あるいは *Picking* に設定されたときには，処理は中断され，**PEXBeginRendering** リクエストが実行され， *RendererState* エラーが返される．

6.4.2 レンダリングの終了

名前:

PEXEndRendering

リクエスト:

rdr_id: RENDERER_ID
flush: BOOLEAN

エラー:

Renderer, RendererState *c1

説明: *c1

rdr_id によって指定されたレンダラに対する現在のレンダリング処理を停止させる． *flush* が *True* に設定されていたとき，本リクエストは *rdr_id* で指定されたレンダラにおいて，ペンディング状態にあるすべての出力をドローアブル上に表示する． *flush* が *False* に設定されていたとき，レンダラについてペンディング状態となっていた出力は破棄される．いずれの場合でも *renderer_state* は *Idle* に設定される． *renderer_state* が *Idle* で あったときには（つまり，いっさいのレンダリングがおこなわれないか，またはサイズの変更によりレンダリングが中止されるときには），リクエストは無視され，エラーは発生しない． *renderer_state* が *Picking* ならば， *RendererState* エラーが生じる．

6.4.3 パイプライン属性の退避

名前:

PEXBeginStructure

リクエスト:

rdr_id: RENDERER_ID
s_id: CARD32

エラー:

Renderer

説明:

rdr_id で指定されるレンダラの *renderer_state* 属性が*Rendering* か *Picking* ならば，レンダリング・パイプライン属性を退避 (save)する． *renderer_state* 属性が *Idle* に設定されていれば，このリクエストは無視される．レンダラ・リソースそのものの属性（つまりレンダラの属性で設定 / 変更が可能なもの．たとえば，テーブル ID，ネーム・セット ID etc. ）は，退避されない．レンダラの *current_path* における最終要素へのオフセットはインクリメント（クライアント側の "execute structure" コマンドをカウントするため）される．その後，要素オフセットは 0（空ストラクチャを示す）とともに *s_id* で指定された名称は *current_path* 属性に適用される．後に続く各コマンドの実行ごとに要素オフセットが 1 だけインクリメントされ， **PEXBeginStructure**リクエスト，または **PEXEndStructure**が実行されるまで続く． *c1

レンダリング・パイプライン属性を退避した後， *global_transform* 属性には，現在の *local_transform* 行列と *global_transform* 行列を掛合わせた行列が設定される．その後， *local_transform* 行列は単位行列に設定される．

6.4.4 パイプライン属性の復元

名前:

PEXEndStructure

リクエスト:

rdr_id: RENDERER_ID

エラー:

Renderer， RendererState

説明:

rdr_id のレンダラの *renderer_state* 属性がRendering か *Picking* のとき，本リクエストは直前に退避したレンダラのパイプライン属性を復元 (restore)する．さらにレンダラの *current_path* における最終要素への参照が消去され，後に続く出力コマンドにより，リスト末尾における要素参照の要素オフセットはインクリメントされる．本リクエスト自身は *current_path* に対してインクリメント動作を生じない． *c1

6.5 出力コマンド実行

以下のリクエストは出力コマンドをレンダラにおいて実行する.

6.5.1 出力コマンドによる表示

名前:

PEXRenderOutputCommands

リクエスト:

fp_format: FLOAT_FORMAT
rdr_id: RENDERER_ID
cmds: LISTofOUTPUT_CMD

エラー:

Renderer, FloatingPointFormat, OutputCommand

説明:

rdr_id で指定されるレンダラの *renderer_state* 属性が *Rendering* か *Picking* であったときには, *cmds* 内の出力コマンドが即座に処理される. *renderer_state* 属性が *Idle* であったときには, 出力コマンドは無視される. *cmds* 内の浮動小数値は *fp_format* に指定された形式となる. 出力コマンドの型については, 「出力コマンド」の章で解説されている. 出力コマンドはサーバにより順に処理され, そのうちのひとつがエラーで終了するか, 最後のコマンドまで到達するかのいずれかにより終了する. エラーで終了したコマンド, および以降のコマンドは無視され, *OutputCommand* エラーがクライアントに返される. レンダラの *current_path* 属性は, 実際に処理された出力コマンドのみ反映するように更新される. *c1

6.5.2 要素の表示

*c1

名前:

PEXRenderElements

リクエスト:

rdr_id: RENDERER_ID
s_id: STRUCTURE_ID
range: ELEMENT_RANGE

エラー:

Renderer, Structure, Value

説明:

rdr_id によって指定されるレンダラの *renderer_state* 属性が *Rendering* か *Picking* に設定されていれば, このリクエストは *s_id* で指定されるストラクチャ内のストラクチャ要素のある範囲をレンダリングする. もし *renderer_state* が *Idle* ならば, このリクエストは無視される. 本リクエストの意味は出力コマンドが指定されたストラクチャからおこなわれることを除いて, **PEXRenderOutputCommands** と同じである.

レンダリングされる範囲の要素を決定するオフセットは, 次の方法によって *range* の位置から計算される. *whence* が *Beginning* のときは, 求められるオフセットは *offset* の指定値と同じである. *whence* が *Current* のときは, オフセットは *s_id* 要素のポインタに *offset* を加えて求められる. *whence* が *End* のときは, オフセットはストラクチャ内の要素数に *offset* を加えて求められる. *offset* は負の数でもよい. どのように計算されたオフセットでも負数ならば, レンダリングの実行前に 0 に設定される. どのようなオフセットでもストラクチャの要素数より大きい数ならば, 最後の要素のオフセットに設定される. ストラクチャ要素のポインタはこのリクエストの影響は受けない.

6.5.3 累積状態

*c1

名前:

PEXAccumulateState

リクエスト:

rdr_id: RENDERER_ID
path: LISTofELEMENT_REF

エラー:

Renderer,Path

説明:

rdr_id で指定されるレンダラの *renderer_state* 属性が *Rendering* か *Picking* ならば，このリクエストはあたかも探訪が *path* で指定された要素に対して実行されたのと同じ状態を累積 (accumulate)する．もし *renderer_state* が *Idle* ならば，このリクエストは無視される．

状態の累積は現在のパイプライン属性で開始される．指定されたパスを順番に降りていき，その中にあるストラクチャ要素を順番に試していく．パイプライン状態を変更するような出力コマンド（たとえば **Set Line Color**）を含むいかなる要素でもレンダラの処理に送られる．他のすべての出力コマンド（たとえば **Poly-Line 3D**, **Execute Structure**, **Label** など）はスキップする．パスの中のストラクチャが実行されても，現在のパイプライン属性は “pushed” されないという点で，この探訪は単調 (flat meaning)である．しかし，累積状態の間に出会う各出力コマンドに対して，現在のパス・オフセット (path offset)はそれぞれ処理される．

6.5.4 ネットワークの表示

名前:

PEXRenderNetwork

リクエスト:

rdr_id: RENDERER_ID
drawable_id: DRAWABLE_ID
s_id: STRUCTURE_ID

エラー:

Renderer, Drawable, Structure, Match, Alloc, RendererState *c1

説明:

本リクエストは *s_id* で root を指定されるストラクチャを探訪し，*rdr_id* で指定されるレンダラを用いて *drawable_id* で指定されるドローアブル上でレンダリングをおこなう．

この機能は以下の擬似 (pseudo)リクエストと同義である．

PEXBeginRendering(*rdr_id*, *drawable_id*)
PEXRenderElements(*rdr_id*, all elements of *s_id*) *c1
PEXEndRendering(*rdr_id*, *True*)

7　レンダラ・ピッキング

*c1
レンダリングにおけるピッキングの意味は，ピックされた要素の画素を変更する代わりにテストをおこなうと *c2
いうことを除いて，まったく同じである．言いかえると，レンダリング・パイプラインを通過するすべてのプリミティブ（すなわち不可視フィルタに引っかからず，かつ選択された裏面でもないもの）はピッキングの対象となる．ピッキングは要素の幾何学的な定義を用いるので，中空，空，透明な多角形でもピック可能である．しかし，要素の面積が増加（たとえば線幅）するような表示効果を考慮するかどうかはインプリメンテーション依存である．

２種類のテストが探訪に用意されている：「単一ピック (pick one)」と「全ピック (pick all)」である．単一ピックの探訪はピック基準を満たす唯一つの要素を返す．全ピック探訪はピック基準を満たし，ユーザーによって指定された最大ヒット数までのすべての要素を返す．どちらの探訪についても，要素に対する空間的なテストは *pick_data* で設定されたヒット・ボックスによって制御される．

ヒットボックスは装置座標系（サーバが *DC_HitBox* のピック装置をサポートしている場合），あるいは NPC 座標系（サーバが *NPC_HitVolume* のピック装置をサポートしている場合）で指定される．サポートされているピック装置の型は **PEXGetEnumeratedTypeInfo** によって問い合わせることができる．装置座標系でのヒットボックスは *pick_position* と *pick_distance*によって指定される． *pick_position* は装置座標系でのヒットボックスの中心点座標であり，ヒットボックスの形状（たとえば正方形，長方形，円）はインプリメンテーション依存である． *pick_distance* は装置座標系において，ヒットボックスの半分幅あるいは半径を定義する．NPC ヒットボリュームは NPC 空間の２点で設定される．ピックボックスの内部にあるか，交差している要素はヒットとみなされる．

pick_method はどの「ヒット」プリミティブがピックされるかを決定する．単一ピックについては， *pick_method* は *Last, ClosestZ, VisibleAny, VisibleClosest* にのいずれかに設定できる．全ピックについては， *pick_method* は *All, Visible* のいずれかに設定できる．もし指定された *pick_method* がサポートされていなければ，エラーが返る．サポートされている単一ピックと全ピックの方法は **PEXGetEnumeratedTypeInfo** を使って問い合わせできる．PEX サーバは，単一ピックに対しては *Last* を，全ピックに対しては *All* をサポートしなければならない．

pick_inclusion と *pick_exclusion* はヒット探訪において，要素をフィルタするための名前セットを設定する．ピック・リクエストはピック・フィルタによって除外されない要素を返す．もし可能ならば，単一ピックはピック基準を満たす最も近い，かつピック・フィルタのテストを通過した要素を返す．単一ピックのリクエストからは，ピック・フィルタを満たさなかった要素が，指定された *pick_method* にとってより良い候補かどうかを示すフラグを返す． *pick_inclusion* と *pick_exclusion* のダイナミックスは **PEXGetRendererDynamics** によって決めることができる．

単一ピック探訪ではただ一つの要素がピックされる．もし *pick_method* が *Last* ならば，ピック・フィルタを満たす最後のヒット要素が返る． *pick_method* が *ClosestZ* ならば，前方クリッピング面に最も近い z 値を持ち，かつピック・フィルタを満たすヒット要素を返す．もし複数の要素がこの基準を満たすならば，それらのどれかが返る． *pick_method* が *VisibleAny* ならば，（現在の HLHSR モードを考慮して）可視であり，かつピック・フィルタを満たすヒット要素を返す． *pick_method* が *VisibleClosest* ならば，可視のすべてのヒット要素が最初に選ばれる．選択された要素の中で， *pick_position* に最も近く，かつピック・フィルタを満たすもの一つを返す．NPC ヒット・ボリュームが設定されているならば， *pick_position* はボリュームの中心である．「可視」のピック法については，現在の HLHSR モードで表示されていて，ピック口径の内部で可視であるような，一つの要素を返す．たとえば，HLHSR モードが *ZBuffer* に設定されているならば，単一ピック法 *ClosestZ* は，ピック・フィルタを満たさない他のプリミティブによって隠蔽された一つの要素を返す．しかし，単一ピック法 *VisibleAny* はピックされた要素が隠蔽されていないことを保証する．

全ピック探訪では複数の要素がピックされる． *pick_method* が *All* ならば，ヒット・ボックスに含まれるか交差する，最大 *max_hits* 個の要素を返す． *pick_method* が *Visible* ならば，（現在の HLHSR モードを考慮して）可視で，かつヒット・ボックスに含まれるか交差する，最大 *max_hits* 個の要素を返す． *max_hits* を１に設定すると，最初の一つのピックが実行できる． *PickAll* はさらにレンダラ属性 *pick_start_path* によって制御される． *pick_start_path* は次の *PickAll* 探訪をどこから開始するかを示す．

レンダラに関連したあるウィンドウが，レンダラが単一ピックか全ピックを実行中に壊されたり，リサイズされたりしたならば，ピック操作は停止し，レンダラは *Idle* 状態に戻る．これに続くすべての出力コマンドと探訪コマンド（すなわち **PEXRenderOutputCommands**, **PEXBeginStructure**, **PEXEndStructure**, **PEXRenderElements**, **PEXAccumulateState**）は，**PEXEndPickOne** か **PEXEndPickAll** リクエストに至るまで無視される．エラーは生じないが，ピックのリプライとして *pick_status* が *Aborted* にセットされる．

レンダラに関連したあるウィンドウが，レンダラが単一ピックか全ピックを実行中に移動 (move) したり，出現 (expose) したり，隠れたり (occlude) したならば，サーバは新しいウィンドウ階層の上で出力コマンドを処理し続

けなければならない．一方，ピック操作は明示的あるいは暗黙的に停止させられる．

7.1 単一ピック

7.1.1 単一ピックの開始

名前:

PEXBeginPickOne

リクエスト:

fp_format: FLOAT_FORMAT
pick_method: PICK_ONE_METHOD
rdr_id: RENDERER_ID
drawable_id: DRAWABLE_ID
s_id: INT32 *c2
pick_data: PICK_RECORD

エラー:

Renderer, Drawable, Match, Alloc, RendererState, Value

説明: *c2

このリクエストは，クライアント側での単一ピック探訪を開始する．レンダラが単一ピック・モードにあるとき，要素は画素を変更される代わりにヒット・テストを受け，ピックされた要素への階層パスが保持される．*pick_method* はただ一つの要素が選択されるように，ピック基準を設定する．これは *Last, ClosestZ, VisibleAny, VisibleClosest* のいずれかにセットされる．（**PEXBeginPickAll** は *max_hits* が 1 に設定されたとき，最初にピックされた要素を返す．） もし指定された *pick_method* がサポートされていなければ， *Value* エラーが返る．

ピック口径は *pick_data* によって定義される． *DC_HitBox*か *NPC_HitVolume* を設定できる．

s_id はクライアント・ストラクチャのルートである．ピック探訪が開始される前に，現在のパスはストラクチャ *s_id* に，要素オフセットは 0 に初期化される．単一ピック探訪から返ってくるピック・パスはピックされた要素が処理された時点のレンダラ・パスに等しい．

もし指定されたドローアブルが，レンダラを生成するために通過したドローアブルと同じルートと深さを持たない場合，または指定されたドローアブルが **PEXMatchRendererTargets** によって返されるドローアブルの一つでない場合は， *Match* エラーが生じる．もし *renderer_state* が *Rendering* か *Picking* に設定されていると，その処理は停止し， **PEXBeginPickOne** リクエストが実行され， *RendererState*エラーが生じる．

出力コマンドの処理や現在の属性値の操作（すなわち **PEXRenderOutputCommands**， **PEXBeginStructure**， **PEXEndStructure**， **PEXRenderElements**， **PEXAccumulateState**）をおこなうリクエストは，レンダラが *Picking* モードのときに発行される．それらはプリミティブが画素を変更される代わりにテストされることを除いて，同じ意味を持っている．

7.1.2 単一ピックの終了 *c1

名前:

PEXEndPickOne

リクエスト:

rdr_id: RENDERER_ID

リプライ:

pick_status: PICK_STATUS *c2
better_pick: BOOLEAN *c2
pick_path: LISTofPICK_ELEMENT_REF

エラー:

Renderer, RendererState

説明: *c2

このリクエストは，レンダラを *Idle* モードにし，階層的なピック・パスに沿ったピック状態を返す．要素がピックされると， *pick_status* は *Ok* にセットされ， *pick_path* はピック要素が処理された時点のレンダラのパスを意味する．どのプリミティブもピックされないと， *pick_status* は *NoPick* にセットされ， *pick_path* は空となる．もしピック操作中にレンダラのドローアブルが破壊されたり，リサイズされたりしたら， *pick_status* は *Aborted* に， *pick_path* は空にセットされる．

もしピック基準を満たす最も近傍の要素で，かつピック・フィルタをパスしなかったものがあれば， *better_pick* が *True* にセットされる．他の場合は， *bettern_pick* は *False* である．

もし *renderer_state* 属性が現在 *Idle* ならば（すなわちピッキングがおこなわれていないか，リサイズのためにレンダリングが中断した），リクエストは無視され，何のエラーも生じない．もし *rendere_state* が現在 *Rendering* か，”pick one” が起きているならば， *RendererState* エラーが生じる．

7.1.3 1つピックする

名前:

PEXPickOne

リクエスト:

fp_format: FLOAT_FORMAT
pick_method: PICK_ONE_METHOD *c2
rdr_id: RENDERER_ID
drawable_id: DRAWABLE_ID
s_id: STRUCTURE_ID
pick_data: PICK_RECORD

リプライ:

pick_status: PICK_STATUS *c2
better_pick: BOOLEAN *c2
pick_path: LISTofPICK_ELEMENT_REF

エラー:

Renderer, Drawable, Structure, Match, Alloc, RendererState, Value

説明:

このリクエストは，ルート *s_id* のストラクチャについて，単一ピック探訪を実行する．これは階層的ピック・パスに沿ったピック状態を返す.

このリクエストは以下の（擬似）リクエストと同義である.

PEXBeginPickOne(*pick_method*, *rdr_id*, *drawable_id*, *s_id*, *pick_data*) *c2
PEXRenderElements(*rdr_id*, all elements of *s_id*)
PEXEndPickOne(*rdr_id*) *c2

7.2 全ピック *c1

7.2.1 全ピックの開始

名前:

PEXBeginPickAll

リクエスト:

fp_format: FLOAT_FORMAT
pick_method: PICK_ALL_METHOD *c2
send_event: BOOLEAN
rdr_id: RENDERER_ID
drawable_id: DRAWABLE_ID
s_id: INT32 *c2
pick_max_hits: CARD32
pick_data: PICK_RECORD

エラー:

Renderer, Drawable, Match, Alloc, RendererState, Value

説明:

このリクエストは，クライアント側の全ピック探訪を開始する．レンダラが「全ピック」モードであれば，要素は画素を変更する代わりにヒット・テストを受ける．ピックされたすべての要素は，*pick_max_hits* に至るまで記録される．続く要素は記録されない．いったん最大ヒット数に至ると，サーバは次に続くすべての要素は無視してもかまわない．*max_hits* を 1 にセットすると，このリクエストは最初にピックされた要素を返す．

pick_method は全ピックに関して，ピック基準を定義する．それは *All* か *Visible* にセットされる．もし指定された *pick_method* がサポートされていなければ，*Value* エラーが生じる．

ピック口径は *pick_data* によって定義される．*DC_HitBox* か *NPC_HitVolume* のどちらかを指定できる．

s_id はクライアント・ストラクチャのルートである．ピック探訪が開始される前に，現在のパスはストラクチャ *s_id* に，要素オフセットは 0 に初期化される．全ピック探訪から返ってくるすべてのピック・パスは，ピックされた要素が処理された時点のレンダラ・パスに等しい．

send_event が *true* ならば，MAX_HITS_REACHED 事象がクライアントに送られる．これはサーバにおいて最大ヒット数に達したこと示す．この事象を受けると，クライアントは探訪を終了し，記録されたヒットが十分か確認する必要がある．

もし指定されたドローアブルがレンダラを生成するために通過したドローアブルと同じルートと深さを持たない場合，または指定されたドローアブルが **PEXMatchRendererTargets** によって返されるドローアブルの一つでない場合は，*Match* エラーが生じる．もし *renderer_state* が *Rendering* か *Picking* に設定されていると，その処理は停止し，**PEXBeginPickAll** リクエストが実行され，*RendererState*エラーが生じる．

7.2.2 全ピックの終了 *c1

名前:

PEXEndPickAll

リクエスト:

rdr_id: RENDERER_ID

リプライ:

pick_status: PICK_STATUS *c2
more_picks: PICK_ALL_STAFF
pick_paths: LISTofLISTofPICK_ELEMENT_REF

エラー:

Renderer, RendererState

説明: *c2

このリクエストは，現在のレンダラ側の全ピック探訪を停止し，レンダラを *Idle* モードにし，階層的なピック・パスに沿ったピック状態を返す．一つ以上の要素がピックされると， *pick_status* は *Ok* にセットされ，*pick_path* はピック要素が処理された時点のレンダラの各パスを探訪した順に指す．この階層ピック・パスはピック処理がおこなわれた時点のレンダラのパスに等しい．どの要素もピックされないと， *pick_status* は *NoPick* にセットされ， *pick_path* は空となる．もしピック操作中にレンダラのドローアブルが破壊されたり，リサイズされたりしたら， *pick_status* は *Aborted* に， *pick_path* は空にセットされる．

もしすべてのピックが記録されたならば， *more_picks* は *NoMoreHits* にセットされ，レンダラの *pick_start_path* は空となる．もし最大ヒット数に達し，続くピックが消去されたならば， *more_hits* は *MoreHits* にセットされ， *pick_start_path* は記録された最後のピック要素にセットされる．もしサーバが，最大ヒット数に達した後，続く出力コマンドを無視し始めたならば， *more_hits* は *MayBeMoreHits* にセットされ， *pick_start_path* は，要素が無視される前にレンダラによって処理された最後の要素にセットされる．

もし *renderer_state* 属性が現在 *Idle* ならば（すなわちピッキングがおこなわれていないか，リサイズのためにレンダリングが中断した），リクエストは無視され，何のエラーも生じない．もし *renderer_state* が現在 *Rendering* か，”pick one” が起きているならば， *RendererState* エラーが生じる．

7.2.3 全ピックする

*c1

名前:

PEXPickAll

リクエスト:

fp_format: FLOAT_FORMAT
pick_method: PICK_ALL_METHOD *c2
rdr_id: RENDERER_ID
drawable_id: DRAWABLE_ID
pick_max_hits: CARD32
pick_data: PICK_RECORD

リプライ:

pick_status: PICK_STATUS *c2
more_picks: BOOLEAN *c2
pick_paths: LISTofLISTofPICK_ELEMENT_REF

エラー:

Renderer, Drawable, Match, Alloc, RendererState, Value, Path

説明:

このリクエストは， *pick_start_path* 属性の設定からストラクチャ・ネットワークの探訪を開始する．ヒット・テストは *pick_start_path* が指す最後の要素の次から開始される．もし開始パスが正しいパスを指していないときは， *Path* エラーが生じ，何の結果も返らない．

ピック口径は *pick_data* によって定義される． *DC_HitBox* か *NPC_HitVolume* のどちらかを指定できる．

すべてのヒット要素のパスは *pick_max_hits* に至るまで記録される．これらのパスは *pick_paths* の中に探訪の順に返る．いったん最大パス数に至ると，サーバは続く要素を無視して，見つかった結果を返してもかまわない． *max_hits* が 1 にセットされていると，最初にピックされた要素を返す．

すべてのヒットが記録されたならば， *more_picks* は *NoMoreHits* にセットされ， *pick_start_path* は空になる．最大ヒット数に達して，続くヒットが消去されたならば， *more_hits* は *MoreHits* にセットされ， *pick_start_path* は要素を無視し始める前にレンダラによって処理された最後の要素にセットされる．

この機能は以下の（擬似）リクエストと同義である．

PEXBeginPickAll(*pick_method*, *send_event*, *rdr_id*, *drawable_id*, *s_id*, *pick_max_hits*, *pick_data*)

PEXRenderElements(*rdr_id*, all elements of *s_id*)
PEXEndPickAll(*rdr_id*) *c2

8 ストラクチャ

ストラクチャ・リソース (structure resource)の使用法と管理法を説明する．ストラクチャ・リソースは出力コマンドを遅延実行 (later excution)するために一時的に蓄えるものである．ストラクチャ・リソースには外部から変更可能な 2 つの属性と問い合わせのみ可能な複数の属性がある．変更可能な属性は以下の 2 つである．

element_ptr

ストラクチャ要素のストラクチャ内でのオフセットである．要素は順序付けされていて，1 から始まる．要素ポインタは変更または問い合わせをおこなう位置を指定する．ストラクチャへの更新要求において，ストラクチャがデスティネーションとして使用されるとき，ストラクチャ要素ポインタは処理の一環として更新される．ストラクチャが新規に作成されたとき，要素ポインタは 0 に設定される．

editing_mode

編集モード属性はストラクチャに対し，どのような更新処理が影響するかを設定する．モードが *StructureInsert*であるときには，続くストラクチャ要素の作成要求により，ストラクチャに要素が挿入される．要素はポインタが指す要素の 直後 に挿入される．挿入後，要素ポインタは挿入された要素数だけインクリメントされる．モードが *StructureReplace* であるときには，要素作成の出力コマンドが要素ポインタが指す位置から始まる要素の書換えをおこなう．ストラクチャ作成時には，モードは *StructureInsert*となる．

ストラクチャの変更不可能な属性としては，ストラクチャ・サイズ (要素数および 4 バイト単位のサイズ) と他のストラクチャから参照される回数がある．ストラクチャに対する付加情報 (親階層 (ancestor)，子階層 (descendant) etc.) は利用可能で，個別に問い合わせが可能である．

8.1 ストラクチャ・リソース管理

ストラクチャは X11 のリソースであり，X11 のリソースに対する管理責任とアクセス権を持つ．ここで説明するリクエストはストラクチャに対する生成，削除，その他一般的な操作を行なう．

8.1.1 ストラクチャの作成

名前:

PEXCreateStructure

リクエスト:

s_id: STRUCTURE_ID

エラー:

IDChoice, Alloc

説明:

s_id により指定される ID を持つストラクチャを生成する．

8.1.2 ストラクチャの複写

名前:

PEXCopyStructure

リクエスト:

src_s_id: STRUCTURE_ID
dest_s_id: STRUCTURE_ID

エラー:

Structure

説明:

src_s_id により指定されるストラクチャ内の要素を *dest_s_id* により指定されるストラクチャへ複写する．*dest_s_id* により指定されたストラクチャ内の要素は，写前にすべて削除される．*dest_s_id* により指定されたストラクチャはあらかじめ用意されていなければならない．*src_s_id* の要素ポインタと編集モードも *dest_s_id* のストラクチャにコピーされる．

8.1.3 ストラクチャの削除

名前:

PEXDestroyStructures

リクエスト:

list: LISTofSTRUCTURE_ID

エラー:

Structure

説明:

list により指定されるストラクチャ・リソースを削除し，削除したストラクチャ・リソースを参照する他のストラクチャ・リソース内の要素も削除する． *list* により指定されるストラクチャ・リソースを参照するストラクチャ要素はすべて消去され， PHIGS ワークステーション・リソースに掲示されている *list* 内のすべてのストラクチャは削除される． *list* 内のストラクチャ・リソースの ID を含むサーチ・コンテキストのパスやピックメジャは問い合わせの対象となり得る．このようなパスは，ストラクチャが削除された後もストラクチャ・リソース ID を保持している しかしながら，上記のパスを使用するような **PEXSearchNetwork** または **PEXUpdatePickMeasure** を実行すると *Path* エラーが発生する．

PEXDestroyStructures により要素が削除されたストラクチャ・リソースは，以下の規則にしたがって要素ポインタを移動させる．要素ポインタが削除された要素より前にあったとき，要素ポインタは変更されない．要素ポインタが削除された要素を指していたときには，ひとつ前の要素を指すように変更される．要素ポインタが削除された要素より後を指していたときには， **PEXDestroyStructures** が発行される以前に指していた要素を指すように変更される．

8.2 ストラクチャの問い合わせ

以下のリクエストは属性やストラクチャに関する情報を問い合わせる.

8.2.1 ストラクチャ情報を得る

名前:

PEXGetStructureInfo

リクエスト:

fp_format: FLOAT_FORMAT
s_id: STRUCTURE_ID
item_mask: BITMASK_SHORT *c2

リプライ:

editing_mode: EDIT_MODE
element_ptr: CARD32
num_elements: CARD32
total_length: CARD32
has_refs: BOOLEAN

エラー:

Structure, FloatingPointFormat, Value

説明:

本リクエストは *s_id* により指定されるストラクチャに関する情報を返す. *item_mask* は復帰値のどのフィールドがクライアントにより要求されたかを示す. *item_mask* により指定された Item には復帰値として有効な値が返されるが, 残りの Item には不確定な値が返される. ストラクチャの編集モード属性値は *editing_mode* に返される. ストラクチャ要素ポインタは *element_ptr* に返される. 指定されたストラクチャ内にあるストラクチャ要素数は *num_elements* に返される. ストラクチャが "空" であったときには, *num_elements* には 0 が返される. (クライアント・サイドでの) ストラクチャ・サイズが 4 バイト単位として *total_length* に格納される. (この情報は, **PEXFetchElements** を実行する前にクライアントが必要とするメモリ・サイズを提供する.) *s_id* を参照する "execute structure" が他のストラクチャ・リソースに存在したときには, *has_refs* が *True* で返される. 存在しなかったときには, *False* で返される. 返された属性値は, 復帰値が取る *fp_format* で指定された型で計算される.

8.2.2 ストラクチャ要素情報を得る

名前:

PEXGetElementInfo

リクエスト:

fp_format: FLOAT_FORMAT
s_id: STRUCTURE_ID
range: ELEMENT_RANGE

リプライ:

info: LISTofELEMENT_INFO

エラー:

Structure, Value, FloatingPointFormat

説明:

本リクエストは *s_id* で指定されるストラクチャにおける要素に関する情報を通知する．得られるべき情報に関与する要素の位置を決定するオフセットは， *range* に与えられる値を用いて以下の規則により求められる．*whence* が *Beginning* であったときには，オフセットは *off_set* で与えられる数値そのものとなる． *whence* が *Current* であったときには， *s_id* で指定されるストラクチャの要素ポインタに *offset* を加えた値となる．*whence* が *End*であったときには，ストラクチャ内の要素数に *offset* を加えた値となる． *offset* は負の値を取ることも許されている．

計算されたオフセットが負であったときには，要素情報を得る前に 0 に置き換えられる．オフセットがストラクチャ内の要素数より大きかったときには，ストラクチャの最終要素を指すように変更される．復帰値としては，各要素の ELEMENT_TYPE フィールド，および要素長を 4 バイト単位で通知する．ストラクチャ内のすべての要素長の合計は **PEXGetStructureInfo** リクエストにより得られるデータ長と等しい．返された属性値は，復帰値が取る *fp_format* で指定された型で計算される．オフセットが 0 のとき，復帰値はない． *s_id* の要素ポインタは本リクエストにより影響を受けることはない．

8.2.3 ネットワーク内のストラクチャの通知

名前:

PEXGetStructuresInNetwork

リクエスト:

s_id: STRUCTURE_ID
which: {*All*, *NoCrossRefs*}

リプライ:

structures: LISTofSTRUCTURE_ID

エラー:

Structure, Value

説明:

本リクエストは， *s_id* で指定されるストラクチャをルートとするストラクチャ・ネットワークにおいて参照されるストラクチャ・リソースの ID リストを返す． *which* が *All* であったときには， *s_id* により指定されるルートを持つストラクチャ・ネットワーク内に存在するすべてのストラクチャ・リソースの ID リストを *structure* に返す． *which* が *NoCrossRefs* であったときには，指定されたストラクチャ・ネットワーク外から参照されないストラクチャの ID のみが返される． *s_id* は，それが無効な ID （返されたリストが空の場合）でないかぎり，常にリスト内に返される．

8.2.4 ネットワーク内の先祖ストラクチャの検索

名前:

PEXGetAncestors

リクエスト:

s_id: STRUCTURE_ID
path_part: {*TopPart*, *BottomPart*}
path_depth: CARD32

リプライ:

paths: LISTofLISTofELEMENT_REF

エラー:

Structure, Value

説明:

本リクエストは *s_id* に続く（先立つ）ストラクチャ・ネットワーク・パスを返す．パス・リストはストラクチャ ID と要素オフセットのペアとして与えられ，ルート・ストラクチャから *s_id* に向かう．完全なパスを与えるため，パス・リストの最後のエントリは *s_id* と等しい ID と 0 のオフセットを保持しなければならない．*path_depth* はパスの最大長（ストラクチャ ID と要素オフセットのペアの数）を与える．ただし， *path_depth* = 0 が指定されたときには，フル・パスが返される． *path_part* は，各パスの先頭か最尾のいずれが返されるかを指定する． *path_part* = *TopPart* が指定されたときには，一番最初のストラクチャ ID と要素オフセットのペアが返され， *path_part* = *BottomPart*が指定されたときには， *path_depth* で指定される範囲内で一番最後のストラクチャ ID と要素オフセットのペアが返される．返されるパスはユニークである．（つまり，返されるパス・リストの中に重複はない．）

たとえば， *path_depth* = 0, *path_part* = *TopPart* と設定すると *s_id* までのパスはすべて返される． *path_depth* = 1， *path_part* = *TopPart* と設定すると *s_id* を含むストラクチャ・ネットワークについて，おのおのの先頭のストラクチャに関してパスが返される． *path_depth* = 1, *path_part* = *BottomPart* と設定すると *s_id* が参照されているか否かを判定することができる． *path_depth* = 2， *path_part* = *BottomPart* と設定 すると *s_id* に対する直接の先祖の数を知ることが可能である．

8.2.5 ネットワーク内の子孫ストラクチャの検索

名前:

PEXGetDescendants

リクエスト:

s_id: STRUCTURE_ID
path_part: {*TopPart*, *BottomPart*}
path_depth: CARD32

リプライ:

paths: LISTofLISTofELEMENT_REF

エラー:

Structure, Value

説明:

本リクエストは，ストラクチャ・ネットワークにおいて *s_id* から階層構造の末節に向かうパス・リストを返す．パス・リストはストラクチャ ID と要素オフセットのペアとして与えられ，*s_id* から階層構造の末節 (leaf node)に向かう．完全なパスを与えるため，パス・リストの最初のエントリは，*s_id* と等しい ID と階層構造において子供を参照する要素を指すエントリへのオフセットを含まなければならない．さらに，最後のエントリは階層構造における末節のストラクチャの ID と 0 のオフセットを持たなければならない．*path_depth* はパスの最大長（ストラクチャ ID と要素オフセットのペアの数）を与える．ただし，*path_depth* = 0 が指定されたときには，フル・パスが返される．*path_part* は，各パスの先頭 (head)か末尾 (tail)のいずれが返されるかを指定する．*path_part* = *TopPart* が指定されたときには，一番最初のストラクチャ ID と要素オフセットのペアが返され，*path_part* = *BottomPart* が指定されたときには，*path_depth* で指定される範囲内で一番最後のストラクチャ ID と要素オフセットのペアが返される．返されるパスはユニークである．（つまり，返されるパス・リストの中に重複はない．）

たとえば，*path_depth* = 0，*path_part* = *TopPart* と設定すると *s_id* から末節までのすべてのパスが返される．*path_depth* = 1，*path_part* = *TopPart* と設定すると *s_id* において直接参照されるすべてのストラクチャへのパスが返される．*path_depth* = 1，*path_part* = *BottomPart* と設定すると *s_id* をルートとするストラクチャ・ネットワークの末節のストラクチャを知ることができる．

8.2.6 ストラクチャ要素を得る (Fetch)

名前:

PEXFetchElements

リクエスト:

fp_format: FLOAT_FORMAT
s_id: STRUCTURE_ID
range: ELEMENT_RANGE

リプライ:

elements: LISTofOUTPUT_CMD

エラー:

Structure, FloatingPointFormat, Value

説明:

本リクエストは *s_id* で指定されたストラクチャにおけるストラクチャ要素の位置を返す．フェッチ (fetch)される要素の位置を決めるオフセットは，*range* に設定される位置を用いて以下のように求められる．*whence* が *Beginning* であったときには，オフセットは *offset* で与えられる数値そのものとなる．*whence* が *Current* であったときには，*s_id* で指定されるストラクチャの要素ポインタに *offset* を加えた値となる．*whence* が *End* であったときには，ストラクチャ内の要素数に *offset* を加えた値となる．*offset* は負の値を取ることも許されている．

計算されたオフセットが負であったときには，フェッチがおこなわれる前に 0 に置き換えられる．オフセットがストラクチャ内の要素数より大きかったときには，ストラクチャの最終要素を指すように変更される．*s_id* の要素ポインタはフェッチ動作により影響を受けることはない．各ストラクチャ要素のフォーマットは，出力コマンドにおいて使用されるフォーマットと同じである．

elements に返される浮動小数値は，*fp_format* において指定されるフォーマットによって返される（これは PEX のインプリメンテーションにおいてサポートされていることが前提である）．

オフセットを 0 として本リクエストを発行しても情報は得られない．

8.3 ストラクチャ・リソース属性の変更

ここではストラクチャ・リソース属性の変更 (modification)に使用するリクエストを説明する.

8.3.1 編集モードの設定

名前:

PEXSetEditingMode

リクエスト:

s_id: STRUCTURE_ID
mode: EDIT_MODE

エラー:

Structure, Value

説明:

本リクエストは *s_id* のストラクチャに対する編集 (editing)モードを変更する. 編集モード属性はストラクチャに対して編集処理がいかなる影響を及ぼすかを規定する. 編集モードが *StructureInsert* に設定されると, 以下に続くストラクチャ要素の生成リクエストにより, ストラクチャ内に新規な要素が挿入される. 新規な要素は要素ポインタの指す要素の 直後 に挿入される. 挿入動作にともない, 要素ポインタは挿入された要素の数だけインクリメントされる. 編集モードが *StructureReplace* に設定されると, 要素ポインタの指す要素の位置から新規な要素に置き換える.

8.3.2 要素ポインタの設定

名前:

PEXSetElementPointer

リクエスト:

s_id: STRUCTURE_ID
position: ELEMENT_POS

エラー:

Structure, Value

説明:

本リクエストは *s_id* で指定されるストラクチャに対する要素ポインタを設定する. ストラクチャにおける要素ポインタは *position* で指定される位置に設定される. *position* は *whence* と *offset* のペアを含んでいる. *whence* が *Beginning* であったときには, 要素ポインタが *offset* で指定される位置に設定される. *whence* が *Current*であったときには, 要素ポインタは現在の要素ポインタの位置に *offset* を加えることにより求められる. *whence* が *End* であったときには, 要素ポインタはストラクチャの最終要素のオフセット値に *offset* を加えた結果となる. *offset* は負の値を取ることも許されている.

計算の結果, 要素ポインタが負となったときには, 要素ポインタは 0 に設定される. 要素ポインタがストラクチャ内の要素数より大きくなったときには, 要素ポインタは最終要素を指すように変更される.

8.3.3 要素ポインタのラベルへの移動

名前:

PEXSetElementPointerAtLabel

リクエスト:

s_id: STRUCTURE_ID
label: INT32
offset: INT32

エラー:

Structure, Label

説明:

本リクエストは *s_id* で指定されるストラクチャに対する要素ポインタを設定する．設定は "label" が付いたストラクチャ要素を基準におこなわれる．ラベルの検索は（現在の要素ポインタ + 1）の位置から始められ，オフセット増加方向におこなわれる． *label* を含んだラベル・ストラクチャが検出されると，要素ポインタはラベル・ストラクチャ位置 + *offset* へ設定される．

計算の結果，要素ポインタが負となったときには，要素ポインタは 0 に設定される．要素ポインタがストラクチャ内の要素数より大きくなったときには，要素ポインタは最終要素を指すように変更される．検索はオフセット増加方向におこなわれる．（したがって，"execute structure" により参照されるストラクチャは検索の対象とはならない．） 指定されたラベル・ストラクチャが検出されないときには，エラーが返され，要素ポインタは変更されない．

8.3.4 要素の検索

名前:

PEXElementSearch

リクエスト:

s_id: STRUCTURE_ID
position: ELEMENT_POS
direction: {*Forward*, *Backward*}
incl: LISTofELEMENT_TYPE
excl: LISTofELEMENT_TYPE

リプライ:

status: {*Found*, *NotFound*}
found_offset: CARD32

エラー:

Structure, Value

説明:

本リクエストは *s_id* で指定されるストラクチャにおいて，要求された型の要素の検索をおこなう．検索が開始される位置オフセットは，*position* に含まれる *whence* と *offset* により求められる．*whence* が *Beginning* であったときには，検索は *offset* で指定される要素からおこなわれる．*whence* が *Current* であったときには，検出開始位置を求めるために，*offset* が *s_id* で指定されるストラクチャの要素ポインタに加えられる．*whence* が *End* であったときには，検出開始位置を求めるために，*offset* が *s_id* で指定されるストラクチャの最終要素のオフセットに加えられる．*offset* は負の値を取ることも許されている．

計算されたオフセットが負となったときには，検索はストラクチャの先頭要素から始められる．計算されたオフセットがストラクチャ内の要素数より大きくなったとき，検索はストラクチャの最終要素から始められる．検索は常に検索開始要素を検索の対象とする．

検索対象の要素型が *excl* に含まれておらず，*incl* に含まれていたときに，その要素は選択される．検索要素型に *all* を指定したときには，すべての要素型が該当となる．同一の要素型が *excl* と *incl* の両方に含まれていたときには，その型は検索対象から除外される．

該当するものが検出されるか，ストラクチャの末尾に到達したときには，検索は終了する．検索は検索開始位置から始められ，*direction* の指定により，ストラクチャ内を上方に検索するか，もしくは下方に検索するかが選択できる．検索はどちらか一方向に向かっておこなわれる．（したがって，”execute structure” により参照されるストラクチャは検索の対象とはならない．） 該当するものが検出されて，検索が成功したときには，*Found* が返され，該当要素のストラクチャ内オフセットが *found_offset* に通知される．検索が不成功に終わったときには，*NotFound* が返され，*found_offset* には 0 が格納される．

本リクエスト結果により *s_id* で指定されるストラクチャの要素ポインタは影響を受けない．

8.4 ストラクチャの編集

ここではストラクチャ・リソースの内容を変更するためのリクエストについて説明する.

8.4.1 要素の退避

名前:

PEXStoreElements

リクエスト:

fp_format: FLOAT_FORMAT
s_id: STRUCTURE_ID
elements: LISTofOUTPUT_CMD

エラー:

Structure, FloatingPointFormat, OutputCommand

説明:

本リクエストは *s_id* で指定されるストラクチャに要素を退避する. ストラクチャに対する *editing_mode*属性が *StructureInsert* となっていたときには, *elements* に登録された各要素は要素ポインタで指示された要素の後ろ に追加される. リクエスト中の出力コマンドは要素のひとつがエラーで終わるか, またはリストの最後に到達するまで処理される. エラー終了したコマンド, および後に続くコマンドは無視される (ストラクチャにも追加されない). さらに, *OutputCommand* エラーがクライアントに通知される. 要素ポインタは要素追加にともなって加算される. リクエストの終了後, 要素ポインタは挿入された要素の最終要素を指すように設定される.

editing_mode 属性が *StructureReplace* となっていたときには, 置き換えられる要素並びが **PEXDeleteElement** を *position1* whence/offset= {*Current*, 0}, *position2* whence/offset= {*Current*, *n-1*} (ここで *n* は削除する要素の数) で発行したときのように削除される. (要素 0 が削除される領域に含まれていたときには, 要素 0 を除くすべての要素が削除される.) 要素ポインタの指す要素と最終要素の間に, *elements* に指定される出力コマンド数分の大きさがなかったときには, 要素ポインタの指す要素と最終要素の間にある分の要素が削除される. このとき要素ポインタは削除される領域の直前を指す. 削除後, *elements* に登録された出力コマンドが要素ポインタの直後に挿入される.

elements 内の浮動小数点データの型は, *fp_format* に指定される型となる. 出力コマンドに使用される浮動小数点データの型は, 「出力コマンド」の章に説明されている.

8.4.2 ストラクチャ要素の削除

名前:

PEXDeleteElements

リクエスト:

s_id: STRUCTURE_ID
range: ELEMENT_RANGE

エラー:

Structure, Value

説明:

本リクエストは *s_id* で指定されるストラクチャから *range* で示される領域の要素を削除する．削除される要素の範囲は *position1* と *position2* の *whence* と *offset* を用いて計算される．

削除される要素範囲へのオフセットは， *range* に指定される位置を用いて，以下のように求められる． *whence* が *Beginning* であったときに，オフセットは *offset* で与えられる数値そのものとなる． *whence* が *Current* であったときには， *s_id* で指定されるストラクチャの要素ポインタに *offset* を加えた値となる． *whence* が *End* であったときに，ストラクチャ内の要素数に *offset* を加えた値となる． *offset* は負の値を取ることも許されている．

計算されたオフセットが負であったときには，削除がおこなわれる前に 0 に置き換えられる．オフセットがストラクチャ内の要素数より大きかったときには，ストラクチャの最終要素を指すように変更される． 1 つ目のオフセットが 2 つ目のオフセットより小さくなければならない必要はない．削除は境界を含めておなわれる．これは削除範囲の上／下限境界は共に削除の対象となることを意味している． 2 つのオフセットのいずれが削除範囲の上限と下限を規定していても問題とはならない．オフセット 0 の要素の削除は実効的には ”no-op” である．削除動作の後，ストラクチャの要素ポインタは，削除された範囲に続く要素を指すように変更される．

8.4.3 ラベルまでのストラクチャ要素の削除

名前:

PEXDeleteElementsToLabel

リクエスト:

s_id: STRUCTURE_ID
position: ELEMENT_POS
label: INT32

エラー:

Structure, Label, Value

説明:

本リクエストは *s_id* で指定されるストラクチャにおいて，計算されたオフセットと指定されたラベル間にあるすべての要素を削除する．オフセットは *position* を用いて以下の規則にしたがい，求められる． *whence* が *Beginning* であったときには，オフセットは *offset* に指定される数値そのものとなる． *whence* が *Current* であったときには，オフセットは *s_id* の要素ポインタに *offset* を加えた値となる． *whence* が *End* であったときには，オフセットはストラクチャの要素数に *offset* を加えた値となる． *offset* は負の値を取ることも許されている．

求められたオフセットが負であったときには，削除動作が起こる以前に 0 に変更される．求められたオフセットが，ストラクチャの要素数を越える値であったときには，ストラクチャの最終要素に設定される．削除は計算されたオフセットが指す要素から，最初の ”label” で示されるラベルの位置までおこなわれる．ラベルそのものは削除の対象とはならない．ストラクチャ要素ポインタは削除がおこなわれた後，削除された領域の直前に設定される．指定されたラベルが存在しなかったときには削除動作はおこなわれず，要素ポインタは変化しない．

8.4.4　2ラベル間のストラクチャ要素の削除

名前:

PEXDeleteElementsBetweenLabels

リクエスト:

s_id: STRUCTURE_ID
label1: INT32
label2: INT32

エラー:

Structure,　Label

説明:

本リクエストは *s_id* で指定されるストラクチャにおいて，*label1* 指定されるラベルと *label2* で指定されるラベルの間にあるすべての要素を削除する．最初に *label1* で指定されるラベル位置の検出が要素ポインタの指す位置からおこなわれる．その後，*label2* で指定されるラベル位置の検出が *label1* で指定されるラベル位置からおこなわれる．2つのラベルの間にある要素の削除がおこなわれる．2つのラベルそのものは削除されない．ストラクチャ要素ポインタは削除がおこなわれた後，削除された領域の直前（*label1* を含む要素）に設定される．

もしサーチの開始点とストラクチャの最後の間に，設定されたラベルが見つからないときには，削除動作は発生せず，要素ポインタの変更もなされない．

8.4.5 ストラクチャ要素の複写

名前:

PEXCopyElements

リクエスト:

src_s_id: STRUCTURE_ID
src_range: ELEMENT_RANGE
dest_s_id: STRUCTURE_ID
dest_position: ELEMENT_POS

エラー:

Structure, Value

説明:

本リクエストは *src_s_id* で指定されるストラクチャ内の *src_range* で示される領域から, *dest_s_id* で指定されるストラクチャ内の *dest_position* で示される位置へ要素を複写する. 複写される要素の範囲は *src_range* 内の *position1* と *position2* の *whence*/*offset* を用い, 求められる.

複写される要素範囲を規定するオフセットは, *src_range* の位置を用いて計算される. *whence* が *Beginning* であったときには, オフセットは *offset* に指定される数値そのものとなる. *whence* が *Current*であったときには, オフセットは *src_s_id* の要素ポインタに *offset* を加えた値となる. *whence* が *End* であったときには, オフセットはストラクチャの要素数に *offset* を加えた値となる. *offset* は負の値を取ることも許されている.

求められたオフセットが負であったときには, 複写動作が起こる以前に 0 に変更される. 求められたオフセットがソース・ストラクチャの要素数を越える値であったときには, ソース・ストラクチャの最終要素に設定される. 2つのオフセットのいずれが複写範囲の上限／下限を指定しようと問題とはならない. (つまり, *offset1* が *offset2* より大きくても要素の並びが逆転することはない.)

複写される要素の位置は, *dest_position* を用いて以下の規則にしたがい求められる. *whence* が *Beginning* であったときには, デスティネーション・オフセットは *offset* に指定される数値そのものとなる. *whence* が *Current* であったときには, オフセットは *dest_s_id* の要素ポインタに *offset* を加えた値となる. *whence* が *End* であったときにオフセットはデスティネーション・ストラクチャの要素数に *offset* を加えた値となる. *offset* は負の値を取ることも許されている.

src_s_id と *dest_s_id* を同一とすることは可能である. この場合, 複写処理は指定範囲を一時的な場所に複写し, その後, デスティネーションの複写先へ挿入する形でおこなわれる.

複写処理の後, 複写先の要素ポインタは, 挿入された要素群の最終要素を指すように変更される. 複写処理の間, *dest_s_id* の指定するストラクチャの編集モードは無効となる. 複写される要素は常にデスティネーション・ストラクチャに挿入され, 決して既に存在する要素と置換されることはない.

8.4.6 ストラクチャ参照の変更

名前:

PEXChangeStructureReferences

リクエスト:

old_s_id: STRUCTURE_ID
new_s_id: STRUCTURE_ID

エラー:

Structure

説明:

本リクエストは *old_s_id* で指定されるストラクチャを，サーバから参照する "execute structure" を *new_s_id* で指定されるストラクチャを参照する "execute structure" に変更する.

本リクエストが実行される前に存在した *new_s_id* で指定されるストラクチャへの参照は影響を受けない. *new_s_id* で指定されるストラクチャが既に "post" されている PHIGS ワークステーション・リソースではストラクチャはそのまま残され， *old_s_id* で指定されるストラクチャが "post" されていたときには "unpost" される. *new_s_id* で指定されるストラクチャが "post" されておらず， *old_s_id* で指定されるストラクチャが "post" されている PHIGS ワークステーション・リソースでは， *old_s_id* で指定されるストラクチャが "post" されたプライオリティと同じプライオリティで *new_s_id* で指定されるストラクチャが "post" され, *old_s_id* で指定されるストラクチャは "unpost"される. *old_s_id* で指定されるストラクチャへの参照があり, *new_s_id* で指定されるストラクチャが存在しなかったときには，エラーが返され, いかなるなる動作もおこなわれない.

9 名前セット

名前セット (name set)は，レンダリング，ピック，サーチの各オペレーション中において，フィルタ出力コマンド (filter output command)として使われる，PEX リソースである．名前セットは，サーバのメモリ・リソースを少々必要とするが，名前セットは大きさが可変なので，これはその内容に依存して変えることができる．

9.1 名前セット・リソース管理

名前セットは，X11 のリソースであり，X11 のすべての責任とアクセス権を受け継いでいる．ここでのリクエストは名前セット・リソースの生成・解放・複写を管理する．

9.1.1 名前セットの生成

名前:

PEXCreateNameSet

リクエスト:

ns_id: NAME_SET_ID

エラー:

IDChoise, Alloc

説明:

この要求は，*ns_id* によって指定された名前セット・リソースを生成する．生成された時，その名前セットは空のリストに初期化される．

9.1.2 名前セットの複写

名前:

PEXCopyNameSet

リクエスト:

src_ns_id: NAME_SET_ID
dest_ns_id: NAME_SET_ID

エラー:

NameSet

説明:

このリクエストは，*dest_ns_id* の中身を空にした後，ネームセット *src_ns_id* を，*dest_ns_id* に複写する．事前に *dest_ns_id* は正当なリソースとなっていなければならない．

9.1.3 名前セットの解放

名前:

PEXFreeNameSet

リクエスト:

ns_id:NAME_SET_ID

エラー:

NameSet

説明:

このリクエストは，リソース ID と名前セットとの関係を解消する．ネームセットの記憶領域は，それを参照するものがなくなった時に解放される．

9.2 名前セット問い合わせ

これらのリクエストは，名前セット・リソースの中身を調べる.

9.2.1 名前セットを得る

名前:

PEXGetNameSet

リクエスト:

ns_id: NAME_SET_ID

リプライ:

names: LISTofNAME

エラー:

NameSet

説明:

このリクエストは， *ns_id* によって指定された名前セットの中身を返す.

9.3 名前セットの更新

これらのリクエストは，名前セット・リソースの中身を更新するのに用いられる.

9.3.1 名前セットの変更

名前:

PEXChangeNameSet

リクエスト:

ns_id: NAME_SET_ID
action: {*Replace*, *Add*, *Remove*}
names: LISTofNAME

エラー:

NameSet, Value

説明:

このリクエストは，名前セット・リソースの中身を変更する．もし *action* が *Add* ならば，指定された名前のリストがその名前セットに付け加えられる．もし *action* が *Remove* ならば，指定された名前のリストが，そのネームセットから取り除かれる．もし *action* が *Replace* ならば，名前セット内のすべての名前が取り除かれ，指定された名前のリストがその名前セットに付け加えられる．もし名前セット中にない名前を削除しようとしたときには，その名前は無視され，エラーは返らない. *c1

10 サーチ・コンテキスト

サーチ・コンテキスト(search context)は，クライアントがストラクチャ・ネットワーク上において「サーチ操作(searching operation)」をおこなうことを可能とする PEX リソースである．この章では，サーチ・コンテキスト上で操作がどのように実行され，また，その操作がどのようにサーチ・コンテキストを使用するかについて述べる．PEX は，今のところクライアント側のストラクチャ・ネットワークのサーチのサポートについては，何も規定しない．

ここでのいくつかのリクエストは，サーチ・コンテキストの属性に影響を及ぼす．*item_mask* と *item_list* パラメタは，どの構成要素が影響を受けるかを指定する．*item_mask* の各ビットは，どの属性が影響を受けるか否かを指定する．サーチ・コンテキストの属性が，設定されたり問い合わせられたりした所では，*item_mask* の各設定ビットに対して対応する *item_list* のエントリが存在する．その結果，一つのリクエストによって，一つ以上のサーチ・コンテキストの属性に影響を及ぼすことが可能となる．

サーチ・コンテキストの 2 つの属性は，名前セット・リソース ID を含んでいる．もし名前セットが生成され，サーチ・コンテキストに結合され，そして解放された場合，その名前セットの中身は残る．それは，それがまだサーチコンテキストによって参照されているからである．しかし，サーチコンテクストが問い合わせされた時，その名前セットの ID に対して *AlreadyFreed* の値が返される．それは，参照可能である有効なリソース ID を持たないからである．

サーチ操作中に，文字列要素や注釈文字列要素に出会った時は，サーチが成功したか否かは，文字列の原点によってのみ決定される．[1]

サーチ・コンテキストの構成要素は，以下のとおりである．

属性名	データ型	省略値
search_pos	COORD_3D	(0.0, 0.0, 0.0)
search_dist	FLOAT	0.0
search_ceiling	CARD16	0
model_clip_flag	BOOLEAN	*False*
start_path	LISTofELEMENT_REF	*Null*
normal_list	LISTofNAME_SET_PAIR	*Null*
inverted_list	LISTofNAME_SET_PAIR	*Null*

サーチ・コンテキストの属性は，以下の通り定義される．

search_pos

この属性は，世界座標系（WC）中でのサーチ参照位置を指定する．

search_dist

この属性は， WC 中でのサーチ参照位置からの距離を指定する．探索は，出力要素がサーチ・フィルタ条件(search filter criteria)を満たし， かつ ，サーチ参照位置からの距離を満たす場合である．もし *search_dist* が 0 以下のとき，サーチ条件を満たすためには，要素は *search_pos* と交差しなければならない．

search_ceiling *c1

この属性は，サーチ操作の限界(ceiling)を定義する．サーチ限界は， *start_path* に含まれるリスト内における指標である．検索は， *search_ceiling* によって指定されたストラクチャの最後に到達した時に終了する．たとえば *start_path* 内に *n* 要素があり， *search_ceiling* が *n* に設定されているときは，サーチ探訪はリスト内の最後のストラクチャの終端で終了する．もしサーチ限界が 1 ならば，検索は限界なしでおこなわれ， *start_path* 内の最初のストラクチャの終端で終了する．

[1] 原点に限定することは， PHIGS が注釈文字列の空間的な順次サーチに対する意味付けと同じである．しかし PHIGS は文字列サーチのときには，文字列要素を囲む矩形を用いるように定義している．探訪時には，文字フォント 1，文字列精度 *stroke*，文字拡大因子 (character expansion factor) 1，文字間隔 0，という文字列属性が，矩形の空間的大きさを決める．この場合， PEX は文字列要素をサーチすることについては PHIGS の意味には従わない．現状の PEX ドキュメントを適時リリースするために，この課題は将来の版の PEX まで延ばしたい．

model_clip_flag

この属性は，サーチ操作中にモデリング・クリッピング (modeling clipping)をおこなうかどうかを指定する．*True* ならば，トラバース中に出会ったモデリング・クリッピング属性を用いて，モデリング・クリッピングが実行される．もし *False* ならば，探訪中に出会ったモデリング・クリッピング属性は，実際には無視され，モデリング・クリッピングは実行されない．

start_path

この属性は，サーチの開始点として用いられるストラクチャ・ネットワーク・パスを定義する．サーチは start path で示された次の要素から始まる．

normal_list

この属性は，サーチ操作中においてフィルタとして用いられる名前セット・リソース ID の対のリストである．もし *normal_list* が *Null* ならば，すべての名前セットは，「標準フィルタ・リスト (normal filter list)」によって受け付けられるものとする．

inverted_list

この属性は，サーチ操作中においてフィルタとして反転 (invert)されて用いられる名前セット・リソース ID のペアのリストである．もし *inverted_list* が *Null* ならば，すべてのネームセットは，「逆フィルタ・リスト (inverted filter list)」によって受け付けられるものとする．

10.1 サーチ・コンテキスト・リソース管理

サーチ・コンテキストは，X11 のリソースであり，X11 リソースへの責任とアクセス権を受け継ぐ．これらのリクエストは，サーチ・コンテキストの生成・解放・複写を管理する．

10.1.1 サーチ・コンテキストの生成

名前:

PEXCreateSearchContext

リクエスト:

fp_format: FLOAT_FORMAT
sc_id: SEARCH_CONTEXT_ID
item_mask: BITMASK
item_list: LISTofVALUE

エラー:

IDChoise, Value, FloatingPointFormat, Alloc, Path, NameSet

説明:

このリクエストは，*sc_id* によって指定されたサーチ・コンテキスト・リソースを生成する．*item_mask* は，リソースが生成された時に，明示的に設定されたサーチ・コンテキスト・リソースを定義する．*item_list* は，新たに生成されたサーチ・コンテキストを修正するのに用いられる値の対応するリストである．*item_list* 内の浮動小数点値は，*fp_ format* で指定された浮動小数点形式になる．

10.1.2 サーチ・コンテキストの複写

名前:

PEXCopySearchContext

リクエスト:

src_sc_id: SEARCH_CONTEXT_ID
dest_sc_id: SEARCH_CONTEXT_ID
item_mask: BITMASK

エラー:

SearchContext, Value

説明:

このリクエストは，ソースのサーチ・コンテキスト *src_sc_id* を，出力側のサーチ・コンテキスト *dest_sc_id* にコピーする．*dest_sc_id* は有効なリソースとして既に存在していなければならない．*item_mask* は，サーチ・コンテキストのどの値がコピーされるかを示す．

10.1.3 サーチ・コンテキストの解放

名前:

PEXFreeSearchContext

リクエスト:

sc_id: NAME_CONTEXT_ID

エラー:

SearchContext

説明:

このリクエストは，指定したサーチ・コンテキスト・リソースを削除し，それに関係した記憶領域を解放する．

10.2 サーチ・コンテキストの問い合わせ

ここでのリクエストは，サーチ・コンテキスト・リソースの属性を調べるのに使われる．

10.2.1 サーチ・コンテキストを得る

名前:

PEXGetSearchContext

リクエスト:

fp_format: FLOAT_FORMAT
sc_id: SEARCH_CONTEXT_ID
item_mask: BITMASK

リプライ:

item_list: LISTofVALUE

エラー:

SearchContext, FloatingPointFormat, Value

説明:

このリクエストは，*sc_id* によって指定されたサーチ・コンテキストの構成要素を返す．*item_mask* は，どの構成要素が問い合わせされ，かつ返されるかを指定する．指定された属性値は *item_list* に返される．*item_list* 内の浮動小数点値は，*fp_format* で指定された浮動小数点形式で返される．

10.3 サーチ・コンテキストの更新

ここでのリクエストは，サーチ・コンテキスト・リソースの属性を更新するのに用いられる．

10.3.1 サーチ・コンテキストの変更

名前:

PEXChangeSearchContext

リクエスト:

fp_format: FLOAT_FORMAT
sc_id: SEARCH_CONTEXT_ID
item_mask: BITMASK
item_list: LISTofVALUE

エラー:

SearchContext, Value, FloatingPointFormat, Path, NameSet

説明:

このリクエストは，サーチ・コンテキストの構成要素を変更する．*item_mask* と *item_list* は，どの構成要素を変更するかを指定する．*item_mask* の各ビットは，*item_list* 内の対応するエントリがあるか否かを示す．その結果，1つの **PEXChangeSearchContext** リクエストによって，一つ以上のサーチ・コンテキストの属性を変更できる．*item_list* 内の浮動小数点値は，*fp_format* で指定された浮動小数点形式になる．

10.4 ストラクチャ・ネットワーク検索

ここではサーチ・コンテキスト・リソースを使ってストラクチャ・ネットワークをサーチするためのリクエストについて説明する.

10.4.1 ネットワーク検索

名前:

PEXSearchNetwork

リクエスト:

sc_id: SEARCH_CONTEXT_ID

リプライ:

found_path: LISTofELEMENT_REF

エラー:

SearchContext, Path

説明:

このリクエストは, ストラクチャ・ネットワーク上で実行される世界座標系 (WC) 上での空間探索 (spatial search)をおこなう. このサーチ操作のパラメタは, サーチ・コンテキスト *sc_id* 内にある. サーチは, *sc_id* の *start_path* 属性によって示される要素の次から始まる. サーチは *search_ceiling* によって示されたストラクチャの終端に到達した時に終了する. サーチ条件に合致した最初の要素が *found_path* に返される. もし何も要素が見つからなければ, *found_path* は空になる. サーチが完了した後は, *sc_id* の *start_path* 属性が, *found_path* 内に返される値に設定される.

サーチの候補と考えられるストラクチャ要素にとって, 現在の名前セットは, サーチ・コンテキストの *normal_list* 内のすべての名前セット対によって受理されなければならない. また *inverted_list* 内のすべての名前セット対によって拒絶されなければならない. もし *normal_list* が *Null* ならば, すべての可能な名前セットは normal_list によって受理される. もし *inverted_list* が *Null* ならば, すべての名前セットは inverted_list によって受理される. その結果, 省略値の場合はすべての出力要素が考慮される.

11 PHIGS ワークステーション

PHIGS ワークステーション (PHIGS workstation)は，“ワークステーション”という PHIGS 抽象 (abstraction)にいくぶん似たように振る舞う一つの実体 (entity)内に，他のリソースを結合する PEX のリソースである．PHIGS ワークステーションは既に述べたレンダラ・リソースのすべての能力を組み込んである．PHIGS ワークステーションは，そのルックアップ・テーブルと名前セットとともに，レンダラのインスタンスを実質的に包含するだけでなく，レンダラ・リソースの機能以上のものも含んでいる．

この章では，PHIGS ワークステーション・リソース上でできる操作，および，PHIGS ワークステーション・リソースを使ってできる操作について述べる．以下に掲げる属性の他に，PHIGS ワークステーション・リソースは，インプリメンテーションに依存する数のピック装置記述子も含んでいる．これらは，次の「ピッキング」の章にあるように，リクエストとともに問い合わせされたり，設定されたりする．

PHIGS ワークステーションとレンダラにはいくつかの違いがある．視野 (view)を扱うための PHIGS の意味に一致させるという要望をみたすために，PHIGS ワークステーション・リソースは，定義された PHIGS ワークステーション・リクエストを通してのみアクセス可能な，組み込み視野テーブルを持っている．しかし，この視野テーブルはレンダラによって使われたものと機能的に同じであり，それに関する情報は視野型のテーブルについての情報を得るためのルックアップ・テーブル・リクエストを使って得られる．破壊された関連ドローアブルをもつ PHIGS ワークステーション・リソースは暗黙的に解放される．

ルックアップ・テーブルと名前セットは，PHIGS ワークステーション・リソースに拘束され，そして解放される．それらのリソースの内容は PHIGS ワークステーションによってまだ参照されるので，それらの内容は残る．しかし PHIGS ワークステーションの属性が問い合わされたときには，*AlreadyFreed* の値がそれらのリソースに対して返され，どれが参照できうるのかを示す有効なリソース ID をもはや持たないことを知らせる．

11.1 PHIGS ワークステーション・リソース管理

PHIGS ワークステーションは X11 のリソースであり，X11 リソースのアクセス権限および責任のすべてを持つ．これらのリクエストは PHIGS ワークステーション・リソースの生成，解放，複写を管理する．

11.1.1 PHIGS ワークステーションの生成

名前:

PEXCreatePhigsWKS

リクエスト:

wks_id: PHIGS_WKS_ID
drawable_id: DRAWABLE_ID
marker_bundle: LOOKUP_TABLE_ID
text_bundle: LOOKUP_TABLE_ID
line_bundle: LOOKUP_TABLE_ID
interior_bundle: LOOKUP_TABLE_ID
edge_bundle: LOOKUP_TABLE_ID
color_table: LOOKUP_TABLE_ID
depth_cue_table: LOOKUP_TABLE_ID
light_table: LOOKUP_TABLE_ID
color_approx_table: LOOKUP_TABLE_ID
pattern_table: LOOKUP_TABLE_ID
text_font_table: LOOKUP_TABLE_ID
highlight_inclusion: NAME_SET_ID
highlight_exclusion: NAME_SET_ID
invisibility_inclusion: NAME_SET_ID
invisibility_exclusion: NAME_SET_ID
buffer_mode: BUFFER_MODE

エラー:

IDChoice, Drawable, Match, LookupTable, NameSet, Alloc, Value

説明:

このリクエストは，*wks_id* で示された PHIGS ワークステーション・リソースを生成する．*drawable_id* で指定されたウィンドウまたはピクスマップは，新しく作られる PHIGS ワークステーション・リソースに対して関連付けられる．名前の付けられたテーブルや名前セットもまたレンダリングの際に使用するために，PHIGS ワークステーション・リソースに対して結合される．現在の，かつリクエストされた視野テーブル・エントリをサポートする視野テーブルは，PHIGS ワークステーションが作られたときに自動的に割り当てられる．リクエスト **PEXSetViewRep** および **PEXGetViewRep** は PHIGS ワークステーション・視野テーブルを変更したり問い合わされたりするのに使われる．*buffer_mode* 属性はワークステーションがシングル・バッファ・モードもしくはダブル・バッファ・モードで動作しているのかを示すのに使われる．もしダブル・バッファならば，ダブル・バッファリングをサポートするために，インプリメンテーション依存の形でそのドローアブルのために付加的な画像バッファが作られる．もし *drawable_id* がピクスマップならば，ダブル・バッファリングはされない．

11.1.2 PHIGS ワークステーションの解放

名前:

PEXFreePhigsWKS

リクエスト:

wks_id: PHIGS_WKS_ID

エラー:

PhigsWKS

説明:

このリクエストはダブル・バッファリングをサポートするために，PEX 拡張によって作られた画像バッファと PHIGS ワークステーション・リソースを消去する．このリソースに関連したメモリも同様に解放される．

11.2 PHIGS ワークステーションの問い合わせ

ここでは PHIGS ワークステーション・リソースについての情報を得るために使われるリクエストを定義する.

11.2.1 PHIGS ワークステーション情報を得る

名前:

PEXGetWKSInfo

リクエスト:

fp_format: FLOAT_FORMAT
wks_id: PHIGS_WKS_ID
item_mask: WKS_BITMASK

リプライ:

item_list: LISTofVALUE

エラー:

PhigsWKS, FloatingPointFormat, Value

説明:

このリクエストは *wks_id* によって指定された PHIGS ワークステーション・リソースの属性を返す. 浮動小数点の値は *fp_format* によって示される浮動小数点形式で返される. パラメータ *item_mask* はどの属性が返されるのを示している. *item_mask* に設定されたおのおののビットに対して, 対応する値が *item_list* 内に返される.

以下に示すテーブルでは, “create wks req.”という省略形は, パラメータが **PEXCreatePhigsWKS** リクエストによって設定されることを示している. “imp.def.”は省略値がインプリメンテーション依存であることを意味している. 返ってくる属性は次のようなものである:

属性名	データ型	省略値
display_update	DISPLAY_UPDATE	*VisualizeEach*
visual_state	VISUAL_STATE	*Correct*
display_surface	DISPLAY_STATE	*Empty*
view_update	UPDATE_STATE	*NotPending*
defined_views	LISTofTABLE_INDEX	View 0 defined
wks_update	UPDATE_STATE	*NotPending*
req_NPC_subvolume	NPC_SUBVOLUME	(0.0, 0.0, 0.0), (1.0, 1.0, 1.0)
cur_NPC_subvolume	NPC_SUBVOLUME	(0.0, 0.0, 0.0), (1.0, 1.0, 1.0)
req_wks_viewpt	VIEWPORT	{imp.dep., imp.dep., True}
cur_wks_viewpt	VIEWPORT	{imp.dep., imp.dep., True}
HLHSR_update	UPDATE_STATE	*NotPending*
req_HLHSR_mode	HLHSR_MODE	1
cur_HLHSR_mode	HLHSR_MODE	1
drawable_id	DRAWABLE_ID	create_wks_req.
marker_bundle	LOOKUP_TABLE_ID	create_wks_req.
text_bundle	LOOKUP_TABLE_ID	create_wks_req.
line_bundle	LOOKUP_TABLE_ID	create_wks_req.
interior_bundle	LOOKUP_TABLE_ID	create_wks_req.
edge_bundle	LOOKUP_TABLE_ID	create_wks_req.
color_table	LOOKUP_TABLE_ID	create_wks_req.
depth_cue_table	LOOKUP_TABLE_ID	create_wks_req.
light_table	LOOKUP_TABLE_ID	create_wks_req.
color_approx_table	LOOKUP_TABLE_ID	create_wks_req.
pattern_table	LOOKUP_TABLE_ID	create_wks_req.
tex_font_table	LOOKUP_TABLE_ID	create_wks_req.
highlight_inclusion	NAME_SET_ID	create_wks_req.
highlight_exclusion	NAME_SET_ID	create_wks_req.
invisibility_inclusion	NAME_SET_ID	create_wks_req.
invisibility_exclusion	NAME_SET_ID	create_wks_req.
posted_structs	LISTofSTRUCTURE_INFO	*Null*
num_priorities	CARD32	imp.dep.
buffer_update	UPDATE_STATE	*NotPending*
req_buffer_mode	BUFFER_MODE	create_wks_req.
cur_buffer_mode	BUFFER_MODE	create_whs_req.

display_update

PHIGS ワークステーションの現在のディスプレイ更新モードを返す．このモードはどのように PHIGS ワークステーションが変化を見せようとするかを示しており，**PEXSetDisplayUpdateMode** リクエストによって設定される．

visual_state

PHIGS ワークステーションの現在のビジュアル状態を返す．もし後まわしにされた動作がなく，かつ変化がディスプレイ上にシミュレートされていなければ，ビジュアル状態は *Correct* である．もし後まわしにされた動作がある，または後まわしにされた動作があっていくつかの変化がディスプレイ上にすでにシミュレートされているなら，ビジュアル状態は *Deferred* である．もし，後まわしにされたアクションはないがいくつかの変化がディスプレイ上にシミュレートされたならば，ビジュアル状態は *Simulate* である．

display_surface

PHIGS ワークステーションのディスプレイの現在の状態を返す．ドローアブル上になにも表示されていなけれ

ば，ディスプレイは *Empty* であり，そうでなければ， *NotEmpty* である.

view_update
まだカレントにされるべき視野の変更リクエストがあるかどうかによって， *Pending* もしくは *NotPending* を返す.

defined_views
PHIGS ワークステーションの現在の視野テーブル内に定義されている視野テーブル指標のリストを返す. 定義された視野テーブル指標のリストは視野変換入力優先順位の順で返される.

wks_update
まだカレントにされるべきワークステーション変換リクエストがあるかないかによって， *Pending* もしくは *NotPending* を返す.

req_NPC_subvolume
リクエストはされたがまだカレントとなっていない PHIGS ワークステーションの NPC サブボリュームの設定値を返す.

cur_NPC_subvolume
PHIGS ワークステーションの NPC サブボリュームの現在の設定値を返す.

req_wks_viewpt
リクエストはされたがまだカレントとはなっていない PHIGS ワークステーションのビューポート設定に対する値を返す.

cur_wks_viewpt
PHIGS ワークステーションのビューポート設定値を返す.

HLHSR_update
まだカレントになるべきである HLHSR モード・リクエストがあるかないかによって， *Pending* または *NotPending* を返す.

req_HLHSR_mode
リクエストされたがカレントになっていない PHIGS ワークステーションの HLHSR モードに対する値を返す.

cur_HLHSR_mode
PHIGS ワークステーションの HLHSR モードの現在値を返す.

drawable_id
ディスプレイとして使われているドローアブルのリソース ID を返す.

marker_bundle
PHIGS ワークステーションのマーカ束テーブルのリソース ID を返す.

text_bundle
PHIGS ワークステーションの文字列束テーブルのリソース ID を返す.

line_bundle
PHIGS ワークステーションの線束テーブルのリソース ID を返す.

interior_bundle
PHIGS ワークステーションの内部様式束テーブルのリソース ID を返す.

edge_bundle
PHIGS ワークステーションの外周線束テーブルのリソース ID を返す.

color_table
指標付けられた色値を参照するために PHIGS ワークステーションによって使われる色ルックアップ・テーブルのリソース ID を返す.

depth_cue_table
PHIGS ワークステーションのデプス・キュー・テーブルのリソース ID を返す.

light_table
PHIGS ワークステーションの光源テーブルのリソース ID を返す.

color_approx_table
PHIGS ワークステーションによって使われる色近似テーブルのリソース ID を返す.

pattern_table
PHIGS ワークステーションのパターン・テーブルのリソース ID を返す.

text_font_table
PHIGS ワークステーションの文字列フォント・テーブルのリソース ID を返す.

highlight_inclusion
PHIGS ワークステーションの顕示包含名前セットのリソース ID を返す.

highlight_exclusion
PHIGS ワークステーションの顕示除外名前セットのリソース ID を返す.

invisibility_inclusion
PHIGS ワークステーションの不可視包含名前セットのリソース ID を返す.

invisibility_exclusion
PHIGS ワークステーションの不可視除外名前セットのリソース ID を返す.

posted_structs
PHIGS ワークステーションのポストされたストラクチャ・リスト内のストラクチャの，それぞれに対する優先順位とリソース ID を返す．そのリストはストラクチャ優先順で返される.

num_priorities
サポートされるディスプレイ優先権の数を返す．値 0 は連続した優先権がサポートされることを示す.

buffer_update
まだカレントにされるべきバッファ・モード・リクエストがあるかどうかにより，*Pending* もしくは *NotPending* を返す.

req_buffer_mode
リクエストされたがカレントにはなっていない PHIGS ワークステーションのバッファ・モードに対する値を返す.

cur_buffer_mode
PHIGS ワークステーションのバッファ・モードに対する値，*Single* か *Double* かのどちらか，を返す． PHIGS ワークステーション・リソースによって関連しているドローアブルがピクスマップならば，ダブル・バッファリングはおこなわれない.

11.2.2 Dynamic を得る

名前:

PEXGetDynamics

リクエスト:

drawable_id: DRAWABLW_ID

リプライ:

view_rep: DYNAMIC_TYPE
marker_bundle: DYNAMIC_TYPE
text_bundle: DYNAMIC_TYPE
line_bundle: DYNAMIC_TYPE
interior_bundle: DYNAMIC_TYPE
edge_bundle: DYNAMIC_TYPE
color_table: DYNAMIC_TYPE
pattern_table: DYNAMIC_TYPE
wks_transform: DYNAMIC_TYPE
highlight_filter: DYNAMIC_TYPE
invisibility_filter: DYNAMIC_TYPE
HLHSR_mode: DYNAMIC_TYPE
structure_modify: DYNAMIC_TYPE
post_structure: DYNAMIC_TYPE
unpost_structure: DYNAMIC_TYPE
delete_structure: DYNAMIC_TYPE
reference_modify: DYNAMIC_TYPE
buffer_modify: DYNAMIC_TYPE
light_table: DYNAMIC_TYPE
depth_cue_table: DYNAMIC_TYPE
color_approx_table: DYNAMIC_TYPE

エラー:

Drawable, Match *c1

説明:

このリクエストは *drawable_id* によって指定された型のドローアブルに結び付いた, PHIGS ワークステーションにサポートされているダイナミックスに関する情報を返す. ダイナミックスのリストは順に以下のようになっている.

属性名	説明
view_rep	視野テーブルに切替え
marker_bundle	マーカ束に切替え
text_bundle	文字列束に切替え
line_bundle	線束に切替え
interior_bundle	内部様式束に切替え
edge_bundle	外周線束に切替え
color_table	色テーブルに切替え
pattern_table	パターン・テーブルに切替え
wks_transform	ワークステーション変換に切替え
highlight_filter	顕示名前セットに切替え
invisibility_filter	不可視名前セットに切替え
HLHSR_mode	HLHSR モードに切替え
structure_modify	ストラクチャの修正 (編集)
post_structure	ポストされたストラクチャ・リストへの追加
unpost_structure	ポストされたストラクチャ・リストからの削除
delete_structure	ストラクチャ・リソースの削除
reference_modify	ストラクチャ参照修正
buffer_modify	バッファリング・モードに切替え
light_table	光源テーブルに切替え
depth_cue_table	デプス・キュー・テーブルに切替え
color_approx_table	色近似テーブルに切替え

上記リスト内の項目のおのおのから得られる値は *IMM*, *IRG*, *CBG* のどれかである. ここで *IMM* は示された動作が実行可能であり, 正しい画像が直ちに描かれうることを意味し, *IRG* は指示された動作が画像の再生成を必要とすることを, *CBS* は指示された動作が PHIGS ワークステーションの *display_update* モードが *SimulateSome* に設定されているならば, すぐにシミュレートできることを意味している.

11.2.3 視野表現を得る

名前:

PEXGetViewRep

リクエスト:

fp_format: FLOAT_FORMAT
wks_ID: PHIGS_WKS_ID
index: TABLE_INDEX

リプライ:

view_update: {Pending, NotPending}
requested: VIEW_REP
current: VIEW_REP

エラー:

PhigsWKS, FloatingPointFormat, Value

説明:

このリクエストは視野更新状態の値，リクエストされた特定のエントリ，そして *wkd_id* で指定された PHIGS ワークステーション内の現在の視野テーブルの値を返す．*view_update* は，もし視野の変化がリクエストされたものの，確立していないならば，*Pending*である．示されたエントリがまだ定義されていなければ，エラーが生じて応答パラメータの内容は定義されない．浮動小数点は *fp_format* に指定された浮動小数点の形式で返る．

11.3 PHIGS ワークステーションの操作

ここでは PHIGS ワークステーション・リソースを修正するリクエストを説明する.

11.3.1 すべてのストラクチャの再表示

名前:

PEXRedrawAllStructures

リクエスト:

wks_id: PHIGS_WKS_ID

エラー:

PhigsWKS

説明:

このリクエストは *wkd_id* で指定された, PHIGS ワークステーション・リソース内にあるポストされたすべてのストラクチャを再表示する. 最初に, PHIGS ワークステーションの *display_surface* 属性が *NotEmpty*ならば, そのドローアブルがクリアされて, 色ルックアップ・テーブルのエントリ 0 の色がストアされる. そして, *view_update*, *wks_update*, *HLHSR_update*, *buffer_update* 属性のどれかが *Pending* に設定されているならば, リクエストされた値を現在値とし, 属性を *NotPending* にする. その後, すべてのポストされたストラクチャが探訪されて (優先順位にしたがって) 表示される. 最後に, PHIGS ワークステーションの *visual_state* 属性が *Correct* に設定される.

11.3.2 ワークステーションの更新

名前:

PEXUpdateWorkstation

リクエスト:

wks_id: PHIGS_WKS_ID

エラー:

PhigsWKS

説明:

このリクエストは, *visual_state* 属性が *Deferred* もしくは *Simulated* に現在設定されているとき, *wks_id* によって指定される PHIGS ワークステーションについて **PEXRedrawAllStructures** と同一の動作をする.

11.3.3 クリップ領域の再表示

名前:

PEXRedrawClipRegion

リクエスト:

wks_id: PHIGS_WKS_ID
clip_list: LISTofDEVICE_RECT

エラー:

PhigsWKS

説明:

このリクエストは，いかなる PHIGS ワークステーション状態属性もこのリクエストによって更新または修正されないことを除いては，**PEXRedrawAllStructures** リクエストと似た動作をする．表示は *clip_list* によって定義された領域内でのみおこなわれる．*wkd_id* の色ルックアップ・テーブルのエントリ 0 にある色を使って *clip_list* によって定義される矩形領域がクリアされる．*clip_list* 内の矩形は，グラフィックスの結果が未定義であるため，重なりあわない．（もし z-バッファ・アルゴリズムが使われるならば，クリップ・リスト内の矩形内の画素のみが初期化された z 値を持つことになる．） *wks_id* に対してポストされたすべてのストラクチャが再表示されるが，それは *clip_list* 内の矩形の下の画素セルにしか関与しない．Pending の変化はこのリクエストによってカレントにされないし，また *visual_state* 属性も修正されない．

11.3.4 実行遅延動作

名前:

PEXExecuteDeferredActions

リクエスト:

wks_id: PHIGS_WKS_ID

エラー:

PhigsWKS

説明:

このリクエストにより， *wkd_id* によって指定された PHIGS ワークステーション上のすべての遅延された動作が実行される．

11.3.5 視野優先順位の設定

名前:

PEXSetViewPriority

リクエスト:

wks_id: PHIGS_WKS_ID
index1: TABLE_INDEX
index2: TABLE_INDEX
priority: {*Higher*, *Lower*}

エラー:

PhigsWKS, Value

説明:

このリクエストは *wkd_id* の現在の視野テーブル内のエントリについて相対的な優先順位を定める．視野テーブル・エントリ *index2* に対する視野テーブル・エントリ *index1* の優先順位は， *Higher* または *Lower*であるという優先順位で定められる．これらの優先順位は，視野テーブル・エントリが装置座標から世界座標への変換のために使う逆視野変換を選ぶときに，視野テーブル・エントリを試すのに使う．

11.3.6 ディスプレイ更新モードの設定

名前:

PEXSetDisplayUpdateMode

リクエスト:

wks_id: PHIGS_WKS_ID
display_update: DISPLAY_UPDATE

エラー:

PhigsWKS, Value

説明:

このリクエストは *wks_id* によって指定された PHIGS ワークステーションのリソースの *display_update* 属性を定める．この属性はディスプレイにどのような変化が表示されるかを定義する．許可されるディスプレイ・モードのリストは，列挙された型を説明する「拡張情報」の章に定義している．

もしダブル・バッファリング (double buffering)が可能（**PEXSetWKSBufferMode** を参照）ならば，ディスプレイ更新モードはどのバッファが表示されるかに影響する．ディスプレイ更新モードが *VisualizeNone*, *VisualizeEach* もしくは *VisualizeWhenever* であるならば，出力プリミティブはストラクチャ・ネットワーク が探訪されている間，背後の（ディスプレイされない）バッファに表示される．探訪が終わると，前面と背面のバッファを交換して表示イメージをディスプレイする．ディスプレイ更新モードが *VisualizeEasy* もしくは *SimulateSome* ならば，出力プリミティブは前の（見えている）バッファに表示される． *c2

11.3.7 DC から WC への写像

名前:

PEXMapDCtoWC

リクエスト:

fp_format: FLOAT_FORMAT
wks_id: PHIGS_WKS_ID
dc_points: LISTofDEVICE_COORD

応答:

wc_points: LISTofCORRD_3D
view_index: TABLE_INDEX

エラー:

PhigsWKS, FloatingPointFormat

説明:

このリクエストは， *dc_point* 内の装置座標の点を， *wks_id* によって指定された PHIGS ワークステーションのリソースを使って， *wc_points* 内の世界座標の点に写像するものである．（クライアントはドローアブル座標内のポインタの位置を装置座標に変換しなければならない．） PHIGS ワークステーションの現在の視野テーブル内のおのおのの視野は，それが逆行列 (inverse)を持っているならば，すべての指定された装置座標の点を含んでいるかどうかを調べられる．（もし視野変換が逆行列を持たないならば，これは考えない．） すべての点を含む最も高い視野変換入力優先順位を持つ視野の指標が *view_index* に返される．どの視野もすべての点を含んでいないならば，最も多くの点を含んでいる視野の指標が返される．それらの点は，視野指標によって結びついた視野変換の逆を通って世界座標に変換され， *wc_points* 内に返される．浮動小数点の値が渡されると， *fp_format* で指定された浮動小数点形式で返ってくる．クリップされた（ビューポート外側）の点は変換されないし，また *wc_point* リスト内に返ってこない．それゆえ，返ってくる点はの数は送った点の数よりも少ないこともある． *c2 *c1

11.3.8 WC から DC への写像

名前:

PEXMapWCtoDC

リクエスト:

fp_format: FLOAT_FORMAT
wks_id: PHIGS_WKS_ID
wc_points: LISTofCOORD_3D
view_index: TABLE_INDEX

応答:

dc_points: LISTofDEVICE_COORD

エラー:

PhigsWKS, FloatingPointFormat

説明:

このリクエストは, *wkd_id* によって指定された PHIGS ワークステーション・リソースと *view_index* によって指定された視野とを使って, *wc_points* の中の世界座標の点を *dc_points* の中に装置座標の点に写像する. それらの点は *view_index* によって結びついた視野変換を通して, 装置座標に変換される. 送って受け取った浮動小数点の値は *fp_format* によって指定された浮動小数点形式になっている. クリップされた点は *dc_points* 内には返ってこないので, 返ってくる点の数は送った数より少ないこともある.

11.4 PHIGS ワークステーションの更新

ここでは PHIGS ワークステーション・リソースに対する“リクエストされた”値を定めるのに使われるリクエストを定義する.

11.4.1 視野表現の設定

名前:

PEXSetViewRep

リクエスト:

fp_format: FLOAT_FORMAT
wks_id: PHIGS_WKS_ID
view_rep: VIEW_REP

エラー:

PhigsWKS, FloatingPointFormat, Alloc

説明:

このリクエストは, *wks_id* によって指定された PHIGS ワークステーション内の視野テーブルの指定された視野テーブル・エントリのリクエストされた値を, *view_rep* に示された視野表現に設定する. PHIGS ワークステーションの *view_update* 属性には, その変化がすぐに可視化できないときには *Pending* を設定し, それ以外のときには *NotPending* を設定する. もし *view_update* 属性が *NotPending* ならば, 現在の視野テーブル・エントリはリクエストされた値に設定される. そうでないときには, PHIGS ワークステーションが更新されるまで現在値は変更されない.

11.4.2 ワークステーション・ウィンドウの設定

名前:

PEXSetWKSWindow

リクエスト:

fp_format: FLOAT_FORMAT
wks_id: PHIGS_WKS_ID
NPC_subvolume: NPC_SUBVOLUME

エラー:

PhigsWKS, FloatingPointFormat

説明:

このリクエストは, *wks_id* に指定された PHIGS ワークステーションの *req_NPC_subvolume* を *NPC_subvolume* の中の値に設定する. PHIGS ワークステーション中の *wks_update* 属性は, もし変化がすぐに可視化できないならば *Pending* に設定され, それ以外のときは *NotPending* に設定される. もし *wks_id* の属性が *NotPending* ならば, *cur_NPC_subvolume* はリクエストされた値に設定される. またそうでなければ, PHIGS ワークステーションが更新されるまで現在値は変更されない.

11.4.3 ワークステーション・ビューポートの設定

名前:

PEXSetWKSViewport

リクエスト:

fp_format: FLOAT_FORMAT
wks_id: PHIGS_WKS_ID
viewport: VIEWPORT

エラー:

PhigsWKS, FloatingPointFormat, Value

説明:

このリクエストは *wks_id* で指定された PHIGS ワークステーションの *req_wks_viewport* を *viewport* 内の値に設定する. PHIGS ワークステーション内の *wks_update* は変化がただちに可視化できないならば *Pending* に設定され, そうでないならば *NotPending* に設定される. もし *wks_update* 属性が *NotPending* ならば, *cur_wks_viewport* はリクエストされた値に設定される. そうでなければ, PHIGS ワークステーションが更新されるまで現在の値は変更されない.

11.4.4 HLHSR モードの設定

名前:

PEXSetHLHSRMode

リクエスト:

wks_id: PHIGS_WKS_ID
mode: HLHSR_MODE

エラー:

PhigsWKS, Value, Alloc *c1

説明:

このリクエストは *wks_id* で指定された PHIGS ワークステーションの *req_HLHSR_mode* を *mode* 内の値に設定する. もし PHIGS ワークステーションの *display_surface* 属性が *Empty* ならば, または, その *HLHSR_mode* の動的修正が *IMM* ならば, その *cur_HLHSR_mode* 属性が *mode* 内に含まれる値で修正され, *HLHSR_update* が *NotPending* に設定される. そうでなければ, *HLHSR_mode* は *Pending* に設定され, 現在の値は PHIGS ワークステーションが更新されるまで変更されない.

11.4.5 バッファ・モードの設定

名前:

PEXSetWKSBufferMode

リクエスト:

wks_id: PHIGS_WKS_ID
buffer_mode: BUFFER_MODE

エラー:

PhigsWKS, Value, Alloc

説明:

このリクエストは *wks_id* で指定された PHIGS ワークステーションの *req_buffer_mode* を *buffer_mode* の値に設定する．もし，PHIGS ワークステーションの *display_surface* モードが *Empty* ならば，または *buffer_mode* に対する動的修正が *IMM* ならば，*cur_buffer_mode* 属性は *buffer_mode* 内に含まれる値を使って修正され，*buffer_update* は *NotPending* に設定される．それ以外のときには，*buffer_update* は *Pending* に設定され，現在値は PHIGS ワークステーションが更新されるまで変更されない．

buffer_mode は *Single*（レンダリングはシングル・バッファでおこなわれる）もしくは *Double*（レンダリングはダブル・バッファでおこなわれる）のうちのどれかである．*Alloc* エラーはリクエストされたモードが *Double* であって，かつサーバが画像バッファをこれ以上，領域確保できないときに返される．

11.5 ストラクチャのポスト / アンポスト

ここでは PHIGS ワークステーション・リソース上にストラクチャをポスト・アンポストをするのに使われるリクエストを説明する.

11.5.1 ストラクチャのポスト

名前:

PEXPostStructure

リクエスト:

fp_format: FLOAT_FORMAT
wks_id: PHIGS_WKS_ID
s_id: STRUCTURE_ID
priority: FLOAT

エラー:

PhigsWKS, Structure, FloatPointFormat

説明:

このリクエストは *s_id* によって指定されるストラクチャを PHIGS ワークステーション *wks_id* の中にポストされたストラクチャのリストに加える. 優先順位もまた既にポストされたストラクチャ・リスト内のストラクチャに対して, 新しくポストされたストラクチャの優先順位を示すために定められる. 多重ストラクチャが同じディスプレイ位置にポストされるならば, インプリメンテーションは更に高い優先順位のストラクチャのほうを表示することを保証する. もし 2 つのストラクチャが同じ優先順位を持っているならば, 後にポストされたストラクチャが高い優先順位を持つ. もし *s_id* が有効なリソース ID でない場合には, リクエストは無視され, エラーが生じる.

11.5.2 ストラクチャのアンポスト

名前:

PEXUnpostStructure

リクエスト:

wks_id: PHIGS_WKS_ID
s_id: STRUCTURE_ID

エラー:

PhigsWKS, Structure

説明:

このリクエストは *wkd_id* のポストされたストラクチャ・リストから *s_id* で指定したストラクチャを取り除く. もし *s_id* がポストされたストラクチャ・リスト内に見つからなければ, リクエストは無視され, エラーが生じる.

11.5.3 すべてのストラクチャのアンポスト

名前:

PEXUnpostAllStructures

リクエスト:

wks_id: PHIGS_WKS_ID

エラー:

PhigsWKS

説明:

このリクエストはすべてのストラクチャを *wks_id* のポストされたストラクチャ・リストから取り除く.

11.5.4　ポストしている PHIGS ワークステーションを得る

名前:

PEXGetWKSPostings

リクエスト:

s_id: STRUCTURE_ID

応答:

wks_id: LISTofPHIGS_WKS_ID

エラー:

Structure

説明:

このリクエストは *s_id* をポストしたすべての PHIGS ワークステーション・リソースのリストを返す.

12　ワークステーション・ピッキング

*c1

ワークステーション・ピッキング(workstation picking)の議論は，PHIGS ワークステーションにあるピック装置記述子(*pick device descriptor*)と，ピックメジャ(*pick measure*)の両方を含んでいる．

12.1　ピック装置記述子

各 PHIGS ワークステーション・リソースは，*pick device* 記述子のリストを保持している．このリスト内の各エントリは，マウスや 3D タブレットのようなピック装置の特別な型の状態値を保持する．さらに，ピック装置記述子リスト内のエントリは，ワークステーション・リソースによってサポートされるすべてのピック装置の状態値を保持する．この値のリストは，各サポートされているピック装置ごとに設定あるいは問い合わせされる．

ピック装置記述子のうち，2 つの構成要素は名前セット・リソース ID である．もし名前セットが生成され，ピック装置記述子に結合され，そして解放されると，その名前セットの中身は残る．それは，まだピックデバイス記述子によって参照されているからである．しかし，ピック装置記述子が問い合わせされた場合，名前セット ID に対して *AlreadyFreed* の値が返される．なぜならば，もはや参照されるべき有効なリソース ID を持っていないからである．

ピック装置記述子は，下記の表に順に掲げた．省略語 "pick dev. dep." は，"pick-device-dependent" の略で，ピック装置型によって決定される省略値を意味する．また省略語 "imp. dep" は，デフォルト値がインプリメンテーション依存という意味である．

属性名	データ型	省略値
pick_status	{*NoPick*, *Ok*}	*NoPick*
pick_path	LISTofPICK_ELEMENT_REF	*Null*
pick_path_order	{*TopFirst*, *BottomFirst*}	*TopFirst*
pick_inclusion	NAME_SET_ID	*Null*
pick_exclusion	NAME_SET_ID	*Null*
pack_data_rec	LISTofCARD8	pick dev. dep.
prompt_echo_type	PROMPT_ECHO_TYPE	pick dev. dep.
echo_volume	VIEWPORT	{imp.dep., imp.dep., True}
echo_switch	{*NoEcho*, *Echo*}	*NoEcho*

ピック装置記述子の構成要素は，以下の通りである．

pick_status
この属性は，ピックメジャ・リソースに結合されるであろう初期ピック状態である．

pick_path
この属性は，ピックメジャ・リソースに結合されるであろう初期ピック・パスである．

pick_path_order
この属性は，ピックメジャ・リソース内に書かれたピック・パスの要素の順序を指定する．もしその順序が *TopFirst* ならば，ピックメジャの *picked_path* 属性の要素は，探訪中に出会った順序でリストされる．その順序が *BottomFirst* ならば，その逆順でリストされる．

pick_inclusion
この属性は，ピック操作中にピック包含フィルタとして用いられるネームセット・リソースのリソース ID を指定する．

pick_exclusion
この属性は，ピック操作中にピック除外フィルタとして用いられるネームセット・リソースのリソース ID を指定する．

pick_data_rec
この属性は，ピックメジャ・リソースが生成された際に初期値として用いられる，ピック装置依存のデータ・レコードである．

prompt_echo_type

この属性は，ピック操作中で用いられる prompt/echo 型を指定する列挙型の値である．その許される型は「拡張情報」の章に書かれている．

echo_volume

この属性は，プロンプティング and/or エコーイングが起こったところで指定される．デフォルトは，それが生成されたときの PHIGS ワークステーション・リソースに結合されたドローアブルのサイズとして定義されたエコー・ボリュームである．

echo_switch

この属性は，ピックメジャの初期エコー状態を指定する．

ここでのリクエストは，クライアントに PHIGS ワークステーション・リソース内のピック装置記述子の値を，設定または問い合わせをすることができる．

12.1.1 ピック装置の記述子を得る

名前:

PEXGetPickDevice

リクエスト:

fp_format: FLOAT_FORMAT
wks_id: PHIGS_WKS_ID
dev_type: PICK_DEVICE_TYPE
item_mask: BITMASK

リプライ:

item_list: LISTofVALUE

エラー:

PhigsWKS, Value, FloatingPointFormat

説明:

このリクエストは, *wks_id* によって指定された PHIGS ワークステーション・リソースのためのピック記述子の構成要素を返す. その返された記述子は, *dev_type* によって指定された型のピック装置として, 現在定義されている記述子である. *item_mask* は, どの構成要素が問い合わされて返されるのかを指定する. その指定されたピック装置記述子の属性は *item_list* 内に返される. *item_list* 内の浮動小数点値は, *fp_format* で指定された浮動小数点形式で返される.

12.1.2 ピック装置記述子の変更

名前:

PEXChangePickDevice

リクエスト:

fp_format: FLOAT_FORMAT
wks_id: PHIGS_WKS_ID
dev_type: PICK_DEVICE_TYPE
item_mask: BITMASK
item_list: LISTofVALUE

エラー:

PhigsWKS, Value, FloatingPointFormat, Path, NameSet

説明:

このリクエストは, *wks_id* によって指定された PHIGS ワークステーション・リソースのためのピック記述子の構成要素を更新する. その更新される記述子は, *dev_type* によって指定された型のピック装置として, 現在定義されている記述子である. *item_ mask* はどの構成要素が変更されるのかを指定する. ピック装置記述子の指定された属性は *item_list* 内に含まれる値に設定される. *item_list* 内の浮動小数点値は, *fp_format* で指定された浮動小数点形式になる.

12.2 ピックメジャ

ピックメジャ (*pick measure*)・リソースは，実際にピック操作が実行されたら生成されなければならない．ピック装置の型は，ピック操作のパラメタを規定するためのピックメジャ・リソース生成時に指定される．ピックメジャ・リソースは，ピック装置の各型が定義されている入力レコードの形での入力を受け付ける．ピックメジャ・リソースが正当な入力レコードを通過させると，その属性は更新される．ピックメジャ・リソース上での操作は，潜在的に長くかかる．それは，大量のストラクチャ要素がピック結果を得るために処理されなければならないからである．

ピックメジャ・リソースの構成要素を下記の表に順に掲げた．省略語 "from wks" は，ピックメジャが生成された時に，PHIGS ワークステーション内にあったピック装置記述子から継承された省略値を示す．

属性名	データ型	省略値
pick_status	{*NoPick*, *Ok*}	from wks
picked_prim	LISTofPICK_ELEMENT_REF	from wks

ピックメジャ・リソースの属性は，以下の通り定義される．

pick_status

この属性は，最近実行された更新操作の結果である．もし要素のピックに成功した場合は，これが *Ok* に設定され，ピックに成功しなかった場合は，*NoPick* に設定される．

picked_prim

この属性は，最近の更新操作の結果としてピックされたストラクチャ要素のパスである．

ピックメジャは，X11 リソースであり，X11 のすべての責任とアクセス権を負う．ここでのリクエストは，ピックメジャの生成と解放を管理している．

12.2.1 ピックメジャの生成

名前:

PEXCreatePickMeasure

リクエスト:

pm_id: PICK_MEASURE_ID
wks_id: PHIGS_WKS_ID
dev_type: PICK_DEVICE_TYPE

エラー:

IDChoice, PhigsWKS, Alloc, Value

説明:

このリクエストは, *dev_type* によって指定された型のピックメジャ・リソースを生成する. ピックメジャのリソース ID は, *pm_id* によって指定される. 新規に生成されたピックメジャは, *wks_id* の PHIGS ワークステーション・リソース内に格納されている, 適切なピック装置記述子内に含まれている値に初期化される.

12.2.2 ピックメジャの解放

名前:

PEXFreePickMeasure

リクエスト:

pm_id: PICK_MEASURE_ID

エラー:

PickMeasure

説明:

このリクエストは, ピックメジャ・リソースを消去し, それに関連した記憶領域を解放する.

12.2.3 ピックメジャ属性を得る

名前:

PEXGetPickMeasure

リクエスト:

pm_id: PICK_MEASURE_ID
item_mask: BITMASK

リプライ:

item_list: LISTofVALUE

エラー:

PickMeasure, Value

説明:

このリクエストは, *pm_id* によって指定されたピックメジャの構成要素を返す. *item_mask* はどの構成要素が問い合わせされ, そして返されるかを指定する. これらの指定されたピックメジャの属性は, *item_list* 内に返される.

12.2.4 ピックメジャの更新

名前:

PEXUpdatePickMeasure

リクエスト:

pm_id: PICK_MEASURE_ID
input_record: LISTofCARD8

エラー:

PickMeasure, Path

説明:

このリクエストは, *pm_id* によって指定されたピックメジャを, *input_record* 内にある情報を用いて更新する. もしこの更新が要素の更新を引き起こすならば, ピックメジャの *pick_status* は *Ok* に設定され, かつ *picked_prim* はピックされた要素のパスに設定される. もしどの要素もピックされなければ, *pick_status* は *NoPick* に設定される.

pick_prim はピックメジャが生成されるときのエコーのために使用できる. しかしピックを始める開始パスにはならない. *p

input_record 内のデータは, ピックメジャ・リソースを作成したピック装置の型に依存している. この型は **PEXCreatePickMeasure** によって指定される *dev_type* である. **PEXGetPickMeasure** は **PEXUpdatePickMeasure** リクエストの結果を返す.

登録されているピック装置のための, ピックメジャ入力データ・レコードは以下のとおりである.

DC_HitBox	:[pick_position:DEVICE_COORD_2D, pick_distance:FLOAT]
NPC_HitVolume	:[pick_volume:NPC_SUBVOLUME]

13 PEX フォント

PEX フォント (PEX font)の操作メカニズムは，X11 フォントと同じである．PEX フォント・リソースは，X11 によって生成される（そして管理される）フォント・リソースによく似ている．PEX のストローク精度の文字列で使われる PEX フォントは，現在 X11 中に見られるものよりもより機能的である．特にストローク精度文字列は，拡縮・回転可能な文字列フォントであることを要求する．PEX フォントに付加された能力のため，PEX はそれ自身の PEX フォントのオープン，クローズ，問い合わせのリクエストを定義している．X11 フォント操作リクエストと，それに対応する PEX フォント操作リクエストは，以下の通りである．

OpenFont	PEXOpenFont
CloseFont	PEXCloseFont
ListFonts	PEXListFonts
ListFontsWithInfo	PEXListFontsWithInfo
QueryFont	PEXQueryFont
QueryTextExtents	PEXQueryTextExtents

X11 フォントと PEX フォントは，サーバの同じ場所に格納することができる．PEX は，PEX フォント・ファイルがどこに位置しているかを見つけるために，X11 フォント・パスを使う．（X11 リクエスト **SetFontPath** は，現在のフォント・パスをセットするのに使用され，また，X11 リクエスト **GetFontPath** は，現在のフォント・パスを問い合わせるのに使用される．）

13.1 PEX フォント・リソース管理

PEX フォントは一つの X11 リソースであり，X11 リソースとしてのすべてのアクセス権と責任を持つ．以下のリクエストは PEX フォント・リソースのオープンとクローズを管理する．

13.1.1 PEX フォントのオープン

名前:

PEXOpenFont

リクエスト:

f_id: PEX_FONT_ID
name: STRING

エラー:

PEXFont， IDChoice， Alloc

説明:

このリクエストは，必要ならば指定された PEX フォントをロードし，それと *identifierf_id* とを関係付ける．フォント名は ISO ラテン-1・エンコーディングを用いるべきであり，大文字または小文字は問題ではない．PEX フォントは，特定のスクリーンとは関連しておらず，また，どのレンダラもしくは PHIGS ワークステーション・リソースに対しても使用できる．もし指定されたフォントが，”PEX で使用可能”でない場合は，すなわち PEX 文字列属性の全レンジのサポートができない場合，エラーが生じる．

13.1.2 PEX フォントのクローズ

名前:

PEXCloseFont

リクエスト:

f_id: PEX_FONT_ID

エラー:

PEXFont

説明:

このリクエストは，リソース ID と PEX フォントとの間の関係を解消する． PEX フォント自身は，それを参照するリソースがなくなった時点で解放される．

13.2 PEX フォントの問い合わせ

PEX フォント・リクエストは，フォント特有の論理情報を含むリプライを生成する．その情報は，以下のデータ構造でエンコーディングされる．

PEX_FONTPROP は，

name	:ATOM
value	:CARD32

として．また， PEX_FONTINFO は，

first_glyph	:CARD32
last_glyph	:CARD32
default_glyph	:CARD32
all_glyphs_exist	:BOOLEAN
stroke_font	:BOOLEAN
properties	:LISTofPEX_FONTPROP

first_glyph， *last_glyph*， *default_glyph* は，フォントの最初，最後，デフォルトのグリフ (glyph)への指標である． *default_glyph* は，未定義または存在しないグリフが使用された時に表示されるグリフを指定する． *default_glyph* は，存在しないグリフを指定しても良い．この場合，もしクライアントが未定義または存在しないグリフを表示しようとしても，何も表示されない．もし *all_glyphs_exist* が *True* ならば， *first_glyph* と *last_glyph* の間にあるすべてのグリフは，非ゼロの存在範囲を持つ． *stroke_font* はフォントが PEX フォントであり，クライアントがすべて同じ文字列精度でフォント・グループを構築できることを保証するフラグである．

ある一つのフォントは，どんなプロパティ (properties)を持つことも保証していない．プロパティ値が，符号の有り／無しであるかどうか，そしてどのユニットであるかは，プロパティの事前知識から引き出されねばならない．すべてのフォントが，少なくとも CHARSET_REGISTRY プロパティ（たとえば ”ISO8859”）と， CHARSET_ENCODINGロパティ（たとえば“1”）を持つことを強く推奨する．

13.2.1 PEX フォントの問い合わせ

名前:

PEXQueryFont

リクエスト:

f_id: PEX_FONT_ID

リプライ:

font_info: PEX_FONTINFO

エラー:

PEXFont

説明:

このリクエストは， PEX フォントについての論理情報を含むリプライを生成する．

13.2.2 PEX フォントのリスト

名前:

PEXListFonts

リクエスト:

pattern: STRING
max_names: CARD16

リプライ:

font_names: LISTofSTRING

エラー:

None

説明:

このリクエストは，PEX 文字列属性（すなわち“PEX で使用可能な”）の全レンジをサポートできるフォントの名前のみを返すところを除いては，X11 の **ListFonts** と似ている．このリストは，X11 の **ListFonts** リクエストによって返されるのと同じフォントのいくつかを含んでも含まなくてもよい．このリクエストは，高々 *max_name* 個のエントリのリストからなる．この各エントリは，*pattern* とのフォント・マッチングについての情報を含んでいる． *pattern* は ISO ラテン -1・エンコーディングを用いる文字列であり，大文字または小文字は問題ではない．パターンの中で '?' 文字（8 進数で 77）は，どの一文字ともマッチする．また '*' 文字 (8 進数で 52) はどの何文字ともマッチする．返される名前は小文字であり，ISO ラテン -1・エンコーディングにしたがった文字列である．

13.2.3 PEX フォント情報のリスト

名前:

PEXListFontsWithInfo

リクエスト:

pattern: STRING
max_names: CARD16

リプライ:

font_names: LISTofSTRING
fonts: LISTofPEX_FONTINFO

エラー:

None

説明:

このリクエストは，PEX 文字列属性（すなわち“PEX で使用可能な”）の全レンジをサポートできるフォントの名前のみを返すところを除いては，X11 の **ListFontsWithInfo** と似ている．このリストは X11 の **ListFonts** リクエストによって返されるのと同じフォントのいくつかを含んでも含まなくてもよい．このリクエストは，高々 *max_name* 個のエントリのリストからなる．この各エントリは， *pattern* とのフォント・マッチングについての情報を含んでいる． *pattern* は ISO ラテン -1・エンコーディングを用いる文字列であり，大文字または小文字は問題ではない．パターンの中で '?' 文字（8 進数で 77）は，どの一文字ともマッチする．また '*' 文字（8 進数で 52）はどの何文字ともマッチする．返される名前は小文字であり， ISO ラテン -1・エンコーディングにしたがった文字列である．この情報は **PEXQueryFont** が返すであろうものと同じフォントを返す．

13.2.4 PEX 文字列範囲の問い合わせ

名前:

PEXQueryTextExtents

リクエスト:

fp_format: FLOAT_FORMAT
resource_id: RESOURCE_ID
font_group: TABLE_INDEX
path: CARD16
expansion: FLOAT
spacing: FLOAT
height: FLOAT
alignment: TEXT_ALIGNMENT
strings: LISTofSTRING

リプライ:

extents: LISTofEXTENT_INFO

エラー:

FloationgPointFormat, Value, Match

説明:

このリクエストは，指定された各文字列ごとのローカルな 2D 文字列座標系における領域範囲情報を含むリプライを生成する．*strings* 中の各文字列の範囲は，リスト中の他の文字列には無関係に計算される．もし *resource_id* がレンダラあるいは PHIGS ワークステーションならば，範囲の計算をするときに使われる *TextFont* テーブルはレンダラあるいは PHIGS ワークステーションに結びついたものである．もし *resource_id* が型 *TextFont* のルックアップ・テーブル・リソースであるならば，それは直接用いられる．*font_group* はフォント・グループを得るときに用いられるエントリの指標を用意する． *c2

ストローク精度が仮定される．文字列位置はローカルな 2D 文字列座標系においては (0, 0) である．*concat_point* はもし文字列が伸びていく場合，次のグリフがどこに行くべきかの点を返す．この位置は 2D 文字列座標系で与えられる．（文字の立ち上がり方向ベクトル (up vector)を考慮した，しかるべきモデリング変換は，クライアントによってなお適用される必要がある．）

指定されたフォントが定義された *default_glyph* を持たない（すなわち，もし *default_glyph* が存在しないグリフを参照した場合），*strings* 内の未定義のグリフは，すべて長さ 0 （メートル）を取る．

APPENDIX

A 標準 PEX サブセットの定義

*c1

PEXGetExtensionInfo は，PEX サーバがフルインプリメンテーションなのか，標準的なサブセットなのかを示す情報を持つ 32 ビット値（*subset_info*）を返す．この 32 ビット中，上位 16 ビットはベンダーの使用に予約されている．下位 16 ビットが全サポートか否かを示す．現在は 3 種類の PEX サブセットが定義されている．すなわち「即時レンダリング (immediate rendering)」，「PHIGS ワークステーション (PHIGS workstation)」，「ストラクチャ・レンダリング (structure rendering)」である．

もし *subset_info* の下位 16 ビットがすべて 0 であれば，完全なインプリメンテーションとみなされる．他の場合は以下のように解釈される．

設定ビット	意味
bit 0 set	サーバは "immediate rendering" の全リクエストをサポート
bit 1 set	サーバは "PHIGS workstation" の全リクエストをサポート
bit 2 set	サーバは "structure rendering" の全リクエストをサポート

PEX 5.0 への互換性のために，PEX のインプリメンテーションは bit 0 と bit 1 が共にセットされることは許されていない．

もしサーバが，サポートしているサブセットには ない リクエストを受け取ったならば，*Request* エラーを返す．

各サブセットは次の "yes" の項の PEX リソースをすべてサポートしなければならない．

またすべての出力コマンドをサポートしなければならない（ただし "execute structure" 出力コマンドは no-op として扱われる）．各サブセットとしては，少なくとも "yes" の項のリクエストをサポートしなければならない．サポートしていないリクエストには *Implementation* エラーを返さなければならない．

リソース	**Immediate Rendering**	**PHIGS WKS**	**Structure Rendering**
ルックアップ・テーブル	yes	yes	yes
名前セット	yes	yes	yes
PEX フォント	yes	yes	yes
レンダラ	yes	–	yes
パイプライン・コンテキスト	yes	–	yes
ストラクチャ	–	yes	yes
サーチ・コンテキスト	–	yes	yes
ピックメジャ	–	yes	–
PHIGS ワークステーション	–	yes	–

リクエスト	Immediate Rendering	PHIGS WKS	Structure Rendering
PEXGetExtensionInfo	yes	yes	yes
PEXGetEnumeratedTypeInfo	yes	yes	yes
PEXGetImpDepConstants	yes	yes	yes
PEXMatchRendererTargets	yes	yes	yes
PEXCreateLookupTable	yes	yes	yes
PEXCopyLookupTable	yes	yes	yes
PEXFreeLookupTable	yes	yes	yes
PEXGetTableInfo	yes	yes	yes
PEXGetPredefinedEntries	yes	yes	yes
PEXGetDefinedIndices	yes	yes	yes
PEXGetTableEntry	yes	yes	yes
PEXGetTableEntries	yes	yes	yes
PEXSetTableEntries	yes	yes	yes
PEXDeleteTableEntries	yes	yes	yes
PEXCreateStructure	–	yes	yes
PEXCopyStructure	–	yes	yes
PEXDestroyStructures	–	yes	yes
PEXGetStructureInfo	–	yes	yes
PEXGetElementInfo	–	yes	yes
PEXGetStructuresInNetwork	–	yes	yes
PEXGetAncestors	–	yes	yes
PEXGetDescendants	–	yes	yes
PEXFetchElements	–	yes	yes
PEXSetEditingMode	–	yes	yes
PEXSetElementPointer	–	yes	yes
PEXSetElementPointerAtLabel	–	yes	yes
PEXElementSearch	–	yes	yes
PEXStoreElements	–	yes	yes
PEXDeleteElements	–	yes	yes
PEXDeleteElementsToLabel	–	yes	yes
PEXDeleteElementsBetweenLabels	–	yes	yes
PEXCopyElements	–	yes	yes
PEXChangeStructureReferences	–	yes	yes
PEXCreatePipelineContext	yes	–	yes
PEXCopyPipelineContext	yes	–	yes
PEXFreePipelineContext	yes	–	yes
PEXGetPipelineContext	yes	–	yes
PEXChangePipelineContext	yes	–	yes

リクエスト	Immediate Rendering	PHIGS WKS	Structure Rendering
PEXBeginPickOne	yes	–	yes
PEXEndPickOne	yes	–	yes
PEXPickOne	–	–	yes
PEXBeginPickAll	yes	–	yes
PEXEndPickAll	yes	–	yes
PEXPickAll	–	–	yes
PEXGetPickDevice	–	yes	–
PEXChangePickDevice	–	yes	–
PEXCreatePickMeasure	–	yes	–
PEXFreePickMeasure	–	yes	–
PEXGetPickMeasure	–	yes	–
PEXUpdatePickMeasure	–	yes	–
PEXCreateRenderer	yes	–	yes
PEXFreeRenderer	yes	–	yes
PEXChangeRenderer	yes	–	yes
PEXGetRendererAttributes	yes	–	yes
PEXGetRendererDynamics	yes	–	yes
PEXBeginRendering	yes	–	yes
PEXEndRendering	yes	–	yes
PEXBeginStructure	yes	–	–
PEXEndStructure	yes	–	–
PEXRenderOutputCommands	yes	–	–
PEXRenderNetwork	–	–	yes
PEXRenderElements	–	–	yes
PEXAccumulateState	–	–	yes
PEXCreateNameSet	yes	yes	yes
PEXCopyNameSet	yes	yes	yes
PEXFreeNameSet	yes	yes	yes
PEXGetNameSet	yes	yes	yes
PEXChangeNameSet	yes	yes	yes
PEXCreateSearchContext	–	yes	yes
PEXCopySearchContext	–	yes	yes
PEXFreeSearchContext	–	yes	yes
PEXGetSearchContext	–	yes	yes
PEXChangeSearchContext	–	yes	yes
PEXSearchNetwork	–	yes	yes

リクエスト	Immediate Rendering	PHIGS WKS	Structure Rendering
PEXCreatePhigsWKS	–	yes	–
PEXFreePhigsWKS	–	yes	–
PEXGetWKSInfo	–	yes	–
PEXGetDynamics	–	yes	–
PEXGetViewRep	–	yes	–
PEXRedrawAllStructures	–	yes	–
PEXUpdateWorkstation	–	yes	–
PEXRedrawClipRegion	–	yes	–
PEXExecuteDeferredActions	–	yes	–
PEXSetViewPriority	–	yes	–
PEXSetDisplayUpdateMode	–	yes	–
PEXMapDCtoWC	–	yes	–
PEXMapWCtoDC	–	yes	–
PEXSetViewRep	–	yes	–
PEXSetWKSWindow	–	yes	–
PEXSetWKSViewport	–	yes	–
PEXSetHLHSRMode	–	yes	–
PEXSetWKSBufferMode	–	yes	–
PEXPostStructure	–	yes	–
PEXUnpostStructure	–	yes	–
PEXUnpostAllStructures	–	yes	–
PEXGetWKSPostings	–	yes	–
PEXQueryFont	yes	yes	yes
PEXListFontsWithInfo	yes	yes	yes
PEXQueryTextExtents	yes	yes	yes
PEXListFonts	yes	yes	yes
PEXOpenFont	yes	yes	yes
PEXCloseFont	yes	yes	yes

B PHIGS/PHIGS-PLUS のための最小サポート

*c1

PHIGS のクライアント側インプリメンテーションと，PHIGS-PLUS 機能のフルサポートのために，PEX 拡張インプリメンテーションは次の評価基準を満たすか越えなければならない．PEX 拡張インプリメンテーションは，ここに掲げた最小基準と Appendix A に掲げたサブセット情報の他に，*PEX Protocol Specification* ドキュメントに記載した機能をすべてサポートしなければならない． *c1

項目	最小基準
列挙型	
MarkerType	少なくとも 1-5 (*Dot*, *Cross*, *Asterisk*, *Circle*, *X*).
ATextStyle	少なくとも 1-2 (*NotConnected*, *Connected*).
InteriorStyle	少なくとも 1, 2, 5 (*Hollow*, *Solid*, *Empty*).
HatchStyle	必要はない.
LineType	少なくとも 1-4 (*Solid*, *Dashed*, *Dotted*, *DashDot*).
SurfaceEdgeType	少なくとも 1 (*Solid*).
PickDeviceType	少なくとも 1 (*DC_HitBox*).
	（フルインプリメンテーション，あるいは PHIGS WKS サブセット）
PolylineInterpMethod	少なくとも 1-2 (*None*, *Color*).
CurveApproxMethod	少なくとも 1-8 (*imp.dep.*, *ConstantBettweenKnots*, *WCS_ChordalSize*, *NPC_ChordalSize*, *DC_ChordalSize*, *WCS_ChordalDev*, *NPC_ChordalDev*, *DC_ChordalDev*).
ReflectionModel	少なくとも 1-4 (*NoShading*, *Ambient*, *Diffuse*, *Specular*).
SurfaceInterpMethod	少なくとも 1-2 (*None*, *Color*).
SurfaceApproxMethod	少なくとも 1-8 (*imp.dep.*, *ConstantBetweenKnots*, *WCS_ChordalSize*, *NPC_ChordalSize*, *DC_ChordalSize*, *WCS_PlanarDev*, *NPC_PlanarDev*, *DC_PlanarDev*).
TrimCurveApproxMethod	少なくとも 1-2 (*imp.dep.*, *ConstantBetweenKnots*).
ModelClipOperator	少なくとも 1-2 (*Replace*, *Intersection*).
LightType	少なくとも 1-4 (*Ambient*, *WCS_Vector*, *WCS_Point*, *WCS_Spot*).
ColorType	少なくとも 0-1 (*Indexed*, *RGBFloat*).
FloatFormat	少なくとも 1-2 の うち一つ (*IEEE_754_32*, *DEC_F_Floating*).
HLHSRMode	少なくとも 1 (*Off*) と他 (*ZBuffer*, *Painters*, *Scanline*, *HiddenLineOnly*, *imp.dep.Type*) の中の一つ.
PromptEchoType	少なくとも 1 (*EchoPrimitive*).
DisplayUpdateMode	少なくとも 1-3 (*VisualizeEach*, *VisualizeEasy*, *VisualizeNone*).
ColorApproxType	少なくとも 1-2 の うち一つ (*ColorSpace*, *ColorRange*).
ColorApproxModel	少なくとも 1-5 の うち一つ (*RGB*, *HSV*, *HLS*, *CIE*, *YIQ*).
RenderingColorModel	少なくとも 0 (*imp.dep.*).
ParametricSurfaceCharacteristics	少なくとも 1-3 (*None*, *imp.dep.*, *IsoparametricCurve*).
BufferMode	double buffering をサポートする必要はない.
GDP	必要はない.
GDP3	必要はない.
GSE	必要はない.

項目	最小基準
出力要素属性	
marker_scale	1.0 以外は無視してもよい.
text_precision	すべての型 (*Stroke*, *Char*, *String*).
char_expansion	1.0 以外は無視してもよい.
char_height	0.01 以外は無視してもよい.
atext_height	0.01 以外は無視してもよい.
line_width	1.0 以外は無視してもよい.
interior_style_index	*Pattern* と *Hatch* がともに未サポートならば無視してもよい.
reflection_attr	透過係数属性は無視してもよい.
bf_interior_style_index	*Pattern* と *Hatch* がともに未サポートならば無視してもよい.
bf_reflection_attr	透過係数属性は無視してもよい.
pattern_size	*Pattern* 様式が未サポートならば無視してもよい.
pattern_ref_pt	*Pattern* 様式が未サポートならば無視してもよい.
pattern_ref_vec1	*Pattern* 様式が未サポートならば無視してもよい.
pattern_ref_vec2	*Pattern* 様式が未サポートならば無視してもよい.
surface_edge_width	1.0 以外は無視してもよい.
model_clip_volume	少なくとも 6 つの半空間の組み合わせをサポートしなければならない.
depth_cue 属性	省略値 (depth cue off) 以外は無視してもよい.
HLHSR_identifier	少なくとも 1 つは利用可能でなければならない.
特定の GSE	無視してもよい.
出力要素	
NURB 曲面	6 次までサポートしなければならない.
NURB 曲線	6 次までサポートしなければならない.
トリム曲線	4 次までサポートしなければならない.
セル配列	セル配列の外周を描くようにシミュレートできなければならない.
特定の GDP	無視してもよい.
ルックアップ・テーブル	
LineBundle	少なくとも 5 エントリ.
MarkerBundle	少なくとも 5 エントリ.
TextBundle	少なくとも 6 エントリ.
InteriorBundle	少なくとも 5 エントリ.
EdgeBundle	少なくとも 5 エントリ.
Pattern	内部様式に *Pattern* がサポートされているならば少なくとも 10 エントリ.
Color	少なくとも 2 エントリ.
TextFont	少なくとも 2 エントリ.
View	少なくとも 6 エントリ.
Light	少なくとも 2 エントリ.
DepthCue	少なくとも 2 エントリ.
ColorApprox	少なくとも 1 エントリ.
その他	
フォント	X11 フォントで文字列要素を表示する必要はない.
	フォント 1 は単一間隔でなければならない.
名前セット	少なくとも 64 ネーム.
サーチ・コンテキスト	少なくとも 1 つのノーマルと 1 つの反転フィルタ・リスト.
ws 表示優先度	少なくとも 2 つの ws 優先度.

C PEX エラーの定義

PEXGetExtensionInfo

none

PEXGetEnumeratedTypeInfo

Drawable: 指定されたドローアブル ID が正しくない.
Value: 指定された列挙型の番号が正しくない.
Match: 指定されたドローアブルがサポートされていない. *c2

PEXGetInpDepConstants

Value: 指定された定数名が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Drawable: 指定されたドローアブル ID が正しくない.
Match: 指定されたドローアブルがサポートされていない. *c2

PEXMatchRendererTargets *c1

Drawable: 指定されたドローアブル ID が正しくない.
Value: 指定された visual が正しくない.

PEXEscape *c1

Value: 指定された定数名が正しくない.

PEXCreateLookupTable

IDChoice: ID がすでに使われているか，クライアントに割り当てられた範囲にない.
Drawable: 指定されたドローアブル ID が正しくない.
Match: 指定されたドローアブルがサポートされていない. *c2
Value: table_type の値が正しいテーブル型を指していない.
Alloc: サーバがリクエストされたリソースのアロケートに失敗した.
LookupTable: テーブル型がインプリメントされていない.

PEXCopyLookupTable

LookupTable: src_lut_id か dest_lut_id がリソース ID として正しくない.
LookupTable: テーブル型がインプリメントされていない.
Match: src_lut_id と dest_lut_id ガ同じクラスのドローアブル上に作られ，また同じ型のルックアップ・テーブルである.

PEXFreeLookupTable

LookupTable: lut_id が正しくないリソース ID を含む.

PEXGetTableInfo

Drawable: 指定されたドローアブル ID が正しくない.
Match: 指定されたドローアブルがサポートされていない. *c2
Value: table_type 値が正しいテーブル型を指していない.

PEXGetPredefinedEntries

Drawable: 指定されたドローアブル ID が正しくない.
Match: 指定されたドローアブルがサポートされていない. *c2
Value: table_type 値が正しいテーブル型を指していない.
Value: start $<$ 事前設定された最小のエントリ.
Value: start + count $>$ 事前設定された最大のエントリ.
Value: エントリ 0 がこのテーブル型に正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
LookupTable: テーブル型がインプリメントされていない.

PEXGetDefinedIndices

LookupTable: lut_id が正しくないリソース ID を含む.
LookupTable: テーブル型がインプリメントされていない.

PEXGetTableEntry

LookupTable: lut_id が正しくないリソース ID を含む.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Value: エントリ 0 がこのテーブル型に正しくない.

PEXGetTableEntries

LookupTable: lut_id が正しくないリソース ID を含む.
LookupTable: テーブル型がインプリメントされていない.
Value: start + count が 65535 より大きい.
Value: エントリ 0 がこのテーブル型に正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.

PEXSetTableEntries

LookupTable: lut_id が正しくないリソース ID を含む.
LookupTable: テーブル型がインプリメントされていない.
Value: start + count が 65535 より大きい.
Value: テーブル・エントリ中のあるフィールドの値が正しくない.
Value: エントリ 0 がこのテーブル型に正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
ColorType: 装置が指定された色型をサポートしていない.
ColorType: 色型が色ルックアップ・テーブルに対して *Indexed* されていないかもしれない.
Alloc: テーブルがフルである.

PEXDeleteTableEntries

LookupTable: lut_id が正しくないリソース ID を含む.
LookupTable: テーブル型がインプリメントされていない.
Value: start + count が 65535 より大きい.
Value: エントリ 0 がこのテーブル型に正しくない.

PEXCreatePipelineContext

IDChoice: ID がすでに使われているか, クライアントに割り当てられた範囲にない.
Value: item_list 中の項目が範囲外である.
Value: 項目マスク・パラメータ中のビット設定が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
ColorType: 装置が指定された色型をサポートしていない.
Alloc: サーバがリクエストされたリソースのアロケートに失敗した.
NameSet: 指定された名前セット・リソース ID が正しくない. *p

PEXCopyPipelineContext

PipelineContext: pc_ic が正しくないパイプライン・コンテキスト ID を持つ.
Value: 項目マスク・パラメータ中のビット設定が正しくない.

PEXFreePipelineContext

PipelineContext: pc_ic が正しくないパイプライン・コンテキスト ID を持つ.

PEXGetPipelineContext

PipelineContext: pc_ic が正しくないパイプライン・コンテキスト ID を持つ.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Value: 項目マスク・パラメータ中のビット設定が正しくない.

PEXChangePipelineContext

PipelineContext: pc_ic が正しくないパイプライン・コンテキスト ID を持つ.

Value: item_list 中の項目が範囲外である.
Value: 項目マスク・パラメータ中のビット設定が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
ColorType: 装置が指定された色型をサポートしていない.
NameSet: 指定された名前セット・リソース ID が正しくない. *p

PEXCreateRenderer

IDChoice: ID がすでに使われているか, クライアントに割り当てられた範囲にない.
Drawable: 指定されたドローアブル ID が正しくない.
PipelineContext: pc_ic が正しくないパイプライン・コンテキスト ID を持つ.
NameSet: 指定された名前セット ID が正しくない.
LookupTable: 指定されたルックアップ・テーブル ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Value: item_list 中の項目が範囲外である.
Value: 項目マスク・パラメータ中のビット設定が正しくない.
Alloc: サーバがリクエストされたリソースのアロケートに失敗した.
Match: ルックアップ・テーブルの root/depth が例に示されたドローアブルのと一致しない.
Match: 指定されたドローアブルがサポートされていない. *c1

PEXFreeRenderer

Renderer: 指定されたレンダラ ID が正しくない.

PEXChangeRenderer

Renderer: 指定されたレンダラ ID が正しくない.
Match: 指定されたルックアップ・テーブルが指定されたレンダラと同じ root と depth を持つドローアブルとして作られていない.
Value: item_list 中の項目が範囲外である.
Value: 項目マスク・パラメータ中のビット設定が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
NameSet: 指定された名前セット ID が正しくない.
LookupTable: 指定されたルックアップ・テーブル ID が正しくない.
PipelineContext: 指定されたパイプライン・コンテキストの ID が正しくない.

PEXGetRendererAttributes

Renderer: 指定されたレンダラ ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Value: 項目マスク・パラメータ中のビット設定が正しくない.

PEXGetRendererDynamics

Renderer: 指定されたレンダラ ID が正しくない.

PEXBeginRendering

Renderer: 指定されたレンダラ ID が正しくない.
Drawable: 指定されたドローアブル ID が正しくない.
Match: 指定されたルックアップ・テーブルが指定されたレンダラと同じ root と depth を持つドローアブルとして作られていない. *c1
Match: 指定されたドローアブルがサポートされていない. *c1
Alloc: サーバがレンダリングをおこなうのに必要なリソースをアロケートできなかった.
RendererState: レンダラがすでに *Rendering* 状態か *Picking* 状態である. *c2

PEXEndRendering

Renderer: 指定されたレンダラ ID が正しくない.
RendererState: レンダラが *Picking* 状態にある. *c2

PEXBeginStructure

Renderer: 指定されたレンダラ ID が正しくない.

PEXEndStructure

Renderer: 指定されたレンダラ ID が正しくない.
RendererState: begin structure にマッチしない.

PEXRenderOutputCommands

Renderer: 指定されたレンダラ ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
OutputCommand: 出力コマンドの中に正しくない値がある.

PEXRenderElements *c1

Renderer: 指定されたレンダラ ID が正しくない.
Structure: 指定されたストラクチャ ID が正しくない.
Value: "whence" パラメータの値が正しくない.

PEXAccumulateState *c1

Renderer: 指定されたレンダラ ID が正しくない.
Path: サーチ・パスが正しくない（正しくないストラクチャ ID を含んでいるか，要素のオフセット値が正しくない）.

PEXRenderNetwork

Renderer: 指定されたレンダラ ID が正しくない.
Drawable: 指定されたドローアブル ID が正しくない.
Structure: 指定されたストラクチャ ID が正しくない.
Match: 指定されたレンダラ・リソースが指定されたレンダラと同じ root と depth を持つドローアブルとして作られていない. *c1
Match: 指定されたドローアブルがサポートされていない. *c1
Alloc: サーバがレンダリングをおこなうのに必要なリソースをアロケートできなかった. *c1
RendererState: レンダラがすでに *Rendering* 状態か *Picking* 状態である. *c2

PEXCreateStructure

IDChoice: ID がすでに使われているか，クライアントに割り当てられた範囲にない.
Alloc: サーバがリクエストされたリソースのアロケートに失敗した.

PEXCopyStructure

Structure: 指定されたストラクチャ ID が正しくない.

PEXDestroyStructure

Structure: 指定されたストラクチャ ID が正しくない.

PEXGetStructureInfo

Structure: 指定されたストラクチャ ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Value: 項目マスク・パラメータ中のビット設定が正しくない.

PEXGetElementInfo

Structure: 指定されたストラクチャ ID が正しくない.
Value: "whence" パラメータの値が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.

PEXGetStructureInNetwork

Structure: 指定されたストラクチャ ID が正しくない.
Value: "whence" パラメータの値が正しくない.

PEXGetAncestors

Structure: 指定されたストラクチャ ID が正しくない.
Value: "path_part" パラメータの値が正しくない.

PEXGetDescendants

Structure: 指定されたストラクチャ ID が正しくない.
Value: "path_part" パラメータの値が正しくない.

PEXFetchElements

Structure: 指定されたストラクチャ ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Value: "whence" パラメータの値が正しくない.

PEXSetEditingMode

Structure: 指定されたストラクチャ ID が正しくない.
Value: "mode" パラメータの値が正しくない.

PEXSetElementPointer

Structure: 指定されたストラクチャ ID が正しくない.
Value: "whence" パラメータの値が正しくない.

PEXSetElementPointerAtLabel

Structure: 指定されたストラクチャ ID が正しくない.
Label: ストラクチャ内に指定されたラベルが見つからない.

PEXElementSearch

Structure: 指定されたストラクチャ ID が正しくない.
Value: "whence" または "direction" パラメータが正しくない.

PEXStoreElements

Structure: 指定されたストラクチャ ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
OutputCommand: 出力コマンドの中に正しくない値がある.

PEXDeleteElements

Structure: 指定されたストラクチャ ID が正しくない.
Value: "whence" パラメータの値が正しくない.

PEXDeleteElementsToLabel

Structure: 指定されたストラクチャ ID が正しくない.
Label: ストラクチャ内に指定されたラベルが見つからない.
Value: "whence" パラメータの値が正しくない.

PEXDeleteElementsBetweenLabels

Structure: 指定されたストラクチャ ID が正しくない.
Label: ストラクチャ内に指定されたラベルが見つからない.

PEXCopyElements

Structure: 指定されたストラクチャ ID が正しくない.
Value: "whence" パラメータの値が正しくない.

PEXChangeStructureReferences

Structure: 指定されたストラクチャ ID が正しくない.

PEXCreateNameSet

IDChoice: ID がすでに使われているか, クライアントに割り当てられた範囲にない.
Alloc: サーバがリクエストされたリソースのアロケートに失敗した.

PEXCopyNameSet

NameSet: 指定された名前セット ID が正しくない.

PEXFreeNameSet

NameSet: 指定された名前セット ID が正しくない.

PEXGetNameSet

NameSet: 指定された名前セット ID が正しくない.

PEXChangeNameSet

NameSet: 指定された名前セット ID が正しくない.
Value: "action" パラメータの値が正しくない.

PEXCreateSearchContext

IDChoice: ID がすでに使われているか, クライアントに割り当てられた範囲にない.
Value: item_list 中の項目が範囲外である.
Value: 項目マスク・パラメータ中のビット設定が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Alloc: サーバがリクエストされたリソースのアロケートに失敗した.
Path: サーチ・パスが正しくない (正しくないストラクチャ ID を含んでいるか, 要素のオフセット値が正しくない).
NameSet: 指定された名前セット ID が正しくない.

PEXCopySearchContext

SearchContext: 指定されたサーチ・コンテキスト ID が正しくない.
Value: 項目マスク・パラメータ中のビット設定が正しくない.

PEXFreeSearchContext

SearchContext: 指定されたサーチ・コンテキスト ID が正しくない.

PEXGetSearchContext

SearchContext: 指定されたサーチ・コンテキスト ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Value: 項目マスク・パラメータ中のビット設定が正しくない.

PEXChangeSearchContext

SearchContext: 指定されたサーチ・コンテキスト ID が正しくない.
Value: item_list 中の項目が範囲外である.
Value: 項目マスク・パラメータ中のビット設定が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Path: サーチ・パスが正しくない (正しくないストラクチャ ID を含んでいるか, 要素のオフセット値が正しくない).
NameSet: 指定された名前セット ID が正しくない.

PEXSearchNetwork

SearchContext: 指定されたサーチ・コンテキスト ID が正しくない.
Path: サーチ・パスが正しくない (正しくないストラクチャ ID を含んでいるか, 要素のオフセット値が正しくない).

PEXCreatePhigsWKS

IDChoice: ID がすでに使われているか, クライアントに割り当てられた範囲にない.
Drawable: 指定されたドローアブル ID が正しくない.
Match: 指定されたルックアップ・テーブルが指定されたレンダラと同じ root と depth を持つドローアブルとして作られていない.
Match: 指定されたドローアブルがサポートされていない. *c1
LookupTable: 指定されたルックアップ・テーブル ID が正しくない.
NameSet: 指定された名前セット ID が正しくない.
Alloc: サーバがリクエストされたリソースのアロケートに失敗した.

Alloc: サーバが double-buffering に必要なリソースをアロケートできなかった.
Value: "buffer_mode" パラメータの値が正しくない.

PEXFreePhigsWKS
PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.

PEXGetWKSInfo
PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Value: 項目マスク・パラメータ中のビット設定が正しくない.

PEXGetDynamics
Drawable: 指定されたドローアブル ID が正しくない.
Match: 指定されたドローアブルがサポートされていない. *c1

PEXGetViewRep
PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Value: 指定されたビュー・テーブルのエントリが定義されていない.

PEXRedrawAllStructure
PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.

PEXUpdateWorkstation
PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.

PEXRedrawClipRegion
PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.

PEXExecuteDeferredActions
PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.

PEXSetViewPriority
PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
Value: "priority" パラメータの値が正しくない.
Value: 指定されたテーブルのエントリが定義されていない.

PEXSetDisplayUpdateMode
PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
Value: "display_update" パラメータの値が正しくない.

PEXMapWCtoDC
PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.

PEXMapDCtoWC
PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.

PEXSetViewRep
PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Alloc: テーブルがフルである.

PEXSetWKSWindow
PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.

PEXSetWKSViewport

PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Value: "use_drawable" パラメータの値が正しくない.

PEXSetHLHSRMode

PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
Value: "mode" パラメータの値が正しくない.
Alloc: サーバが必要なリソースをアロケートできなかった. *c1

PEXSetWKSBufferMode

PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
Value: "buffer_mode" パラメータの値が正しくないか, サポートされていない. *c1
Alloc: サーバが double-buffering に必要なリソースをアロケートできなかった.

PEXPostStructure

PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
Structure: 指定されたストラクチャ ID が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.

PEXUnpostStructure

PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
Structure: 指定されたストラクチャ ID が正しくない.

PEXUnpostAllStructures

PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.

PEXGetWKSPostings

Structure: 指定されたストラクチャ ID が正しくない.

PEXGetPickDevice

PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
Value: "dev_type" パラメータの値が正しくない.
Value: 項目マスク・パラメータ中のビット設定が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.

PEXChangePickDevice

PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
Value: "dev_type" パラメータの値が正しくない.
Value: item_list 中の項目が範囲外である.
Value: 項目マスク・パラメータ中のビット設定が正しくない.
FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Path: サーチ・パスが正しくない (正しくないストラクチャ ID を含んでいるか, 要素のオフセット値が正しくない).
NameSet: 指定された名前セット ID が正しくない.

PEXCreatePickMeasure

IDChoice: ID がすでに使われているか, クライアントに割り当てられた範囲にない.
PhigsWKS: 指定された PHIGS ワークステーション ID が正しくない.
Alloc: サーバがリクエストされたリソースのアロケートに失敗した.
Value: "dev_type" パラメータの値が正しくない.

PEXFreePickMeasure

PickMeasure: 指定されたピックメジャ ID が正しくない.

PEXGetPickMeasure

PickMeasure: 指定されたピックメジャ ID が正しくない.
Value: 項目マスク・パラメータ中のビット設定が正しくない.

PEXUpdatePickMeasure

PickMeasure: 指定されたピックメジャ ID が正しくない.
Path: サーチ・パスが正しくない（正しくないストラクチャ ID を含んでいるか，要素のオフセット値が正しくない）.

PEXBeginPickOne *c1

Renderer: 指定されたレンダラ ID が正しくない.
Drawable: 指定されたドローアブル ID が正しくない.
Match: 指定されたレンダラ・リソースが指定されたレンダラと同じ root と depth を持つドローアブルとして作られていない.
Match: 指定されたドローアブルがサポートされていない.
Alloc: サーバがレンダリングをおこなうのに必要なリソースをアロケートできなかった.
RendererState: レンダラがすでに *Rendering* 状態か *Picking* 状態である. *c2
Value: "pick_type" か "pick_data" の値が正しくない.

PEXEndPickOne *c1

Renderer: 指定されたレンダラ ID が正しくない.
RendererState: レンダラがすでに *Rendering* 状態である.

PEXPickOne *c1

Renderer: 指定されたレンダラ ID が正しくない.
Drawable: 指定されたドローアブル ID が正しくない.
Structure: 指定されたストラクチャ ID が正しくない.
Match: 指定されたレンダラ・リソースが指定されたレンダラと同じ root と depth を持つドローアブルとして作られていない.
Match: 指定されたドローアブルがサポートされていない.
Alloc: サーバがレンダリングをおこなうのに必要なリソースをアロケートできなかった.
RendererState: レンダラがすでに *Rendering* 状態か *Picking* 状態である. *c2
Value: "pick_type" か "pick_data" の値が正しくない.

PEXBeginPickAll *c1

Renderer: 指定されたレンダラ ID が正しくない.
Drawable: 指定されたドローアブル ID が正しくない.
Match: 指定されたレンダラ・リソースが指定されたレンダラと同じ root と depth を持つドローアブルとして作られていない.
Match: 指定されたドローアブルがサポートされていない.
Alloc: サーバがレンダリングをおこなうのに必要なリソースをアロケートできなかった.
RendererState: レンダラがすでに *Rendering* 状態か *Picking* 状態である. *c2
Value: "pick_data" の値が正しくない.

PEXEndPickAll *c1

Renderer: 指定されたレンダラ ID が正しくない.
RendererState: レンダラがすでに *Rendering* 状態である.

PEXPickAll *c1

Renderer: 指定されたレンダラ ID が正しくない.
Drawable: 指定されたドローアブル ID が正しくない.
Structure: 指定されたストラクチャ ID が正しくない.
Match: 指定されたレンダラ・リソースが指定されたレンダラと同じ root と depth を持つドローアブルとして作られていない.
Match: 指定されたドローアブルがサポートされていない.

Alloc: サーバがレンダリングをおこなうのに必要なリソースをアロケートできなかった.
RendererState: レンダラがすでに *Rendering* 状態か *Picking* 状態である. *c2
Value: "pick_data" の値が正しくない.
Path: 初期のピックパスが正しくない.

PEXOpenFont

PEXFont: 文字列が使用可能な PEX フォントを指していない.
IDChoice: ID がすでに使われているか, クライアントに割り当てられた範囲にない.
Alloc: サーバがリクエストされたリソースのアロケートに失敗した.

PEXCloseFont

PEXFont: 指定された PEX フォント ID が正しくない.

PEXQueryFont

PEXFont: 指定された PEX フォント ID が正しくない.

PEXListFonts

none

PEXListFontsWithInfo

none

PEXQueryTextExtents

FloatingPointFormat: 装置が指定された fp 形式をサポートしていない.
Value: 入力された文字列属性が正しくないか, 範囲外である.
Value: リソース ID が正しいレンダラ, PHIGS ワークステーション, あるいはルックアップ・テーブルを指していない.
Match: リソース ID が TextFont 以外の型のテーブルを指している.

D テーブルの省略値の定義

もし定義されていないテーブル・エントリが参照されたときは省略値のテーブル・エントリの内容が使用される．すべてのテーブルに対する省略値エントリは１である．ただし省略値エントリが０であるビュー，デプス・キュー，色近似テーブルを除く．もし省略値のテーブル・エントリの内容が未定義ならば，以下の掲げる省略値の属性値が代わりに使用される．テーブルの各型について，属性名，データ型，省略値を表にしている． PEX の各インプリメンテーションでは，「インプリメンテーション依存」としてある属性について，その選択値を記述したドキュメントを供するべきである．

属性名	データ型	省略値
LineBundle		
line_type	LINE_TYPE	*LineTypeSolid*
polyline_interp	POLYLINE_INTERP	*PolylineInterpNone*
curve_approx	CURVE_APPROX	{1, 1.0}[1]
line_width	FLOAT	1.0
line_color	COLOR_SPECIFIER	{*Indexed*,1}
MarkerBundle		
marker_type	MARKER_TYPE	*MarkerAsterisk*
marker_scale	FLOAT	1.0
marker_color	COLOR_SPECIFIER	{*indexed*,1}
TextBundle		
text_font_index	TABLE_INDEX	1
text_precision	TEXT_PRECISION	*String*
char_expansion	FLOAT	1.0
char_spacing	FLOAT	0.0
text_color	COLOR_SPECIFIER	{*Indexed*,1}
InteriorBundle		
interior_style	INTERIOR_STYLE	*InteriorStyleHollow*
interior_style_index	TYPE_OT_TABLE_INDEX	1
surface_color	COLOR_SPECIFIER	{*Indexed*,1}
reflection_attr	REFLECTION_ATTR	{1.0,1.0,1.0,0.0,0.0,{*Indexed*,1}}
reflection_model	REFLECTION_MODEL	*ReflectionNoShading*
surface_interp	SURFACE_INTERP	*SurfaceInterpNone*
bf_interior_style	INTERIOR_STYLE	*InteriorStyleHollow*
bf_interior_style_index	TYPE_OR_TABLE_INDEX	1
bf_surface_color	COLOR_SPECIFIER	{*Indexed*,1}
bf_reflection_attr	REFLECTION_ATTR	{1.0,1.0,1.0,0.0,0.0,{*Indexed*,1}}
bf_reflection_model	REFLECTION_MODEL	*ReflectionNoShading*
bf_surface_interp	SURFACE_INTERP	*SurfaceInterpNone*
surface_approx	SURFACE_APPROX	{1,1.0,1.0}

[1] PHIGS-PLUS はデフォルトの曲線近似型を１に定義しているが，これはインプリメンテーション依存方法である．

属性名	データ型	省略値
EdgeBundle		
surface_edges	SWITCH	*Off*
surface_edges_type	SURFACE_EDGE_TYPE	*SurfaceEdgeSolie*
surface_edges_width	FLOAT	1.0
surface_edges_color	COLOR_SPECIFIER	{*Indexed*,1}
Pattern		
color_type	COLOR_TYPE	imp.dep.
numx	CARD16	imp.dep.
numy	CARD16	imp.dep.
colors	LISTofCOLOR	imp.dep.
Color		
color_type	COLOR_TYPE	imp.dep.
color	DIRECT_COLOR	imp.dep.
TextFont		
font	LISTofFONT_ID	imp.dep.
View		
clip_flags	BITMASK_SHORT	すべて *On*
clip_limits	NPC_SUBVOLUME	(0,0,0),(1,1,1)
orientation	MATRIX	単位行列
mapping	MATRIX	単位行列
Light[2]		
light_type	LIGHT_TYPE	imp.dep.
direction	VECTOR_3D	imp.dep.
point	COORD_3D	imp.dep.
concentration	FLOAT	imp.dep.
spread_angle	FLOAT	imp.dep.
attenuation	[factor1, factor2: FLOAT]	imp.dep.
color	COLOR_SPECIFIER	imp.dep.
DepthCue		
mode	SWITCH	*Off*
front_plane	FLOAT	imp.dep.
back_place	FLOAT	imp.dep.
front_scaling	FLOAT	imp.dep.
back_scaling	FLOAT	imp.dep.
color	COLOR_SPECIFIER	imp.dep.

[2]光源の型により，テーブル・エントリ中のいくつかの値は無視されるかもしれない．光源テーブル中の未定義エントリは，*Off* に設定された光源とみなす．

属性名	データ型	省略値
ColorApprox		
type	COLOR_APPROX_TYPE	imp.dep.
color_model	COLOR_APPROX_MODEL	imp.dep.
max1	CARD16	imp.dep.
max2	CARD16	imp.dep.
max3	CARD16	imp.dep.
mult1	CARD32	imp.dep.
mult2	CARD32	imp.dep.
mult3	CARD32	imp.dep.
weight1	FLOAT	imp.dep.
weight2	FLOAT	imp.dep.
weight3	FLOAT	imp.dep.
base_pixel	CARD32	imp.dep.
dither	SWITCH	imp.dep.

装置に対する X の RGB_DEFAULT_MAP の定義に関連させる形で，PEX のインプリメンテーションが省略値を設定することを提案する．たとえば装置が 8 ビット画素を持ち，24 ビットのハードウェア色マップを指標しているとする．また RGB_DEFAULT_MAP は概念的にセル番号 16 から始まる 216 の色マップ・エントリに符号化された，6 × 6 × 6 の色立方体とする．例はこの場合の提案値である．

属性名	データ型	省略値
ColorApprox の例		
type	COLOR_APPROX_TYPE	*ColorSpace*
color_model	COLOR_APPROX_MODEL	*RGB*
max1	CARD16	5
max2	CARD16	5
max3	CARD16	5
mult1	CARD32	1
mult2	CARD32	6
mult3	CARD32	36
weight1	FLOAT	1.0
weight2	FLOAT	1.0
weight3	FLOAT	1.0
base_pixel	CARD32	16
dither	SWITCH	*Off*

E 登録されている PEX エスケープ

*c2

PEX のインプリメンテーションでは，登録された PEX エスケープのサポートは必要としない．特定のインプリメンテーションがサポートしているエスケープのリストは，**PEXGetEnumeratedTypeInfo** を呼ぶことにより問い合わせられる．すべての登録エスケープは一つずつの *escape_id* を持っている．これはエスケープ・リクエストを特定するための正の数である．

E.1 エコー色の設定

名前:

PEXSetEchoColor

リクエスト:

escape_id: CARD32
fp_format: FLOAT_FORMAT
rdr_id: RENDERER_ID
echo_color: COLOR_SPECIFIER

エラー:

Renderer, ColorType, FloatingPointFormat

説明:

このリクエストは，レンダラの *echo_color* 属性を設定する．更新はダイナミックであり，即座に効果がある．エコー色はレンダラの *echo_mode* が *Echo* に設定されているときに使用される．エコーされた要素についてのライティングとシェーディングはインプリメンテーション依存である．

echo_color はこのリクエストによって導入された新しいレンダラ属性であり，その概念はこのエスケープをサポートしないようなインプリメンテーションではサポートされないだろう． *echo_color* 属性は要求されないし，その省略値はインプリメンテーション依存である．

もし指定されたレンダラ記述子が正しくないときは， *Renderer* エラーが発生する．もし指定された（ *echo_color* の中の）色型がサポートされていないときは， *ColorType* エラーが生じる．もし指定された浮動小数形式がサポートされていないときは， *FloatingPointFormat* エラーが生じる．サポートされている色型と浮動小数形式は， **PEXGetEnumeratedTypeInfo** を呼ぶことによって問い合わせできる．

F　和訳用語対照表

dots and dashes line ..... 一点鎖線
abstraction ..... 抽象
accumulate state traversal ..... 累積状態探訪
accumulate ..... 累積
additional information ..... 付加情報
ambient coefficient ..... 環境係数
ambient light source ..... 環境光源
ambient light ..... 環境光源
ambient term ..... 環境項
ancestor ..... 親階層
annotation point ..... 注釈位置
annotation text ..... 注釈文字列
aspect source flag ..... ASF
aspect source flag ..... ASF
attenuation coefficient ..... 減衰係数
attribute ..... 属性
back depth cueing plane 後方デプス・キューイング面
backfacing surface primitive ..... 裏面領域要素
byte-swapping ..... バイト交換
ceiling ..... 限界
cell array primitive ..... セル配列要素
character expansion factor ..... 文字拡大因子
character expansion ..... 拡大率
character height ..... 文字高
character spacing ..... 文字間隔
character up vector ..... 文字上方向ベクトル
client ..... クライアント
client-side traversal ..... クライアント側探訪
clip ..... クリップ
close ..... クローズ
color approximation model ..... 色近似モデル
color approximation table ..... 色近似テーブル
color approximation type ..... 色近似型
color lookup table ..... 色ルックアップ・テーブル
color space ..... 色空間
color type ..... 色型
color value ..... 色値
color ..... 色
colormap ..... 色マップ
common type ..... 共通型
concentration exponent ..... 集中指数
cone of influence ..... 放射円錐
connection line ..... 引き出し線
connection ..... コネクション
contour ..... 外周
control point ..... 制御点
convex ..... 凸型多角形
culling mode ..... 識別モード
curve approximation method ..... 曲線近似法
curve approximation ..... 曲線近似
curve ..... 曲線
dashed line ..... 破線
data field ..... データ・フィールド
degenerate ..... 縮退
depth cue table ..... デプス・キュー・テーブル
depth ..... 深さ
depth-cueing ..... デプス・キューイング
descendant ..... 子階層
destination drawable ..... 出力ドローアブル
destroy ..... 破壊
device coordinates ..... 装置座標系
diffuse coefficient ..... 拡散係数
diffuse reflection ..... 拡散反射
diffuse term ..... 拡散項
direct value ..... 直値
direction vector ..... 方向ベクトル
directional light source ..... 平行光源
disable list ..... 無効リスト
display update mode ..... 表示更新モード
dithering hint ..... ディザリング・ヒント
dithering ..... ディザ
dithering ..... ディザリング
dot product ..... 内積項
dotted line ..... 点線
double buffering ..... ダブル・バッファリング
double-buffering ..... ダブル・バッファリング
drawable ..... ドローアブル
dynamic modification ..... 動的変更
dynamics ..... ダイナミックス
echo color ..... エコー色
echo type ..... エコー型
edge switch ..... 外周線スイッチ
edge ..... 外周線
editing ..... 編集
element type ..... 要素型
empty ..... 空
enable list ..... 有効リスト
entity ..... 実体
enumerated type ..... 列挙型
error code ..... エラーコード
error ..... エラー
escape ..... エスケープ
escape-specific ..... エスケープ定義
event code ..... 事象コード
event ..... 事象
expose ..... 露出
facet ..... 小面
fast rendering code path ..... 高速処理コード
fetch ..... フェッチ
fill area set ..... 多角形集合要素
fill ..... フィル
filter output command ..... フィルタ出力コマンド

filtering ..... フィルタリング
flat meaning ..... 単調
floating-point format ..... 浮動小数形式
font file ..... フォント・ファイル
font group ..... フォント・グループ
format word ..... フォーマット・ワード
front clipping plane ..... 前方クリップ面
front depth cueing plane . 前方デプス・キューイング面
geometric primitive ..... 幾何要素
glyph ..... グリフ
halfspace ..... 半空間
hatch style ..... ハッチ様式
hatch ..... ハッチ
head ..... 先頭
header ..... ヘッダ
hierarchical path ..... 階層パス
highlight exclusion set ..... 顕示除外集合
highlight inclusion set ..... 顕示包含集合
highlight ..... 顕示
hit box ..... ヒット・ボックス
hit test ..... ヒット・テスト
HLHSR mode ..... HLHSR モード
homogeneous modeling coordinates . 同次モデリング座標系
hue ..... 色相
illumination ..... 照明
immediate rendering ..... 即時レンダリング
immediate ..... 即時
index color ..... 指標色
index value ..... 指標値
indirect color reference ..... 間接色値参照
instance ..... インスタンス
interior style index ..... 内部様式指標
interior style ..... 内部様式
inverse ..... 逆行列
invert ..... 反転
inverted filter list ..... 逆フィルタ・リスト
invisibility exclusion set ..... 不可視除外集合
invisibility inclusion set ..... 不可視包含集合
isoparametric curve ..... 等パラメトリック曲線
isotropic mapping ..... 等方マッピング
knot ..... 節点
later excution ..... 遅延実行
leaf node ..... 末節
length field ..... データ長フィールド
level curve ..... レベル曲線
light source description ..... 光源情報
light source ..... 光源
light table ..... 光源表
light type ..... 光源型
lighting equation ..... 照明方程式
lightness ..... 明度
line type ..... 線種型
linearly interpolating ..... 線形補間
local model transformation matrix ローカル・モデル変換行列
lookup table ..... ルックアップ・テーブル
luminance ..... 輝度
major opcode ..... 主要オペコード
marker primitive ..... マーカ要素
marker size ..... マーカ寸法
marker type ..... マーカ型
minor opcode ..... 副オペコード
model clip indicator ..... モデル・クリップ識別子
model clip method ..... モデル・クリップ法
modeling clipping ..... モデリング・クリッピング
modeling transformation ..... モデリング変換
modeling ..... モデリング
modification ..... 変更
move ..... 移動
name set ..... 名前セット
non-homogenous modeling coordinates 非同次モデリング座標
normal ..... 法線
normalized project coordinates ... 正規化投影座標系
normalized projection coordinates 正規化投影座標系
normal filter list ..... 標準フィルタ・リスト
NURBS ..... パラメトリック
occlude ..... 隠蔽
opaque ..... 不透明
opcode ..... オペコード
open ..... オープン
origin ..... 基点
output command ..... 出力コマンド
packet ..... パケット
padding ..... パディング
parallel projection ..... 平行投影
parallelpipe ..... 平行筒
parametric surface characteristics . パラメトリック曲面特性
path offset ..... パス・オフセット
pattern reference point ..... パターン参照位置
pattern reference vector ..... パターン参照ベクトル
pattern table ..... パターン・テーブル
pattern vector ..... パターン・ベクトル
perspective projection ..... 透視投影
PEX font ..... PEX フォント
PHIGS compatibility ..... PHIGS 互換
PHIGS workstation .... PHIGS ワークステーション
pick all method ..... 全ピック法
pick all ..... 全ピック

pick aperature ..... ピック口径
pick device type ..... ピック装置タイプ
pick device ..... ピック装置
pick distance ..... ピック距離
pick exclusion filter ..... ピック除外フィルタ
pick inclusion filter ..... ピック包含フィルタ
pick measure ..... ピックメジャ
pick one method ..... 単一ピック法
pick one ..... 単一ピック
pick position ..... ピック位置
pick traversal ..... 探訪
picking operation ..... ピッキング操作
pipeline context ..... パイプライン・コンテキスト
polygon with holes ..... 穴空き多角形
polygon with islands ..... 内包多角形
polyline interpolation method 折線シェーディング法
polyline ..... 折線
positional light source ..... 点光源
post ..... ポスト
post-processing ..... 後処理
prompt echo type ..... プロンプト・エコー型
prompt ..... プロンプト
properties ..... プロパティ
pseudo ..... 擬似
quadrilateral mesh primitive ....四辺形メッシュ要素
reflection model ..... 反射モデル
reflection value ..... 反射値
regeneration ..... 再生成
renderer picking ..... レンダラ・ピッキング
renderer ..... レンダラ
rendering color model ..... レンダリング色モデル
rendering pipeline ..... レンダリング・パイプライン
Rendering ..... レンダリング
reply ..... リプライ
request ..... リクエスト
resize ..... リサイズ
resource ..... リソース
restore ..... 復元
root window ..... ルート・ウィンドウ
root ..... ルート
sampling ..... サンプリング
saturation ..... 彩度
save ..... 退避
scan line ..... スキャンライン
search context ..... サーチ・コンテキスト
search filter criteria ..... サーチ・フィルタ条件
searching operation ..... サーチ操作
segment ..... セグメント
sequence number field ..... 連続番号フィールド
server ..... サーバ
shading ..... シェーディング
simulate ..... シミュレート
solid ..... 実線
spatial search ..... 空間探索
specular coefficient ..... 鏡面係数
specular color ..... 鏡面色
specular concentration ..... 鏡面集中指数
specular reflection ..... 鏡面反射
specular term ..... 鏡面項
spread angle ..... 広がり角
stroke font ..... ストローク・フォント
structure component ..... 構成要素
structure element ..... ストラクチャ要素
structure invocation ..... ストラクチャ呼び出し
structure network path ...ストラクチャ・ネットワーク・パス
structure rendering .... ストラクチャ・レンダリング
structure resource ..... ストラクチャ・リソース
structure traversal ..... ストラクチャ探訪
subset ..... サブセット
surface approximation method ..... 曲面近似法
surface approximation ..... 曲面近似法
surface color ..... 領域色
surface edge color ..... 外周線色
surface edge type ..... 外周線型
surface edge width ..... 外周線幅
surface edge ..... 外周線
surface interpolation method .領域シェーディング法
surface primitive ..... 領域要素
surface ..... 領域
switching ..... 切り換え
Syntax ..... 構文
table indices ..... テーブル指標
tail ..... 末尾
text alignment ..... 水平／垂直配置
text font index ..... 文字列フォント指標
text font table ..... 文字列フォント・テーブル
text path ..... 文字列方向
text precision ..... 文字列表示精度属性
text ..... 文字列
tolerance ..... 許容誤差
track of paths ..... パス経路
transmission coefficient ..... 透過係数
triangle strip primitive ..... 三角形ストリップ要素
trim curve approximation method トリム曲線近似法
trim curve ..... トリム曲線
trimming loop ..... トリミング・ループ
type code ..... 型コード
type field ..... 型フィールド
type mnemonic ..... 型ニューモニック
type ..... 型
up vector ..... 立ち上がり方向ベクトル

value ..... 明度
vertices ..... 頂点
view information ..... 視野情報
view table ..... 視野テーブル
view ..... ビュー
view ..... 視野
viewport ..... ビューポート
visibility flag ..... 可視性フラグ
visibility ..... 可視性
WCS point light source ..... WCS 点光源
WCS spot light source ..... WCS スポット光源
WCS vector light source ..... WCS ベクトル光源
wildcard ..... ワイルドカード
workstation picking ワークステーション・ピッキング
world coordinates ..... 世界座標系
write-proteced ..... 書き込みプロテクト
z-buffering ..... z- バッファ
atext_alignment ..... 注釈文字列配置属性
atext_height ..... 注釈文字列高属性
atext_path ..... 注釈文字列方向属性
atext_style ..... 注釈文字列様式属性
atext_up_vector ..... 注釈文字上方向ベクトル属性
bf_interior_style_index ..... 裏面内部様式指標属性
bf_interior_style ..... 裏面内部スタイル属性
bf_reflection_attr ..... 裏面反射特性属性
bf_reflection_model ..... 裏面反射モデル属性
bf_surface_color ..... 裏面領域色属性
bf_surface_interp ..... 裏面内部シェーディング法属性
char_expansion ..... 文字拡大率属性
char_height ..... 文字高属性
char_spacing ..... 文字間隔属性
color_approx_index ..... 色近似指標属性
culling_mode ..... 面識別モード属性
curve_approx ..... 曲線近似属性
depth_cue_index ..... デプス・キュー指標属性
distinguish ..... 面識別フラグ属性
edge_bundle_index ..... 外周線束指標属性
global_transform ..... グローバル変換行列属性
HLHSR_identifier ..... HLHSR- 識別子属性
interior_bundle_index ..... 内部スタイル束指標属性
interior_style_index ..... 内部様式指標属性
interior_style ..... 内部様式属性
light_state ..... 光源状態属性
line_bundle_index ..... 線束指標属性
line_color ..... 線色属性
line_type ..... 線種属性
line_width ..... 線幅属性
local_transform ..... ローカル変換行列属性
marker_bundle_index ..... マーカ束指標属性
marker_color ..... マーカ色属性
marker_scale ..... マーカ寸法属性
marker_type ..... マーカ種属性
model_clip_volume モデル・クリップ・ボリューム属性
model_clip ..... モデル・クリップ属性
NonRational ..... 非有理
order ..... 次数
pattern_ref_pt ..... パターン参照点属性
pattern_size ..... パターン寸法属性
pick device descriptor ..... ピック・装置記述子
pick measure ..... ピックメジャ
pick_id ..... ピック識別子属性
pipeline context ..... パイプライン・コンテキスト
polyline_interp ..... 折線シェーディング法属性
protocol ..... プロトコル
Rational ..... 有理
rdr_color_model ..... レンダリング色モデル属性
reflection_attr ..... 領域反射特性属性
reflection_model ..... 反射モデル属性
release_number ..... リリース番号
surface_approx ..... 曲面近似属性
surface_color ..... 領域色指標
surface_edges ..... 外周線属性
surface_edge_color ..... 外周線色属性
surface_edge_type ..... 外周線種属性
surface_edge_width ..... 外周線幅属性
surface_interp ..... 内部シェーディング法属性
text_bundle_index ..... 文字列束指標属性
text_color ..... 文字列色属性
text_font_index ..... 文字列フォント指標属性
text_path ..... 文字列方向属性
text_precision ..... 文字列表示精度属性
vender ..... ベンダー
view_index ..... ビュー指標属性

索引

Add names to name set 51
All 29
Alloc 8
all_glyphs_exist 160
Ambient 26,27
Annotation text 2D 54
Annotation text 3D 53
Annotation text alignment 42
Annotation text height 42
Annotation text path 42
Annotation text style 42
Annotation text up vector 42
Application data 52
ASF(aspect source flag) 86
ASF(aspect source flag) 属性 47
asfs 80
ASF_ATTRIBUTE 15
ASF_SPECIFIER 15,86
ASF_VALUE 15
Asterisk 24
atext_alignment 79
atext_height 79
atext_path 79
ATEXT_STYLE 15
atext_style 79
atext_up_vector 79
attenuation 69
Backface Surface color index 45
Backface Surface color 45
Backface Surface interior style index 45
Backface Surface interior style 45
Backface Surface interpolation method 45
Backface Surface reflection attributes 45
Backface Surface reflection model 45
background_color 90
back_place 69
back_scaling 69
base_pixel 70
bf_interior_style 67,79
bf_interior_style_asf 48,81
bf_interior_style_index 67,79
bf_interior_style_index_asf 48,81
bf_reflection_attr 67,79
bf_reflection_attr_asf 48,81
bf_reflection_model 67,79
bf_reflection_model_asf 48,81
bf_surface_color 67,79
bf_surface_color_asf 48,81
bf_surface_interp 67,79
bf_surface_interp_asf 48,81
BITMASK 14,15
BITMASK_SHORT 15,9
BOOLEAN 15
BUFFER_MODE 15
buffer_modify 141
buffer_update 137
CARD16 15
CARD32 15
CARD8 15
Cell array 3D 64
Character expansion 41
Character height 41
Character spacing 41
Character up vector 42
CHARACTER 15
CHARSET_ENCODING 160
CHARSET_REGISTRY 160
char_expansion 79
char_expansion_asf 48,81
char_height 79
char_spacing 79
char_spacing_asf 48,81
char_up_vector 79
CIE 30
CIEFloat 28
CIELUV 14
clear_I 90
clear_Z 90
clip_flags 69
CLIP_INDICATOR 15
clip_limits 69
clip_list 90
ClosestZ 29
Color approximation index 50
COLOR 15
Color 24,26
color 57,59,60,61,62,68,69
ColorRange 30
colors 68
ColorSpace 30
color_approx_index 80
COLOR_APPROX_MODEL 15
color_approx_table 137,141,90
COLOR_APPROX_TYPE 15
COLOR_CIE_FLOAT 28
COLOR_HLS_FLOAT 28
COLOR_HSV_FLOAT 28
COLOR_MODEL 15

color_model ... 70
COLOR_RGB_FLOAT ... 28
COLOR_RGB_INT16 ... 28
COLOR_RGB_INT8 ... 28
COLOR_SPECIFIER ... 15
color_table ... 137,141,90
COLOR_TYPE ... 15
color_type ... 68
COMPOSITION ... 15
concentration ... 69
Connected ... 24
ConstantBetweenKnots ... 24,26,27
CONSTANT_NAME ... 15
CONTOUR ... 15
COORD ... 14,15
COORD_2D ... 15
COORD_3D ... 15,55,63
COORD_4D ... 16,55,63
COORD_TYPE ... 16
Cricle ... 24
Cross ... 24
culling_mode ... 80
CULL_MODE ... 16
current_path ... 90
Curve approximation ... 43
CurveApproxMethod ... 24
CURVE_APPROX ... 16
curve_approx ... 79
curve_approx_asf ... 48,81
CURVE_APPROX_METHOD ... 16
cur_buffer_mode ... 137
cur_HLHSR_mode ... 137
cur_NPC_subvolume ... 137
cur_wks_viewpt ... 137
DashDot ... 24,25
Dashed ... 24,25
DC_ChordalDev ... 25
DC_ChordalSize ... 25,26
DC_HitBox ... 157,29
DC_HIT_BOX_DATA ... 16
DC_PlanarDev ... 27
DC_Relative ... 25,27
DEC_D_Floating ... 28
DEC_F_Floating ... 28
default_glyph ... 160
defined_views ... 137
delete_structure ... 141
Depth cue index ... 50
depth_cue_index ... 80
depth_cue_table ... 137,141,90
DEVICE_COORD ... 16
DEVICE_COORD_2D ... 16
DEVICE_RECT ... 16
Diffuse ... 26
direction ... 69
DIRECT_COLOR ... 16
DISPLAY_STATE ... 16
display_surface ... 137
display_update ... 137
DISPLAY_UPDATE ... 16
distinguish ... 80
dither ... 70
Dot ... 24
DotProduct ... 26
Dotted ... 24,25
drawable_id ... 137
DRAWABLE_ID ... 16
DYNAMIC_TYPE ... 16
EchoNetwork ... 29
EchoPrimitive ... 29
EchoStructure ... 29
ECHO_MODE ... 16
echo_mode ... 90
echo_switch ... 152
echo_volume ... 152
Edge bundle index ... 47
EDGE ... 16
edges ... 59,62
edge_bundle ... 137,141,90
edge_bundle_index ... 80
EDIT_MODE ... 16
ELEMENT_INFO ... 16
ELEMENT_POS ... 16
ELEMENT_RANGE ... 16
ELEMENT_REF ... 16
ELEMENT_TYPE ... 114,14,16
Empty ... 25
ENUM_TYPE ... 16
ENUM_TYPE_INDEX ... 16,23
Escape ... 32
Execute Structure ... 12
Execute structure ... 51
Extended cell array 3D ... 65
EXTENT_INFO ... 16
Facet culling mode ... 46
Facet distinguish flag ... 46
FACET ... 16
Fill area 2D ... 56
Fill area 3D with data ... 56
Fill area 3D ... 55
Fill area set 2D ... 58
Fill area set 3D with data ... 58

Fill area set 3D 57
first_glyph 160
FLOAT 14,16
FLOAT_FORMAT 17,38
font 68
FONT_ID 17
front_plane 69
front_scaling 69
GDP 2D 12,65
GDP 3D 12,65
GDP 14
Global transform 2D 49
Global transform 3D 49
global_transform 80
GSE 12,14,52
HALFSPACE 17
HALFSPACE_2D 17
Hatch 25
HATCH_STYLE 17
HiddenLineOnly 28
highlight_exclusion 137,90
highlight_filter 141
highlight_inclusion 137,90
HLHSR identifier 50
HLHSR モード (HLHSR mode) 28
HLHSR- 識別子属性 (*HLHSR_identifier*) 50
HLHSR_identifier 80
HLHSR_mode 141
HLHSR_MODE 17
HLHSR_mode 90
HLHSR_update 137
HLS 14,30
HLSFloat 28
Hollow 25
HSV 14,30
HSVFloat 28
identity 80
IEEE_754_32 28
IEEE_754_64 28
index 23
Indexed 27
INT16 17
INT32 17
INT8 17
Interior bundle index 46
interior_bundle 137,141,90
interior_bundle_index 80
INTERIOR_STYLE 17
interior_style 67,79
interior_style_asf 48,81
interior_style_index 67,79
interior_style_index_asf 48,81
Intersection 27
inverted_list 128
invisibility_exclusion 137,90
invisibility_filter 141
invisibility_inclusion 137,90
ISO-LATAN1 22
IsoparametricCurves 31
ISTRING 17,41
Label 51
Last 29
last_glyph 160
Light source state 50
light_state 80
light_table 137,141,90
LIGHT_TYPE 17
light_type 69
Line bundle index 44
Line color index 43
Line color 43
Line type 43
Line width 43
line_bundle 137,141,90
line_bundle_index 79
line_color 79
line_color_asf 48,81
LINE_TYPE 17
line_type 79
line_type_asf 48,81
line_width 79
line_width_asf 48,81
LISTofVALUE 14
Local transform 2D 48
Local transform 3D 48
local_transform 80
LOOKUP_TABLE_ID 17
mapping 69
Marker 2D 52
Marker 3D 52
Marker bundle index 40
Marker color index 40
Marker color 40
Marker scale 40
Marker type 40
marker_bundle 137,141,90
marker_bundle_index 79
marker_color 79
marker_color_asf 48,81
marker_scale 79
marker_scale_asf 48,81
MARKER_TYPE 17

marker_type ... 79
marker_type_asf ... 48,81
Match ... 8
MATCH_DRAW_TYPE ... 17,8
MATRIX ... 14,17
MATRIX_3X3 ... 17
max1 ... 70
max2 ... 70
max3 ... 70
MaxHitsEventSupported ... 8
MAX_HITS_REACHED ... 107
MC_LevelCurves ... 31
mnemonic ... 23
mode ... 69
Model clip ... 49
model_clip ... 80
model_clip_flag ... 128
model_clip_volume ... 80
MONO_ENCODING ... 17
mult1 ... 70
mult2 ... 70
mult3 ... 70
name ... 160
NAME ... 17
name_set ... 80
NAME_SET_ID ... 17
NAME_SET_PAIR ... 17
Non-uniform B-spline curve ... 55
Non-uniform B-spline surface ... 63
None ... 24,26,30
Noop ... 65
Normal ... 26
normal ... 57,59,60,61,62
normal_list ... 128
NoShading ... 26
NotConnected ... 24
NPC_ChordalDev ... 25
NPC_ChordalSize ... 25,26
NPC_HitVolume ... 157,29
NPC_PlanarDev ... 27
NPC_Relative ... 25,27
NPC_SUBVOLUME ... 17
NPC_subvolume ... 90
Null ... 62
numx ... 68
numy ... 68
num_priorities ... 137
NURB ... 24
Off ... 28
OPERATOR ... 17
OPT_COLOR ... 17
OPT_DATA ... 17
OPT_NORMAL ... 17
OPT_SWITCH ... 17
orientation ... 69
OUTPUT_CMD ... 17
OUTPUT_COMMAND ... 40
pack_data_red ... 152
Painters ... 28
Parametric surface characteristics ... 51
Pattern reference point and vectors ... 46
Pattern reference point ... 46
Pattern size ... 46
Pattern ... 25
pattern_ref_pt ... 80
pattern_ref_vec1 ... 80
pattern_ref_vec2 ... 80
pattern_size ... 80
pattern_table ... 137,141,90
PC_BITMASK ... 17
PEX フォント (PEX font) ... 158,20
PEX_FONT_ID ... 18
PHIGS ワークステーション (PHIGS workstation) 133,164,22
PHIGS 互換 (PHIGS compatibility) ... 52
PHIGS-ASAP ... 29
PHIGS-ASTI/NIVE ... 30
PHIGS-WAIT/NIVE ... 30
PHIGS-WAIT/UQUM ... 30
PHIGS-WAIT/UWOR ... 29
PHIGS_WKS_ID ... 18
Pick ID ... 50
picked_prim ... 155
PICK_ALL_METHOD ... 18,9
PICK_ALL_STATE ... 18,8
PICK_DATA ... 18,8
PICK_DEVICE_TYPE ... 18
PICK_ELEMENT_REF ... 18
pick_exclusion ... 152,90
pick_id ... 80
pick_inclusion ... 152,90
PICK_MEASURE_ID ... 18
PICK_ONE_METHOD ... 18,9
PICK_ONE_TYPE ... 8
pick_path ... 152
pick_path_order ... 152
PICK_RECORD ... 18,8
pick_start_path ... 90
pick_status ... 152,155
PICK_STATUS ... 18,9
pipeline_context ... 90
PIPELINE_CONTEXT_ID ... 18
PIXMAP_ID ... 18

point ... 69
Polyline 2D ... 54
Polyline 3D ... 54
Polyline interpolation method ... 43
Polyline set 3D with data ... 54
POLYLINE_INTERP ... 18
polyline_interp ... 79
polyline_interp_asf ... 48,81
posted_structs ... 137
post_structure ... 141
prompt_echo_type ... 152
PROMPT_ECHO_TYPE ... 18
properties ... 160
PSC_TYPE ... 18
PSURF_CHAR ... 18
psurf_char ... 80
Quadrilateral mesh ... 60
rdr_color_model ... 80
reference_modify ... 141
REFLECTION_ATTR ... 18
reflection_attr ... 67,79
reflection_attr_asf ... 48,81
REFLECTION_MODEL ... 18
reflection_model ... 67,79
reflection_model_asf ... 48,81
Remove names from name set ... 51
RendererState ... 8
RENDERER_ID ... 18
RENDERER_STATE ... 18
renderer_state ... 90
RENDERER_TARGET ... 18,8
Rendering color model ... 50
Replace ... 27
req_buffer_mode ... 137
req_HLHSR_mode ... 137
req_NPC_subvolume ... 137
req_wks_viewpt ... 137
RESOURCE_ID ... 18
Restore model clip volume ... 50
RGB ... 14,30
RGBFloat ... 28
RGBInt16 ... 28
RGBInt8 ... 28
Scanline ... 28
search_ceiling ... 128
SEARCH_CONTEXT_ID ... 18
search_dist ... 128
search_pos ... 128
Set individual ASF ... 47
Set model clip volume 2D ... 49
Set model clip volume 3D ... 49
Set of fill area sets ... 61
SetEchoColor ... 32
SHAPE ... 18
SimulateSome ... 30
Solid ... 24,25
Specular ... 26
spread_angle ... 69
start_path ... 128
STRING ... 18,23
stroke_font ... 160
STRUCTURE_ID ... 18
STRUCTURE_INFO ... 18
structure_modify ... 141
Surface approximation ... 46
Surface color index ... 44
Surface color ... 44
Surface edge color index ... 47
Surface edge color ... 47
Surface edge flag ... 47
Surface edge type ... 47
Surface edge width ... 47
Surface interior style index ... 44
Surface interior style ... 44
Surface interpolation method ... 45
Surface reflection attributes ... 44
Surface reflection model ... 44
SURFACE_APPROX ... 18
surface_approx ... 67,80
surface_approx_asf ... 48,81
SURFACE_APPROX_METHOD ... 18
surface_color ... 67,79
surface_color_asf ... 48,81
surface_edges ... 68,80
surface_edges_asf ... 48,81
surface_edge_color ... 68,80
surface_edge_color_asf ... 48,81
SURFACE_EDGE_TYPE ... 19
surface_edge_type ... 68,80
surface_edge_type_asf ... 48,81
surface_edge_width ... 68,80
surface_edge_width_asf ... 48,81
SURFACE_INTERP ... 19
surface_interp ... 67,79
surface_interp_asf ... 48,81
SWITCH ... 19
TABLE_ENTRY ... 19
TABLE_INDEX ... 19,27
TABLE_TYPE ... 19
Text 2D ... 53
Text 3D ... 52
Text alignment ... 42

Text bundle index ... 43
Text color index ... 41
Text color ... 41
Text font index ... 40
Text path ... 42
Text precision ... 41
TEXT_ALIGNMENT ... 19
text_alignment ... 79
text_bundle ... 137,141,90
text_bundle_index ... 79
text_color ... 79
text_color_asf ... 48,81
text_font_index ... 79
text_font_index_asf ... 48,81
text_font_table ... 90
TEXT_HALIGNMENT ... 19
TEXT_PATH ... 19
text_path ... 79
TEXT_PRECISION ... 19
text_precision ... 79
text_precision_asf ... 81
text_prec_asf ... 48
TEXT_VALIGNMENT ... 19
tex_font_table ... 137
Triangle strip ... 59
TRIM_CURVE ... 19
TRIM_CURVE_APPROX_METHOD ... 19
type ... 70
TYPE_OR_TABLE_INDEX ... 19
Unknown ... 62
unpost_structure ... 141
UPDATE_STATE ... 19
Value ... 14
value ... 160
Value ... 36,37
VECTOR ... 14
VECTOR_2D ... 19
VECTOR_3D ... 19
VERTEX ... 19
View index ... 50
VIEWPORT ... 19
viewport ... 90
VIEWPORT ... 92
view_index ... 80
view_rep ... 141
VIEW_REP ... 19
view_table ... 90
view_update ... 137
Visible ... 29
VisibleAny ... 29
VisibleClosest ... 29
VisualizeEach ... 29
VisualizeEasy ... 29
VisualizeNone ... 30
VisualizeWhenever ... 30
VISUAL_ID ... 19,8
visual_state ... 137
VISUAL_STATE ... 19
WCS スポット光源 (WCS spot light source) ... 69
WCS ベクトル光源 (WCS vector light source) ... 69
WCS 点光源 (WCS point light source) ... 69
WCS_ChordalDev ... 25
WCS_ChordalSize ... 25,26
WCS_PlanarDev ... 27
WCS_Point ... 27
WCS_Relative ... 25,27
WCS_Spot ... 27
WCS_Vector ... 27
WC_LevelCurves ... 31
weight1 ... 70
weight2 ... 70
weight3 ... 70
WINDOW_ID ... 19
WKS_BITMASK ... 19
wks_transform ... 141
wks_update ... 137
X ... 24
X11_FONT_ID ... 19
X3D-PEX ... 22
YIQ ... 30
z- バッファ (z-buffering) ... 28
ZBuffer ... 28
ZBufferID ... 28,8
PEXDeleteElements ... 122
Execute Structure ... 101
GetFontPath ... 158
Label ... 101
ListExtensions ... 22
ListFontsWithInfo ... 161
ListFonts ... 161
MAX_HITS_REACHED ... 21,8
Noop ... 8
PEXAccumulateState ... 101,102,104,8,89
PEXBeginPickAll ... 107,91
PEXBeginPickOne ... 104,91
PEXBeginRendering ... 101,20,78,8,80,89,92,97,98
PEXBeginStructure ... 102,104,89,99
PEXChangeNameSet ... 127
PEXChangePickDevice ... 154
PEXChangePipelineContext ... 88
PEXChangeRenderer ... 95
PEXChangeSearchContext ... 131

PEXChangeStructureReferences 125
PEXCloseFont 159
PEXCopyElements 124
PEXCopyLookupTable 72
PEXCopyNameSet 126
PEXCopyPipelineContext 87
PEXCopySearchContext 130
PEXCopyStructure 111
PEXCreateLookupTable 72
PEXCreateNameSet 126
PEXCreatePhigsWKS 134,136
PEXCreatePickMeasure 156,157
PEXCreatePipelineContext 87
PEXCreateRenderer 94
PEXCreateSearchContext 130
PEXCreateStructure 111,13
PEXDeleteElementsBetweenLabels 123
PEXDeleteElementsToLabel 122
PEXDeleteElement 121
PEXDeleteTableEntries 77
PEXDestroyStructures 112
PEXDestroyStructure 51
PEXElementSearch 120
PEXEndPickAll 102,108,92
PEXEndPickOne 102,105
PEXEndRendering 101,89,94,98
PEXEndStructure 102,104,89,99
PEXEscapeWithReply 37,9
PEXEscape 36,8
PEXExecuteDeferredActions 144
PEXFetchElements 113,117,38
PEXFreeLookupTable 73
PEXFreeNameSet 126
PEXFreePhigsWKS 135
PEXFreePickMeasure 156
PEXFreePipelineContext 87
PEXFreeRenderer 94
PEXFreeSearchContext 130
PEXGetAncestors 115
PEXGetDefinedIndices 75
PEXGetDescendants 116
PEXGetDynamics 140,8
PEXGetElementInfo 114
PEXGetEnumeratedTypeInfo ... 102,183,23,37
PEXGetExtensionInfo 164,22
PEXGetImpDepConstants 33,40,43,51,71
PEXGetNameSet 127
PEXGetPickDevice 154
PEXGetPickMeasure 156,157
PEXGetPipelineContext 88
PEXGetPredefinedEntries 74
PEXGetRendererAttributes 8,96
PEXGetRendererDynamics 102,89,92,96,97
PEXGetSearchContext 131
PEXGetStructureInfo 113,114
PEXGetStructuresInNetwork 114
PEXGetTableEntries 76
PEXGetTableEntry 75
PEXGetTableInfo 74
PEXGetViewRep 134,142
PEXGetWKSInfo 136
PEXGetWKSPostings 151
PEXListFontsWithInfo 161
PEXListFonts 161
PEXMapDCtoWC 145
PEXMapWCtoDC 146
PEXMatchRendererTargets ... 104,107,36,8,98
PEXOpenFont 158
PEXPickAll 109,92
PEXPickOne 106
PEXPostStructure 150
PEXQueryFont 160,161
PEXQueryTextExtents 162
PEXRedrawAllStructures 143,144
PEXRedrawClipRegion 144
PEXRenderElements 100,102,104,8,89
PEXRenderNetwork 101,89
PEXRenderOutputCommands 100,102,104,38,89
PEXRenderOutputElements 101
PEXSampleRequest 13
PEXSearchNetwork 112,132
PEXSetDisplayUpdateMode 137,145
PEXSetEchoColor 183,8
PEXSetEditingMode 118
PEXSetElementPointerAtLabel 119
PEXSetElementPointer 118
PEXSetHLHSRMode 148
PEXSetTabelEntries 77
PEXSetViewPriority 144
PEXSetViewRep 147
PEXSetWKSBufferMode 145,149
PEXSetWKSViewport 148,8
PEXSetWKSWindow 147
PEXStoreElements 121,38
PEXUnpostAllStructures 150
PEXUnpostStructure 150
PEXUpdatePickMeasure 112,157
PEXUpdateWorkstation 143
PolyLine 3D 101
QueryExtension 22
Render Picking 98
SendEvent 12

Set Line Color 101
SetFontPath 158
Aborted 102
Add 127
Alloc 149,89
All 102,114
all 120
AlreadyFreed 128,133,152,68,78,89
ambient 69
asfs 81,86
ATextNotConnected 43,79
ATextStyle 24
atext_alignment 54,82
atext_height 54,82
atext_path 54,82
atext_style 54,82
atext_up_vector 54,82
attenuation 69
attribute 47,48
BackFaces 46,84
background_color 8,92,98
back_plane 69
back_scaling 69
base_pixel 70
Beginning 100,114,117,118,120,122,124
Begin 13
BestColorApproxValues 35,71
bf_interior_style_index 84
bf_interior_style 84
bf_reflection_attr 56,84
bf_reflection_model 84
bf_surface_color 56,84
bf_surface_interp 84
BottomFirst 152
BottomPart 115
buffer_mode 134
buffer_update 139,143
Bundled 86
c1 70
c2 70
c3 70
CBG 141
Change 13
char_expansion 53,82
char_height 53,82
char_set_width 52
char_set 41,52
char_spacing 53,82
char_up_vector 53,82
Char 41,53
ChromaticityBlueU 34
ChromaticityBlueV 34
ChromaticityGreenU 34
ChromaticityGreenV 34
ChromaticityRedU 34
ChromaticityRedV 34
ChromaticityWhiteU 34
ChromaticityWhiteV 34
clear_I 8,92,98
clear_Z 8,89,92,98
client_protocol_major_version 22
client_protocol_minor_version 22
ClipList 97
clip_flags 69
clip_limits 69
clip_list 93,98
Clip 49
ClosestZ 102
ColorApproxModel 15,30
ColorApproxTableContents 97
ColorApproxTable 97
ColorApproxType 30
ColorApprox 70
ColorRange 70
ColorSpace 70
ColorTableContents 97
ColorTable 97
ColorType 15,20,27,68
color_approx_index 70,85
color_approx_table 139,91
color_model 70
color_table 138,91
color_type 55,57,58,59,60,62,65,68,70,88
color 40,41,43,44,64
Color 68
color 69
Complex 56,57,58,60,61
comp_type 48
concentration 69
constants 33
contents 86
contours_all_1 61,63
contours 62,63
contour_hint 58,59,62,63
Convex 56,57,58,60,61
Copy 13
Correct 137,143
Create 13
culling_mode 84
current_path 100,91,95,98,99
Current 100,114,117,118,120,122,124
curve_approx 55,83

cur_buffer_mode 139
cur_HLHSR_mode 138
cur_NPC_subvolume 138,147
cur_wks_viewport 148
cur_wks_viewpt 138
DC_HitBox 102,104
default table entry 66
Default 75
Deferred 137,143
defined_views 138
Delete 13
DepthCueTableContents 97
DepthCueTable 97
DepthCue 69
depth_cue_index 85
depth_cue_table 139,91
Destroy 13
dev_type 157
direction 69
Disjoint 58,62
DisplayUpdateMode 29
display_surface 137,143,148
display_update 137,141
distinguish 84
DitheringSupported 35
Dither 71
DontCare 36
DoubleBufferingSupported 35
Double 139,149
drawable_example 33
drawable_id 138
dynamic 97
EchoMode 8,97
EchoPrimitive 29
echo_mode 8,92
echo_switch 153
echo_volume 153
EdgeBundleContents 97
EdgeBundle 68,97
edge_attributes 62
edge_bundle_index 56,85
edge_bundle 138,91
editing_mode 110,121
elements 121
element_ptr 110
Empty 138,148
enables 86
enable 86
encoding_state 52
End 100,114,117,120,122,124,13
entry 66
escape_data 36
Escape 32,8,9
facet_attributes 57,59,60
facet_data 59,60,61
facet 60
False 129,57,80,84,89,94,98
fas_attributes 62
fas_data 62
Fetch 13
fill area 57
FloatFormat 28
FloatingPointFormat 20,38
flush 89,94
Found 120
Free 13
Frontfaces 46
FrontFaces 84
front_plane 69
front_scaling 69
GDP3 32
GDP 32
Get 13
global_transform 85,99
GSE 32
HalignCenter 42
HalignLeft 42
HalignNormal 42,79
HalignRight 42
HatchStyle 25
Hatch 84
Higher 144
HighlightNamesetContents 97
HighlightNameset 97
highlight_exclusion 139,91
highlight_inclusion 139,91
HLHSRMode 28,8,97
HLHSR_identifier 86
HLHSR_mode 148,86,89,92
HLHSR_update 138,143
HLHSR 98
Idle 100,101,102,105,108,90,98,99
ignore_edges 56,57,58,59
IMM 141,148
Implementation 164
Indexed 40,41,43,44,64
indexed 68,79
index 23,66
Individual 55,57,59,80,86
InteriorBundleContents 97
InteriorBundle 67,97
InteriorStyleEmpty 44,45

InteriorStyleHatch 44,45
InteriorStyleHollow 44,45,79
InteriorStylePattern 44,45,46
InteriorStyleSolid 44,45
InteriorStyle 25
interior_bundle_index 56,57,59,85
interior_bundle 138,91
interior_color 83
interior_style_index 56,83
interior_style 44,56,68,83
Intersecting 58,62
intrinsic color 25
inverted_list 129,132
InvisibilityNamesetContents 97
InvisibilityNameset 97
invisibility_exclusion 139,92
invisibility_inclusion 139,91
IRG 141
IsoparametricCurves 51
item_list 128,14,78
item_mask 128,13,14,78
label 119
Label 20
Last 102
LightTableContents 97
LightTable 97
LightType 27
light_state 85
light_table 139,91
Light 69
LineBundleContents 97
LineBundle 67,97
LineTypeSolid 43,79
LineType 24
line_bundle_index 54,55,83
line_bundle 138,55,91
line_color 54,55,83
line_type 54,55,83
line_width 54,55,83
local_transform 85,99
LookupTable 20
Lower 144
LuminanceBlue 34
LuminanceGreen 34
LuminanceRed 34
LuminanceWhite 34
Mapping 69
MarkerAsterisk 40,79
MarkerBundleContents 97
MarkerBundle 66,67,97
MarkerType 24
marker_bundle_index 52,82
marker_bundle 138,91
marker_color 52,82
marker_scale 52,82
marker_type 52,82
Match 104,107,98
max1 70
max2 70
max3 70
MaxEdgeWidth 34
MaxHitsEventSupported 35
MaxLineWidth 34
MaxMarkerSize 34
MaxModelClipPlanes 35
MaxNamesetNames 35
MaxNonAmbientLights 35
MaxNURBOrder 35
MaxTrimCurveOrder 35
max_hits 102
max_targets 36
MC_LevelCurves 51
MinEdgeWidth 34
MinLineWidth 34
MinMarkerSize 34
mnemonic 23
ModelClipOperator 27
model_clip_flag 129
model_clip_volume 85
model_clip 85
more_picks 108
mpts 63
multi1 70
multi2 70
multi3 70
m_pts 61
NameSet 20
names 33
name_set 86
Nested 62
NoClip 49,80
NoCrossRefs 114
NominalEdgeWidth 34
NominalLineWidth 34
NominalMarkerSize 34
NoMoreHits 108
Nonconvex 56,57,58,60,61
None 46,51,80,84
NonRational 55,63
Nonrational 64
Nonuniform 51
NoPick 155,157

normal_list 129,132
NotEmpty 138,143
NotFound 120
NotPending 138,143,147
NPCSubvolume 97
NPC_HitVolume 102,104
NPC_subvolume 92
npts 63
Null 129,132,89,90,93,98
NULL-ID 89
NumSupportedEdgeWidths 34
NumSupportedLineWidths 34
NumSupportedMarkerSizes 34
numx 68
numy 68
num_edges 62
num_fas 62
num_priorities 139
num_verts 62
n_pts 61
offset 100,117,53
off_set 114
Off 47,80,85
Ok 155,157
On 47,56,57,85
order 55,64
Orientation 69
origin 53
OutputCommand 20,38,63
ParametricSurfaceCharacteristics 30
PathDown 42
PathLeft 42
PathRight 42,79
PathUp 42
Path 112,20
PatternTableContents 97
PatternTable 97
pattern_ref_pt 56,84
pattern_ref_vec1 46,56,84
pattern_ref_vec2 46,56,84
pattern_size 56,84
pattern_table 139,91
Pattern 56,68,84
Pending 138,142,147
per_edge_data 62,63
per_facet_data 62
per_fas_data 62
per_vert_data 62
PEXFont 20
PhigsWKS 20
pick device 152
PickAllMethod 29,9
PickDeviceType 29
picked_prim 155
Picking 100,101,104,107,91,94,98,99
PickMeasure 20
PickNamesetContents 97
PickNameset 8,97
PickOneMethod 29,9
pick_data_rec 152,29
pick_distance 102
pick_exclusion 152,8,92
pick_id 86
pick_inclusion 152,8,92
pick_method 102
pick_path_order 152
pick_path 152
pick_position 102
pick_prim 157
pick_start_path 102,8,92
pick_status 102,152,155,157
PipelineContext 20
pipeline_context 91,98
point1 64
point2 64
points 52
point 69
PolylineInterpMethod 24
PolylineInterpNone 43,79
polyline_interp_asf 55
polyline_interp 55,83
position1 121,122,124
position2 121,122,124
PostConcatenate 48
posted_structs 139
PreConcatenate 48
predefined entries 66
PromptEchoType 29
prompt_echo_type 153
psc_data 85
psc_type 85
PseudoColor 71
psurf_char 85
Rational 55,63,64
rdr_color_model 85
Realized 66,75,76
ReflectionModel 25
ReflectionNoShading 44,45,79
reflection_attr 83
reflection_model 26,56,83
Remove 127
RendererState 104,105,107,108,20,98

renderer_state 100,101,91,92,94,95,98,99
Renderer 20
RenderingColorModel 15,30
Rendering 100,101,104,105,107,108,89,91,92,94,98,99
Replace 127,48
Request 164,22
req_buffer_mode 139
req_HLHSR_mode 138
req_NPC_subvolume 138,147
req_wks_viewport 148
req_wks_viewpt 138
RGB_Float 13
RGB 70
SearchContext 20
search_ceiling 128,132
search_dist 128
search_pos 128
Set 13,66,75,76
shaded color 25
shape 56,58,59,60,61,63
Simulated 143
SimulateSome 141,145
Simulate 137
Single 139,149
specular_conc 83
spread_angle 69
src_range 124
start_path 129,132
Store 13
string 52
String 53,79
stroke 128
Stroke 53
StructureInsert 110,118,121
StructureReplace 110,118,121
Structure 20
subset_info 164,22
SurfaceApproxMethod 26
SurfaceEdgeSolid 47,80
SurfaceEdgeType 25
SurfaceInterpMethod 26
SurfaceInterpNone 45,46,79
surface_approx 84
surface_color 56,57,59,60,61,62
surface_edges 85
surface_edge_color 56,85
surface_edge_type 56,85
surface_edge_width 56,85
surface_edge 56
surface_interp_asf 57,59
surface_interp 56,57,59,83
surface_type 63
TextBundleContents 97
TextBundle 67,97
TextFontTableContents 97
TextFontTable 97
TextFont 162,68
text_alignment 42,53,82
text_bundle_index 53,83
text_bundle 138,91
text_color 53,82
text_font_index 53,82
text_font_table 139,91
text_path 53,82
text_precision 53,82
tmax 55
tmin 55
tolerance 24
TopFirst 152
TopPart 115
TransparencySupported 35
TrimCurveApproxMethod 27
True 129,56,57,61,92,98
type 55,63,64,70
undefined entry 66
Uniform 51
Unknown 56,57,58,60,61
use_drawable 92
u_knots 63
u_order 63
ValianNormal 79
ValignBase 42
ValignBottom 42
ValignCap 42
ValignHalf 42
ValignNormal 42
ValignTop 42
value_type 66
vector1 53
vector2 53
vertices 54,56,57,60
vert_attributes 55,57,59,60,62
viewport 92
Viewport 97
ViewTableContents 97
ViewTable 97
view_index 85
view_table 91
view_update 138,143,147
View 68
VisibleAny 102
VisibleClosest 102

Visible 102
VisualizeEach 145
VisualizeEasy 145
VisualizeNone 145
VisualizeWhenever 145
visual_state 137,143
v_knots 63
v_order 63
WCS point 69
WCS spot 69
WCS vector 69
WC_LevelCurves 51
weight1 70
weight2 70
weight3 70
whence 100,114,117
wks_update 138,143,147,148
ZBufferId 86
インスタンス (instance) 13
エコー型 (echo type) 29
エコー色 (echo color) 32
エスケープ (escape) 32
エスケープ定義 (escape-specific) 36
エラー (error) 12,20,38
エラーコード (error code) 12,20
オープン (open) 68
オペコード (opcode) 12
クライアント (client) 13
クライアント側探訪 (client-side traversal) 91
クリップ (clip) 24,54
クローズ (close) 68
グリフ (glyph) 160
グローバル変換行列属性 (*global_transform*) 49
コネクション (connection) 12
サーチ・コンテキスト (search context) 128,20
サーチ・フィルタ条件 (search filter criteria) 128
サーチ操作 (searching operation) 128
サーバ (server) 13
サブセット (subset) 22
サンプリング (sampling) 30
シェーディング (shading) 25
シミュレート (simulate) 30
スキャンライン (scan line) 28
ストラクチャ・ネットワーク・パス (structure network path) 20
ストラクチャ・リソース (structure resource) 110
ストラクチャ・レンダリング (structure rendering) 164
ストラクチャ・レンダリング 8
ストラクチャ呼び出し (structure invocation) 50
ストラクチャ探訪 (structure traversal) 38
ストラクチャ要素 (structure element) 13,38
ストローク・フォント (stroke font) 41
セグメント (segment) 24
セル配列要素 (cell array primitive) 64
ダイナミックス (dynamics) 92
ダブル・バッファリング (double buffering) 145
ダブル・バッファリング (double-buffering) 35
テーブル指標 (table indices) 38
データ・フィールド (data field) 12
データ長フィールド (length field) 12
ディザ (dithering) 71
ディザリング (dithering) 35
ディザリング・ヒント (dithering hint) 35
デプス・キュー・テーブル (depth cue table) 66
デプス・キューイング (depth-cueing) 24
デプス・キュー指標属性 (*depth_cue_index*) 50
トリミング・ループ (trimming loop) 63
トリム曲線 (trim curve) 27
トリム曲線近似法 (trim curve approximation method) 27
ドローアブル (drawable) 23
ハッチ (hatch) 25
ハッチ様式 (hatch style) 25
バイト交換 (byte-swapping) 52
パイプライン・コンテキスト (pipeline context) 20
パイプライン・コンテキスト (*pipeline context*) 78
パケット (packet) 12
パス・オフセット (path offset) 101
パス経路 (track of paths) 91
パターン・テーブル (pattern table) 25
パターン・ベクトル (pattern vector) 46
パターン参照ベクトル (pattern reference vector) 84
パターン参照位置 (pattern reference point) 84
パターン参照点属性 (*pattern_ref_pt*) 46
パターン寸法属性 (*pattern_size*) 46
パディング (padding) 21
パラメトリック (NURBS) 曲面特性属性 (*psurf_char*) 51
パラメトリック曲面特性 (parametric surface characteristics) 85
ヒット・テスト (hit test) 92
ヒット・ボックス (hit box) 29
ビュー (view) 54
ビューポート (viewport) 92
ビュー指標属性 (*view_index*) 50
ピッキング操作 (picking operation) 86
ピック・装置記述子 (*pick device descriptor*) 152
ピックメジャ (pick measure) 20,29
ピックメジャ (*pick measure*) 152,155
ピックメジャ 29
ピック位置 (pick position) 29
ピック距離 (pick distance) 29

ピック口径 (pick aperature) 29
ピック識別子属性 (*pick_id*) 50
ピック除外フィルタ (pick exclusion filter) 92
ピック装置 (pick device) 29
ピック装置 29
ピック装置タイプ (pick device type) 29
ピック包含フィルタ (pick inclusion filter) 92
フィル (fill) 25
フィルタリング (filtering) 30
フィルタ出力コマンド (filter output command) . 126
フェッチ (fetch) 117
フォーマット・ワード (format word) 14
フォント・グループ (font group) 40,41
フォント・ファイル (font file) 41
プロトコル (*protocol*) 12
プロパティ (properties) 160
プロンプト (prompt) 29
プロンプト・エコー型 (prompt echo type) 29
ヘッダ (header) 12
ベンダー (*vender*) 22
ポスト (post) 30
マーカ型 (marker type) 24
マーカ種属性 (*marker_type*) 40
マーカ色属性 (*marker_color*) 40
マーカ寸法 (marker size) 40
マーカ寸法属性 (*marker_scale*) 40
マーカ束指標属性 (*marker_bundle_index*) 40
マーカ要素 (marker primitive) 24
モデリング (modeling) 54
モデリング・クリッピング (modeling clipping) .. 129
モデリング変換 (modeling transformation) 49
モデル・クリップ・ボリューム属性 (*model_clip_volume*) 50
モデル・クリップ識別子 (model clip indicator) ... 49
モデル・クリップ属性 (*model_clip*) 49
モデル・クリップ法 (model clip method) 27
リクエスト (request) 12
リサイズ (resize) 89,93
リソース (resource) 13
リプライ (reply) 12
リリース番号 (*release_number*) 22
ルート (root) 23
ルート・ウィンドウ (root window) 72
ルックアップ・テーブル (lookup table) 20,66
レベル曲線 (level curve) 31
レンダラ (renderer) 20,38,89
レンダラ・ピッキング (renderer picking) 29
レンダリング (Rendering) 20
レンダリング・パイプライン (rendering pipeline) .30
レンダリング色モデル (rendering color model) ... 30
レンダリング色モデル属性 (*rdr_color_model*) 51
ローカル・モデル変換行列 (local model transformation matrix) 48
ローカル変換行列属性 (*local_transform*) 48
ワークステーション・ピッキング (workstation picking) 152,29
ワイルドカード (wildcard) 36
移動 (move) 89
一点鎖線 ("dots and dashes" line) 24
引き出し線 (connection line) 54
引数 14
隠蔽 (occlude) 89
可視性 (visibility) 28
可視性フラグ (visibility flag) 64
階層パス (hierarchical path) 92
外周 (contour) 58
外周線 (edge) 25
外周線 (surface edge) 85
外周線スイッチ (edge switch) 62
外周線型 (surface edge type) 25,85
外周線種属性 (*surface_edge_type*) 47
外周線色 (surface edge color) 85
外周線色属性 (*surface_edge_color*) 47
外周線束指標属性 (*edge_bundle_index*) 47
外周線属性 (*surface_edges*) 47
外周線幅 (surface edge width) 85
外周線幅属性 (*surface_edge_width*) 47
拡散係数 (diffuse coefficient) 83
拡散項 (diffuse term) 26
拡散反射 (diffuse reflection) 26
拡大率 (character expansion) 82
拡張情報 22
環境係数 (ambient coefficient) 83
環境光源 (ambient light source) 69
環境光源 (ambient light) 26
環境項 (ambient term) 26
間接色値参照 (indirect color reference) 68
基点 (origin) 53
幾何要素 (geometric primitive) 52
輝度 (luminance) 28
擬似 (pseudo) 101
逆フィルタ・リスト (inverted filter list) 129
逆行列 (inverse) 145
許容誤差 (tolerance) 24
共通型 (common type) 14
鏡面係数 (specular coefficient) 83
鏡面項 (specular term) 26
鏡面集中指数 (specular concentration) 83
鏡面色 (specular color) 83
鏡面反射 (specular reflection) 26
曲線 (curve) 24
曲線近似 (curve approximation) 83

曲線近似属性 (*curve_approx*) 43
曲線近似法 (curve approximation method) 24
曲面近似属性 (*surface_approx*) 46
曲面近似法 (surface approximation method) 26
曲面近似法 (surface approximation) 84
空 (empty) 51
空間探索 (spatial search) 132
型 (type) 12,15
型コード (type code) 12
型ニューモニック (type mnemonic) 13
型フィールド (type field) 12
穴空き多角形 (polygon with holes) 58
顕示 (highlight) 29
顕示除外集合 (highlight exclusion set) 91
顕示包含集合 (highlight inclusion set) 91
減衰係数 (attenuation coefficient) 27
限界 (ceiling) 128
後処理 (post-processing) 24
後方デプス・キューイング面 (back depth cueing plane) 70
光源 (light source) 26
光源型 (light type) 27
光源情報 (light source description) 66
光源状態属性 (*light_state*) 50
光源表 (light table) 50
広がり角 (spread angle) 27
構成要素 (structure component) 13
構文 (Syntax) 13
高速処理コード (fast rendering code path) 61
再生成 (regeneration) 12
彩度 (saturation) 28
三角形ストリップ要素 (triangle strip primitive) . . 59
四辺形メッシュ要素 (quadrilateral mesh primitive) 60
子階層 (descendant) 110
指標色 (index color) 14
指標値 (index value) 14
視野 (view) 133
視野テーブル (view table) 66
視野参照座標系 (View_reference_coordinates 69
視野情報 (view information) 66
事象 (event) 12,21
事象コード (event code) 12
次数 (*order*) 55
識別モード (culling mode) 84
実線 (solid) 24
実体 (entity) 133
主要オペコード (major opcode) 12
集中指数 (concentration exponent) 27
縮退 (degenerate) 42
出力コマンド (output command) 12,38
出力ドローアブル (destination drawable) 12
書き込みプロテクト (write-proteced) 93
小面 (facet) 57
照明 (illumination) 26
照明方程式 (lighting equation) 26
色 (color) 14
色マップ (colormap) 70
色ルックアップ・テーブル (color lookup table) 14,38
色近似テーブル (color approximation table) 66
色近似モデル (color approximation model) 30
色近似型 (color approximation type) 30
色近似指標属性 (*color_approx_index*) 50
色空間 (color space) 70
色型 (color type) 14,27,38
色相 (hue) 28
色値 (color value) 38
深さ (depth) 23,72
親階層 (ancestor) 110
水平／垂直配置 (text alignment) 82
世界座標系 (world coordinates) 25,69
制御点 (control point) 55
正規化投影座標系 (normalized project coordinates) 25,92
正規化投影座標系 (normalized projection coordinates) 69
切り換え (switching) 68
折線 (polyline) 24
折線シェーディング法 (polyline interpolation method) 24,83
折線シェーディング法属性 (*polyline_interp*) 43
節点 (knot) 24,55
先頭 (head) 116
線形補間 (linearly interpolating) 24
線種型 (line type) 24
線種属性 (*line_type*) 43
線色属性 (*line_color*) 43
線束指標属性 (*line_bundle_index*) 44
線幅属性 (*line_width*) 43
前方クリップ面 (front clipping plane) 29
前方デプス・キューイング面 (front depth cueing plane) 70
全ピック (pick all) 102,92
全ピック法 (pick all method) 29
装置座標系 (device coordinates) 25,29
即時 (immediate) 54
即時レンダリング (immediate rendering) 164,22
属性 (attribute) 13
多角形集合要素 (fill area set) 58
退避 (save) 51,99
単一ピック (pick one) 102
単一ピック法 (pick one method) 29

単調 (flat meaning) 101
探訪 (pick traversal) 21
遅延実行 (later excution) 110
抽象 (abstraction) 133
注釈位置 (annotation point) 53
注釈文字上方向ベクトル属性 (*atext_up_vector*) 42
注釈文字列 (annotation text) 24,41
注釈文字列高属性 (*atext_height*) 42
注釈文字列配置属性 (*atext_alignment*) 42
注釈文字列方向属性 (*atext_path*) 42
注釈文字列様式属性 (*atext_style*) 43
頂点 (vertices) 54
直値 (direct value) 14
点光源 (positional light source) 69
点線 (dotted line) 24
等パラメトリック曲線 (isoparametric curve) 31
等方マッピング (isotropic mapping) 93
透過係数 (transmission coefficient) 83
透視投影 (perspective projection) 69
動的変更 (dynamic modification) 29
同次モデリング座標系 (homogeneous modeling coordinates) 55
凸型多角形 (convex) 56
内積項 (dot product) 26
内部シェーディング法属性 (*surface_interp*) 45
内部スタイル束指標属性 (*interior_bundle_index*) .. 46
内部様式 (interior style) 25,83
内部様式指標 (interior style index) 83
内部様式指標属性 (*interior_style_index*) 44
内部様式属性 (*interior_style*) 44
内包多角形 (polygon with islands) 58
破壊 (destroy) 89
破線 (dashed line) 24
半空間 (halfspace) 27,49
反射モデル (reflection model) 25,83
反射モデル属性 (*reflection_model*) 44
反射値 (reflection value) 84
反転 (invert) 129
非同次モデリング座標 (non-homogenous modeling coordinates) 55
非有理 (*NonRational*) 55
標準フィルタ・リスト (normal filter list) 129
表示更新モード (display update mode) 29
不可視除外集合 (invisibility exclusion set) 92
不可視包含集合 (invisibility inclusion set) 91
不透明 (opaque) 83
付加情報 (additional information) 57
浮動小数型 14
浮動小数形式 (floating-point format) 14
副オペコード (minor opcode) 12
復元 (restore) 50,51,99
文字拡大因子 (character expansion factor) 128
文字拡大率属性 (*char_expansion*) 41
文字間隔 (character spacing) 82
文字間隔属性 (*char_spacing*) 41
文字高 (character height) 41,82
文字高属性 (*char_height*) 41
文字上方向ベクトル (character up vector) 82
文字列 (text) 41
文字列フォント・テーブル (text font table) 40
文字列フォント指標 (text font index) 41
文字列フォント指標属性 (*text_font_index*) 40
文字列色属性 (*text_color*) 41
文字列束指標属性 (*text_bundle_index*) 43
文字列表示精度属性 (text precision) 53
文字列表示精度属性 (*text_precision*) 41
文字列方向 (text path) 82
文字列方向属性 (*text_path*) 42
平行光源 (directional light source) 69
平行投影 (parallel projection) 69
平行筒 (parallelpipe) 29
変更 (modification) 118
編集 (editing) 118
放射円錐 (cone of influence) 69
方向ベクトル (direction vector) 27
法線 (normal) 26
末節 (leaf node) 116
末尾 (tail) 116
無効リスト (disable list) 50
名前セット (name set) 126,20
明度 (lightness) 28
明度 (value) 28
面識別フラグ属性 (*distinguish*) 46
面識別モード属性 (*culling_mode*) 46
有効リスト (enable list) 50
有理 (*Rational*) 55
要素型 (element type) 14
裏面内部シェーディング法属性 (*bf_surface_interp*) 46
裏面内部スタイル属性 (*bf_interior_style*) 45
裏面内部様式指標属性 (*bf_interior_style_index*) ... 45
裏面反射モデル属性 (*bf_reflection_model*) 45
裏面反射特性属性 (*bf_reflection_attr*) 45
裏面領域色属性 (*bf_surface_color*) 45
裏面領域要素 (backfacing surface primitive) 84
立ち上がり方向ベクトル (up vector) 162
領域 (surface) 25
領域シェーディング法 (surface interpolation method) 83
領域色 (surface color) 26
領域色指標 (*surface_color*) 44
領域反射特性属性 (*reflection_attr*) 44
領域要素 (surface primitive) 83

累積 (accumulate) 101
累積状態探訪 (accumulate state traversal) 65
列挙型 (enumerated type) 23
連続番号フィールド (sequence number field) 12
露出 (expose) 89