OpenGL ES2.0
クロスプラットフォームシェーダ開発

株式会社セガ
CS R&D推進部
林 洋人

自己紹介

- 1997年入社
- Dreamcast SDK開発

- 社内向けおよび外注向けグラフィックス開発サポート
  - Dreamcast, GAMECUBE, Wii, LINDBERGH (OpenGL2.0), PSP, DirectX9, iPhone, Nintendo3DS, …

- ソニックワールドアドベンチャー(Wii版)描画
- リルぷりっゆびぷるひめチェン!描画

OpenGL ES

- 3Dグラフィックス API
  - 組み込み機器向けOpenGL
  - Khronos Group
- OpenGLの仕様を整理
  - 現代のハードウェアに沿った形に
  - 互換性のためのAPIを廃止
  - 実装が小さくなるように
- ハードウェアの能力に合わせたバージョン
  - OpenGL ES1.0 CL, CM
  - OpenGL ES1.1 CL, CM
  - OpenGL ES2.0

OpenGL ES2.0

- OpenGL 2.0ベース
- プログラマブルシェーダ必須

- 仕様のさらなる整理
  - OpenGL ES1.1の上位互換ではない
    - ただし、ES2.0対応プラットフォームはES1.1にも対応していることが多い
  - OpenGL 2.0 のサブセットに近いが一部違う

- 最近のスマートフォン用チップはほぼ対応

クロスプラットフォームの夢（その１）

- あるモバイル機

- さるアーケードボード

OS, 性能は違うがどちらも
OpenGL ES2.0 に対応

クロスプラットフォームの夢（その２）

- iPhone 3GS
- iPod Touch
  （第三世代）
- iPhone 4
  数千万台

- iPad
  数百万台

クロスプラットフォームの夢（その３）

- Android 2.1～
  - Desire
  - Droid
  - 今後も続々と発売予定

これら全てが

OpenGL ES2.0 対応

ばんざい！

夢の統一規格

- 真のクロスプラットフォーム時代
  - モバイル機でも PLAYSTATION3, Xbox360 のようなプログラマブルなシェーダ
  - グラフィックスプログラミングはプラットフォームを問わなくなる
  - オープンな規格で対応プラットフォームも多いため、公開情報も充実している

CEDECで講演するようなことは何もない

夢破れました（その１）

とはいきませんでした…

- 多くの OpenGL ES2.0 情報はプラットフォーム間の同じところについて説明している
- 実際の開発では違うところに問題が発生する
- 似ているということは違うということ

夢破れました（その２）

そこで

- まず同じOpenGL ES2.0プラットフォーム間の違いを理解する
- その上であらためて、クロスプラットフォーム化についてのアイディアへ
- アートワークや実装に関するTIPS

プラットフォームによる違い

- GPUコア
- OpenGL ES2.0拡張
- ターゲットOS
- 開発環境
- 性能

GPUコア

- 主要なGPUコアアーキテクチャ
  - Imagination Technologies PowerVR SGX
  - AMD Z430
  - NVIDIA Tegra 250
  - ARM Mali-200 / Mali-400MP
  - Marvell ARMADA 5xx
  - DMP SMAPH-S

PowerVR SGX

- Samsung S5PC100
  - iPhone 3GS
- Apple A4
  - iPhone 4, iPad
- Freescale i.MX515
  - NetWalker
- Texas Instruments OMAP35x
  - Androidタブレット

AMD Z430

- AMD Z430
  - Qualcomm Snapdragon
  - 非常に多くのスマートフォンに採用
  - ATI → AMD → Qualcomm

その他のGPUコア

- NVIDIA Tegra 250
  - dynabook AZ
- ARM Mali-200 / Mali-400MP
  - CPUはほぼARMに統一された
- Marvell ARMADA
  - スマートフォンの低価格化？
- DMP SMAPH-S
  - Nintendo3DS のPICA200 はOpenGL ES2.0ではない

CPU

- CPU側もバリエーション

- ほとんどが ARM系
  - ARM11 (ARMv6)
  - Cortex-A8 (ARMv7)

- グラフィックスに有効な拡張命令
  - VFP命令
  - NEON命令

OpenGL ES拡張（その１）

- OpenGL ES2.0拡張
  - 追加拡張機能
  - ハードウェア機能の違いを反映
    例：便利な機能が多数追加されているGPU
    例：ほとんど拡張が使えないエミュレータ
  - ドライバの違い

  - 拡張といっても避けては通れない
    例：テクスチャ圧縮は全て拡張

  - glGetString(GL_EXTENSIONS)

OpenGL ES2.0拡張（その２）

	A	B	C	D	E	F	G	H	I	J
1		iPhone 3GS OS4.0	iPad	あるボード	ある端末	Qualcomm Adreno Emulator	AMD GLES2.0 Emulator	NVIDIA ES2.0 Emulator	ARM Mali Emulato	PVR GLES Emula
2										
3	GL_AMD_alpha_test	×	×	×	×	○	○	×	×	×
4	GL_AMD_compressed_3DC_texture	×	×	×	○	○	○	×	×	×
5	GL_AMD_compressed_ATC_texture	×	×	×	○	○	○	×	×	×
6	GL_AMD_logic_op	×	×	×	×	○	○	×	×	×
7	GL_AMD_performance_monitor	×	×	×	○	×	×	×	×	×
8	GL_AMD_program_binary_Z400	×	×	×	○	○	○	×	×	×
9	GL_AMD_tiled_rendering	×	×	×	○	×	×	×	×	×
10	GL_AMD_writeonly_rendering	×	×	×	×	×	○	×	×	×
11	GL_APPLE_framebuffer_multisample	○	×	×	×	×	×	×	×	×
12	GL_APPLE_rgb_422	○	×	×	×	×	×	×	×	×
13	GL_APPLE_texture_format_BGRA8888	○	×	×	×	×	×	×	×	×
14	GL_APPLE_texture_max_level	○	×	×	×	×	×	×	×	×
15	GL_IMG_read_format	○	○	○	×	×	×	×	×	×
16	GL_IMG_program_binary	×	×	○	×	×	×	×	×	×
17	GL_IMG_shader_binary	×	×	○	×	×	×	×	×	×
18	GL_IMG_texture_compression_pvrtc	○	○	○	×	×	×	×	×	○
19	GL_IMG_texture_format_BGRA8888	×	○	○	×	×	×	×	×	×
20	GL_IMG_texture_npot	×	×	○	×	×	×	×	×	×
21	GL_IMG_texture_stream2	×	×	○	×	×	×	×	×	×
22	GL_NV_log_textures	×	×	×	×	×	×	○	×	×
23	GL_EXT_blend_minmax	○	○	×	×	×	×	×	×	×
24	GL_EXT_multi_draw_arrays	×	×	○	×	×	×	×	×	○
25	GL_EXT_texture_filter_anisotropic	○	×	×	○	○	○	×	×	×
26	GL_EXT_texture_format_BGRA8888	×	×	○	×	×	×	×	×	×

OpenGL ES2.0拡張（その３）

	A	B	C	D	E	F	G	H	I	J	K	L
1		iPhone 3GS OS4.0	iPad	あるボード	ある端末	Qualcomm Adreno Emulator	AMD GLES2.0 Emulator	NVIDIA ES2.0 Emulator	ARM Mali Emulato	PVR GLES2 Emulator	iPhone Simulator 3.2	iPhone Simulator 4.0
2												
3	GL_AMD_alpha_test	×	×	×	×	○	○	×	×	×	×	×
4	GL_AMD_compressed_3DC_texture	×	×	×	○	○	○	×	×	×	×	×
5	GL_AMD_compressed_ATC_texture	×	×	×	○	○	○	×	×	×	×	×
6	GL_AMD_logic_op	×	×	×	×	○	○	×	×	×	×	×
7	GL_AMD_performance_monitor	×	×	×	○	×	×	×	×	×	×	×
8	GL_AMD_program_binary_Z400	×	×	×	○	○	○	×	×	×	×	×
9	GL_AMD_tiled_rendering	×	×	×	○	×	×	×	×	×	×	×
10	GL_AMD_writeonly_rendering	×	×	×	×	×	○	×	×	×	×	×
11	GL_APPLE_framebuffer_multisample	○	×	×	×	×	×	×	×	×	×	○
12	GL_APPLE_rgb_422	○	×	×	×	×	×	×	×	×	×	○
13	GL_APPLE_texture_format_BGRA8888	○	×	×	×	×	×	×	×	×	×	○
14	GL_APPLE_texture_max_level	○	×	×	×	×	×	×	×	×	×	○
15	GL_IMG_read_format	○	○	○	×	×	×	×	×	×	○	○
16	GL_IMG_program_binary	×	×	○	×	×	×	×	×	×	×	×
17	GL_IMG_shader_binary	×	×	○	×	×	×	×	×	×	×	×
18	GL_IMG_texture_compression_pvrtc	○	○	○	×	×	×	×	×	○	○	○
19	GL_IMG_texture_format_BGRA8888	×	○	○	×	×	×	×	×	×	○	×
20	GL_IMG_texture_npot	×	×	○	×	×	×	×	×	×	×	×
21	GL_IMG_texture_stream2	×	×	○	×	×	×	×	×	×	×	×
22	GL_NV_log_textures	×	×	×	×	×	×	○	×	×	×	×
23	GL_EXT_blend_minmax	○	○	×	×	×	×	×	×	×	○	○
24	GL_EXT_multi_draw_arrays	×	×	○	×	×	×	×	×	○	×	×
25	GL_EXT_texture_filter_anisotropic	○	×	×	○	○	○	×	×	×	×	○
26	GL_EXT_texture_format_BGRA8888	×	×	○	×	×	×	×	×	×	×	×
27	GL_EXT_texture_type_2_10_10_10_REV	×	×	×	○	○	○	×	×	×	×	×
28	GL_EXT_bgra	×	×	×	○	×	×	×	×	×	×	×
29	GL_EXT_discard_framebuffer	○	×	×	×	×	×	×	×	×	×	○
30	GL_EXT_read_format_bgra	○	×	×	×	×	×	×	×	×	×	×
31	GL_EXT_texture_compression_dxt1	×	×	×	×	×	×	○	×	×	×	×
32	GL_EXT_texture_compression_s3tc	×	×	×	×	×	×	○	×	×	×	×
33	GL_EXT_shader_texture_lod	○	×	×	×	×	×	×	×	×	×	○
34	GL_OES_compressed_ETC1_RGB8_texture	×	×	○	○	○	○	×	○	○	×	×
35	GL_OES_compressed_paletted_texture	×	×	×	○	○	○	○	×	×	×	×
36	GL_OES_depth_texture	○	×	○	○	○	○	×	×	○	×	○
37	GL_OES_depth24	○	○	○	○	○	○	×	×	×	○	○
38	GL_OES_depth32	×	×	×	×	×	×	×	×	×	×	×
39	GL_OES_element_index_uint	×	×	○	○	○	○	○	×	○	×	×
40	GL_OES_fbo_render_mipmap	○	○	×	○	×	×	×	×	×	○	○
41	GL_OES_fragment_precision_high	×	×	○	○	○	○	×	×	○	×	×
42	GL_OES_framebuffer_object	×	×	×	×	×	×	○	×	×	×	×
43	GL_OES_get_program_binary	×	×	○	○	×	○	×	×	×	×	×
44	GL_OES_mapbuffer	○	○	○	×	×	×	○	×	○	○	○
45	GL_OES_packed_depth_stencil	○	○	×	○	○	○	×	×	×	×	×
46	GL_OES_read_format	×	×	×	×	×	×	×	×	×	×	×
47	GL_OES_required_internalformat	×	×	○	×	×	×	×	×	×	×	×
48	GL_OES_rgb8_rgba8	○	○	○	○	○	○	○	×	○	○	○
49	GL_OES_shader_binary	×	×	×	×	×	×	×	×	×	×	×
50	GL_OES_shader_source	×	×	×	×	×	×	○	×	×	×	×
51	GL_OES_standard_derivatives	○	○	○	○	○	○	×	×	×	×	×
52	GL_OES_stencil1	×	×	×	×	×	×	×	×	×	×	×
53	GL_OES_stencil4	×	×	×	×	×	×	×	×	×	×	×
54	GL_OES_stencil8	×	×	×	×	×	×	○	×	×	×	×
55	GL_OES_texture_3D	×	×	×	○	○	○	×	×	×	×	×
56	GL_OES_texture_float	○	×	○	○	○	○	×	×	○	×	○
57	GL_OES_texture_float_linear	×	×	×	○	○	○	×	×	×	×	×
58	GL_OES_texture_half_float	○	×	○	○	○	○	○	×	○	×	○
59	GL_OES_texture_half_float_linear	×	×	×	○	○	○	×	×	×	×	×
60	GL_OES_texture_npot	×	×	×	○	○	○	×	×	×	×	×
61	GL_OES_vertex_array_object	○	×	×	×	×	×	×	×	×	×	○
62	GL_OES_vertex_half_float	×	×	○	○	○	○	×	×	○	×	×
63	GL_OES_vertex_type_10_10_10_2	×	×	×	○	○	○	×	×	×	×	×
64	GL_OES_EGL_image	×	×	○	○	×	×	×	×	×	×	×

拡張に類するもの

- インプリメンテーション定数
  - テクスチャ枚数
  - テクスチャサイズ
  - シェーダ定数
- シェーダバイナリ対応
  - OpenGL2.0 から OpenGL ES2.0 で追加された機能

シェーダバイナリ（その１）

- 実機上シェーダコンパイルは遅い
  - CPUパワーが低い
  - 多くのシェーダバリエーション
  - オフラインで予めシェーダをコンパイル

シェーダバイナリ（その２）

- プリコンパイルができない環境もある
  - 仕様上サポートしなくてもよい
  - GL_NUM_SHADER_BINARY_FORMATS
  - 逆にバイナリにしか対応していないプラットフォームがある

シェーダバイナリ（その３）

- OpenGL 2.0 ではサポートされていない
  - PCとの互換性
  - NVIDIA Cg を –oglsl オプションで使用すると GLSLをアセンブリにプリコンパイルできる
- OpenGL 4.1
  - ES2.0上位互換仕様
  - バイナリシェーダに対応
  - OpenGL ES2.0拡張は？

ターゲットOSと端末 （その１）

- iOS4.0
  - iPhone 3GS
  - iPod Touch（第三世代）
  - iPhone 4
- iOS3.2
  - iPad
  - iOS4 環境でも iOS3.2 用アプリは動作する
  - 使用できる OpenGL ES2.0 拡張が違う

ターゲットOSと端末 （その２）

- Android 2.0以降で対応
  - Android 1.6 は OpenGL ES2.0 未サポート
    - Xperia (Android 1.6)
    - IS01 (Android 1.6)

ターゲットOSと端末 （その３）

- Android 2.1
  - Snapdragon
    - Google Nexus One
    - HTC Desire X06HT
    - Dell Streak
  - OMAP35xx
    - Motorola Droid
  - Tegra
    - TOSHIBA dynabook AZ
  - PowerVR SGX
    - GALAXY S

ターゲットOSと端末 （その４）

- Windows Mobile 6.5
  - Toshiba T-01A, X-02T (Snapdragon)
- Windows Phone 7
  - OpenGL ES2.0 は非対応
  - グラフィックス開発は XNA に移行

ターゲットOSと端末 （その５）

- モバイル向けOSならなんでも
  - Palm webOS
    - Palm Pre Plus / Pixi Plus
  - Symbian
    - Sony Ericsson Satio
  - Linux系
    - Samsung Bada
    - Nokia, Intel MeeGo
    - Netwalker (Ubuntu)
  - Brew
  - LiMo

その他のターゲット

- WebGL
  - HTML5 canvas に描画
  - Javascript

ターゲットOSで開発環境や
プログラミング言語が決まってくる

開発環境

- 開発環境の違い
  - プログラミング言語
  - ホストOS
  - コンパイラ
  - エミュレータ／シミュレータ

  - SDKとして提供される

プラットフォームベンダ製SDK（その１）

- APPLE
  - iOS SDK
  - iPhone, iPodTouch, iPad 専用
  - Mac 環境
  - シミュレータの互換性が高い
    - しかし完全ではない
  - ドキュメントも充実
  - コミュニティもクローズだが活発

プラットフォームベンダ製SDK（その２）

- Imagination Technologies
  - PowerVR SDK
  - iPhone 以外の PowerVR 採用環境
  - Windows, Linux ホスト
  - 充実している
    - アートパイプライン
    - シェーダツール
    - 掲示板
  - 日本語サイトは古い

プラットフォームベンダ製SDK（その３）

- AMD
  - 安定していたがサポート終了
- Qualcomm
  - Adreno SDK
  - AMD環境を引き継いでいる
- NVIDIA
  - Tegra Development Tools
- ARM
  - Mali SDK
- Texas Instruments
  - OMAP Graphics SDK

フレームワーク（その1）

- EGL
  - C言語
  - Windows, Linux など
  - AMD, PowerVR, Tegra, Mali SDK
  - 互換性が比較的高い

  - NVIDIA, PowerVR, AMD, ARM 環境を同時にインストールしてビルド設定で切り替えられるようにする

フレームワーク（その２）

- EAGL
  - iOS SDK
  - Objective-C 言語
- Objective-C
  - Objective-C はC言語の上位互換
  - C++も使える

フレームワーク（その３）

- Android
  - Java 言語
- Android NDK
  - Java からネイティブコードを呼び出せる
  - GLSurfaceView

フレームワークとアプリケーションを分離しておくと、アプリケーションとライブラリは C/C++ で書ける

Android NDK （その１）

- JNI (Java Native Interface)
  - ネイティブコードを呼び出すことができる
- Android NDK (Native Development Kit) で C/C++ で記述
  - 高速
  - バイナリの非互換性
  - OpenGL ES2.0 に対応した NDK r4(b)

Android NDK （その２）

- エミュレータが OpenGL ES2.0 に未対応
  - OpenGL ES1.1には対応しているのに…
  - 実機の入手性
- デバッグ環境が未整備
  - gdb を使う
  - マルチスレッドに未対応
  - OpenGL ES は別スレッド動作
- まだまだこれから
  - しかしオープンなので掲示板などで情報が得やすい

描画ライブラリ移植

- OpenGL2.0向けライブラリとシェーダを移植

- 開発環境
  - Windows上のES2.0エミュレータ環境
  - iPhoneシミュレータ環境
  - iPhone 3GS実機
  - iPad実機

- iPad, iPhone 3GS で家庭用ゲーム機向けデータを描画してみた

サンプルデータ

- 次世代機向け
- 14240頂点
- 41ボーン
- ９マテリアル
- マルチテクスチャ
  - カラーマップ
  - スペキュラマップ
  - 法線マップ
- フラグメントライティング

iPad

- あっさり表示された！
- iPadは表示されたものを他人に見せるのに最適

6FPSですけど…

iPod Touch（第三世代）

- iPod Touch のほうが性能は低いのに速い
- 解像度が違いすぎる

まあそれでも9FPSとかですけど…

裏面の特定の場所が熱々になります
iPhone4も似たようなものです

性能差

- 性能
  - ハードウェア性能の違い
  - コンシューマ機との差

  - バランスの違い
    - 頂点処理
    - フラグメント処理
    - CPU性能
    - 帯域

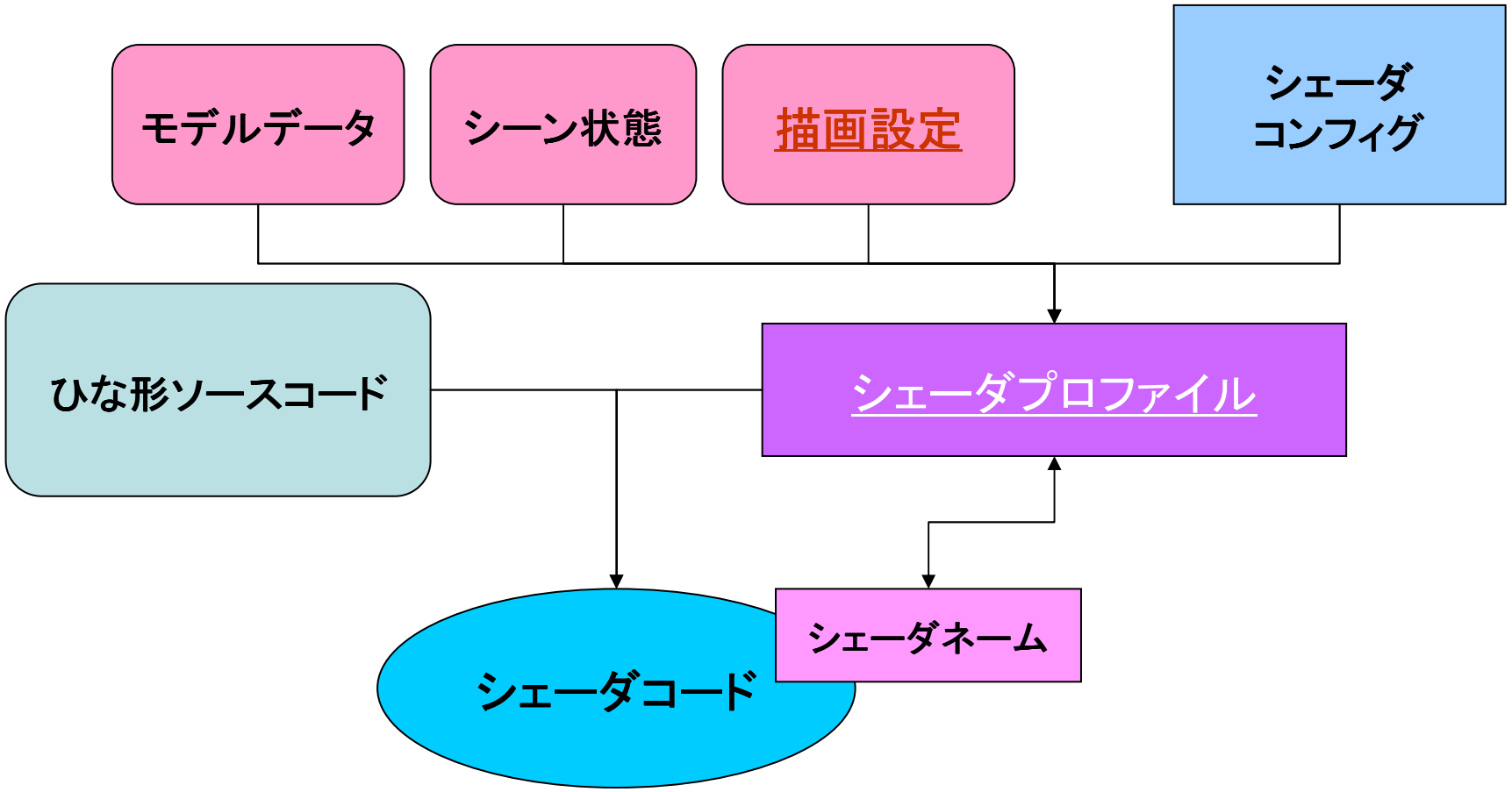
  - 結局プラットフォーム最適化？

OpenGL ES2.0に適した実装

- スケーラビリティを確保
  - シェーダビルドシステム
  - シェーダコンフィグレーション
- 頂点フォーマット
- テクスチャ
- アルファテスト
- その他

シェーダビルドシステム

- 条件に応じて必要なシェーダのバリエーションを作成する



シェーダプロファイルの構成要素（その１）

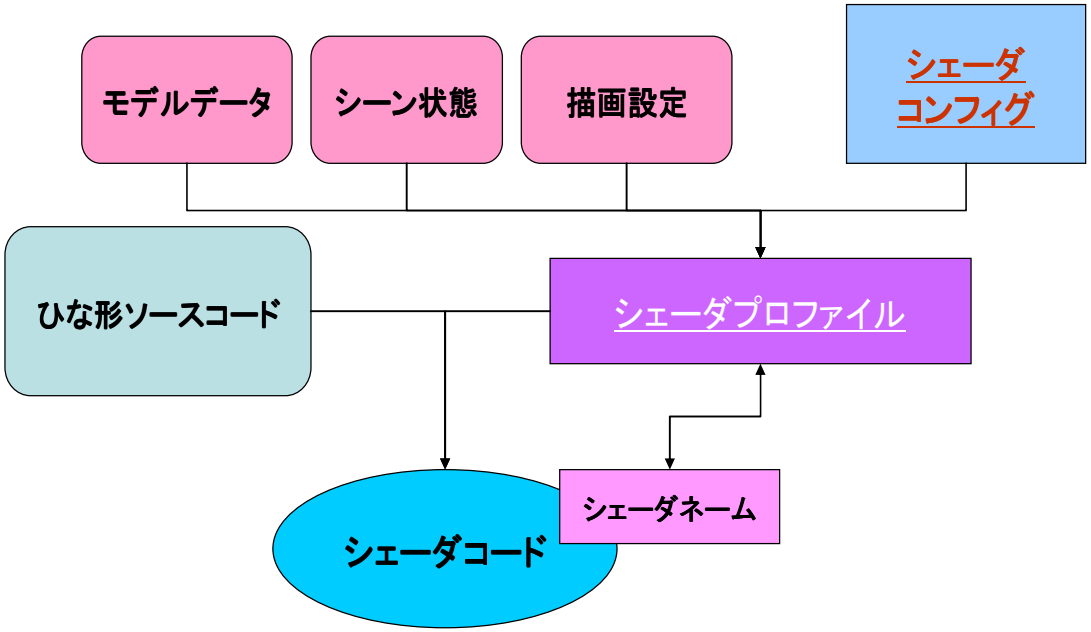
- マテリアル
  - テクスチャ枚数、環境、ブレンド
  - ライティングモデル
- 頂点
  - スキニング
  - テクスチャ座標
  - 頂点カラー
  - 法線／接線／従法線

シェーダプロファイルの構成要素（その２）

- シーン
  - ライト
  - フォグ
- 描画設定
  - インタラクティブな変更
  - シャドウ描画
  - シェーダLOD
  - デバッグ描画

シェーダプロファイルの構成要素（その３）

- シェーダコンフィグ
  - 全てのシェーダで共通になる設定
    - 最大ライト個数
    - 最大ボーン数
    - フォグモデル



  - プラットフォームに合わせた基本設定

シェーダコンフィグ

- 精度指定
  - OpenGL ES2.0 ではシェーダ内変数に精度を指定できる
    - highp
    - mediump
    - lowp
  - ハードウェアが実際に対応していて精度を下げると処理が速い

精度指定（その１）

- highp
  - 2^{-62} ～ 2^{62}
  - 相対精度16ビット
  - フラグメントシェーダでは指定できない場合がある
  - GL_FRAGMENT_PRECISION_HIGH

精度指定（その２）

- mediump
  - 2^{-14} ～ 2^{14}
  - 相対精度10ビット
  - 意外に精度がある
    - 10ビット → 1m に対する 1mm
- lowp
  - -2 ～ 2
  - 絶対精度 1/256
  - カラーや方向ベクトル用途

精度指定（その３）

- 精度を使い分ける
  - シェーダ内変数宣言でマクロ指定
  - 位置座標タイプ
  - テクスチャ座標タイプ
  - 色タイプ
  - 方向ベクトルタイプ
  - タイプごとに調整

精度指定（その４）

- 設定例

高精度
位置座標

中精度
テクスチャ座標

低精度
色要素
方向ベクトル要素

精度指定（その５）

- デフォルト指定
  - precision mediump float など
  - デフォルトのデフォルト
    - 頂点シェーダ　highp
    - フラグメントシェーダは指定を省略できない

精度指定（その６）

- uniform, varyingの精度
  - 頂点シェーダとフラグメントシェーダで同じ精度が必要
  - フラグメントシェーダで高精度が使えない場合
    - 中精度か低精度に統一する
- エミュレータは無視するだけ
  - 実機で確認する必要がある

テクスチャの最適化

- テクスチャ圧縮
- ファイルフォーマット
- アーティスト向けツール

テクスチャ圧縮

- ゲーム開発にテクスチャ圧縮は必須
  - メモリサイズ
  - 描画性能（帯域幅）
    - UMA
  - 配信容量
    - 3Gで配信できるか
- ハードウェアによって対応している圧縮方式が違う
- インストール時やロード時の圧縮も難しい

テクスチャ圧縮形式

- パレット（インデックス）テクスチャ
- ETC
- PVRTC
- ATC
- DXTC (S3TC)

- どれも OpenGL ES2.0 のAPI使用方法は同じなので、全てに対応しておくのがよい

パレットテクスチャ

- OES_compressed_paletted_texture 拡張

- 4bit（16色） / 8bit （256色） ＋ パレット
- OpenGL ES2.0 の標準拡張
- PS2開発などでおなじみ

- iPhone 3GS は対応していない
- APIが用意されていてもハードウェア的に対応しているとは限らない
- ミップマップに弱い

ETC

- GL_OES_compressed_ETC1_RGB8_texture 拡張

- Khronosが提唱する標準的な圧縮形式
- 4bit/pixel
- 多くのビデオチップが対応している
  - PowerVR SGX, AMD Z430, NVIDIA Tegra, ARM Mali

- α値を持てない
- ハードウェアは対応しているはずだが iPhone 3GS では使えない

PVRTC

- GL_IMG_texture_compression_pvrtc 拡張
- 2bit/pixel, 4bit/pixel
- にじむがかなり強力な圧縮

- iPhone, iPad などPowerVR系GPUで使える

- 他社のGPUではサポートされない

S3TC

- GL_EXT_texture_compression_s3tc 拡張
- おなじみDXTCと同じもの
  - DXT1, DXT3, DXT5
- NVIDIA Tegra で使える
- GL_EXT_texture_compression_latc という Luminance-Alpha用も

ATC

- GL_AMD_compressed_ATC_texture 拡張

- 4bit/pixel, 8bit/pixel
- AMD系GPUで使える
- DXTCと同じように使える
  - DXT1, DXT3, DXT5 相当

- アルゴリズムが非公開
  - 圧縮にはAMD提供のツールを使うしかない
- 他社のGPUでは使えない

テクスチャ圧縮ツール

- 各ベンダがライブラリやコマンドライン ツールを提供している
  - nvcompress (NVIDIA)
  - ATI_compress (AMD)
  - PVRTexTool (Imagination Technologies)
  - Mali GPU Texture Compression Tool (ARM)

ファイルフォーマット

- ファイルフォーマットに何を採用するか？
  - OpenGL ESとしては特に定義されていない
  - TGA, PNG, DDS, PVR, etc …
  - 圧縮、ミップマップ、パレットに対応したい

TGA, PNG

- 汎用でアーティストが扱いやすい
- ミップマップや圧縮に対応しない

- 読み込み時にハードウェアに合わせた変換が必要
  - 開発中には使えそうだがモバイル機には重い

DDS

- 事実上の標準
- 表現範囲が広い
  - テクセルフォーマット、圧縮形式、浮動小数点形式
  - ミップマップ
  - パレットも記述できる
- 範囲が広すぎてツールによって対応範囲がまちまち
- PVRTCが格納できない
  - 独自に拡張する？

PVR

- 各種テクセルフォーマットに対応
- PVRTC, ETC圧縮に対応

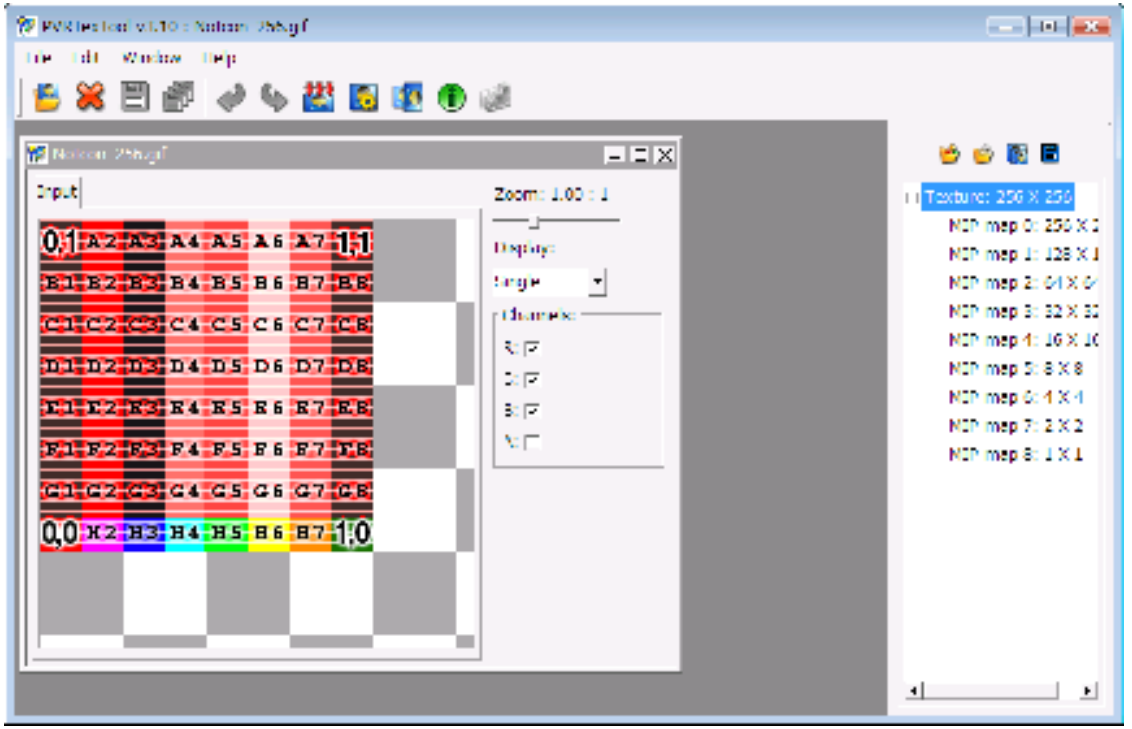
- PVRTexTool は良好
- 他に対応ツールがない

アーティスト向けツール

- 画像を確認しながら変換できるツールは？
  - PVRTexTool (IMGTEC)
  - Compressonator (AMD)
  - Mali GPU Texture Compression Tool (ARM)
  - NVIDIA DDS Plug-in (NVIDIA)
  - DirectX Texture Tool (Microsoft)
  - OPTPiX imesta （ウェブテクノロジ）

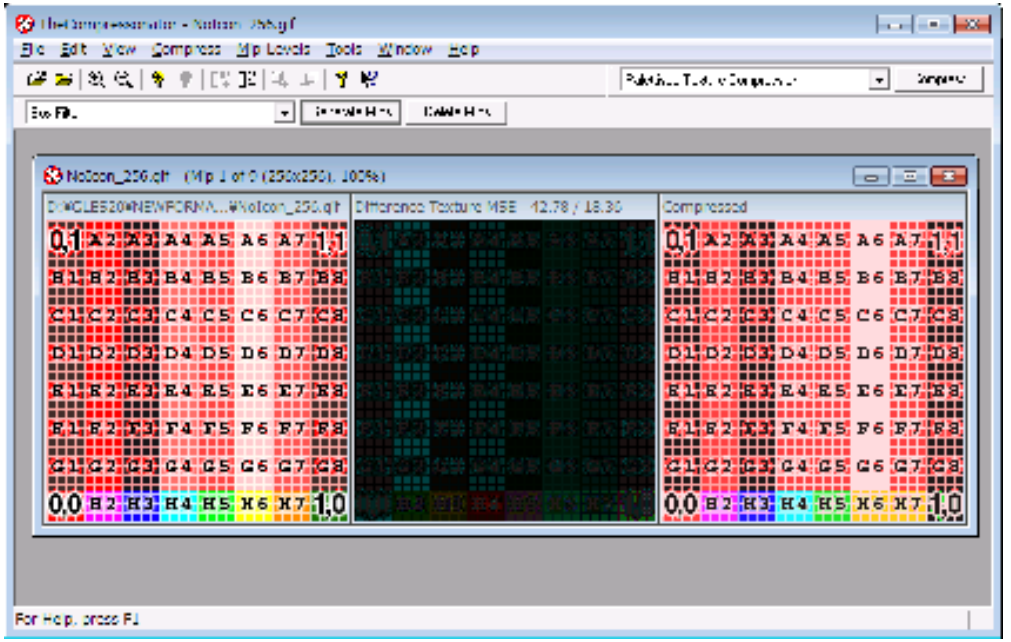
PVRTexTool (IMGTEC)

- PVRTC, ETC が作成できる
- ファイルは PVR
- 機能も充実



Compressonator（AMD）

- ETC, S3TC, ATCが作成できる
- ファイルは DDS
- パレットが扱える…が方言が強い
  - 独自 FourCC
  - RGBA が逆



DirectX Texture Tool (Microsoft)

- ファイルはDDS
- 意外に対応しているフォーマットが狭い
  - ETC, ATC不可
  - パレット不可
  - Direct3D 10形式不可（！）

NVIDIA DDS Plug-in (NVIDIA)

- Photoshop上で使えるのは大きなメリット
  - ミップマップもOK
- ファイルはDDS

- PVR, ETC, ATC, パレットなどOpenGL ESに向いた圧縮に対応していない…

マルチプラットフォーム対応

- ランタイムは DDS と PVR の両対応でだいたいカバーできる

- プレビュー
  - ETC, ATC は AMD のエミュレータで描画できる
  - PVR はエミュレータで描画できる
  - PVRは展開ライブラリも用意されている

OpenGLから無くなっているAPI

- glGet系の多くの関数
  - 自分で覚えておく
  - 標準のOpenGLでも呼び出しを回避できる
- 固定パイプラインパラメータ
  - マトリクス
  - ライトやフォグ
  - 独自に定義する
- アルファテスト

アルファテストの実装

- フラグメントシェーダで分岐
  - discard 命令
  - if (fragAlpha <= alphaRef) discard;
  - しかしはっきりと遅くなる！
    - 不透明ポリゴンのオーバードローを回避するタイルベースアーキテクチャのメリットが無くなる
  - アルファテストをするマテリアルとしないマテリアルでシェーダを分ける

    → シェーダプロファイルに追加

頂点データ

- 頂点データ
  - GL_OES_vertex_half_float
  - 16bit浮動小数点数は C/C++ 言語で直接記述できないが編集しなければ問題ない
  - 拡張なのでプラットフォームによる
  - 16bit固定小数点数とスケーリングも有効
    - テクスチャ座標
    - 場合によっては位置座標も
    - 法線は8bit×３でもいい

最適化結果（その１）

- 頂点フォーマット
  - 位置, テクスチャ座標 float → short
  - 法線 float → byte
- テクスチャ
  - PVR圧縮 （カラー系 2bpp, 法線 4bpp）
- シェーダ
  - 中精度
  - カラーと方向ベクトルは低精度

最適化結果（その２）

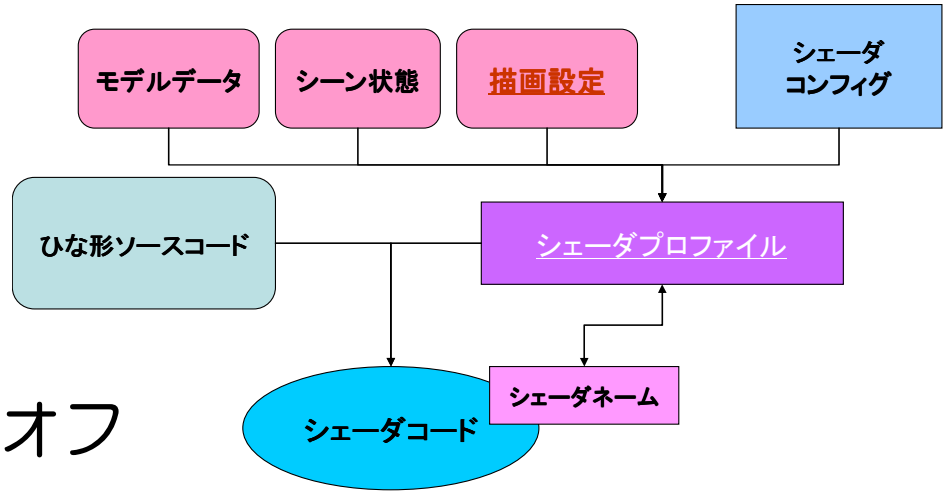
6FPS → 8FPS に！（iPad）

9FPS → 12FPS に！（iPod Touch）

えぇー？

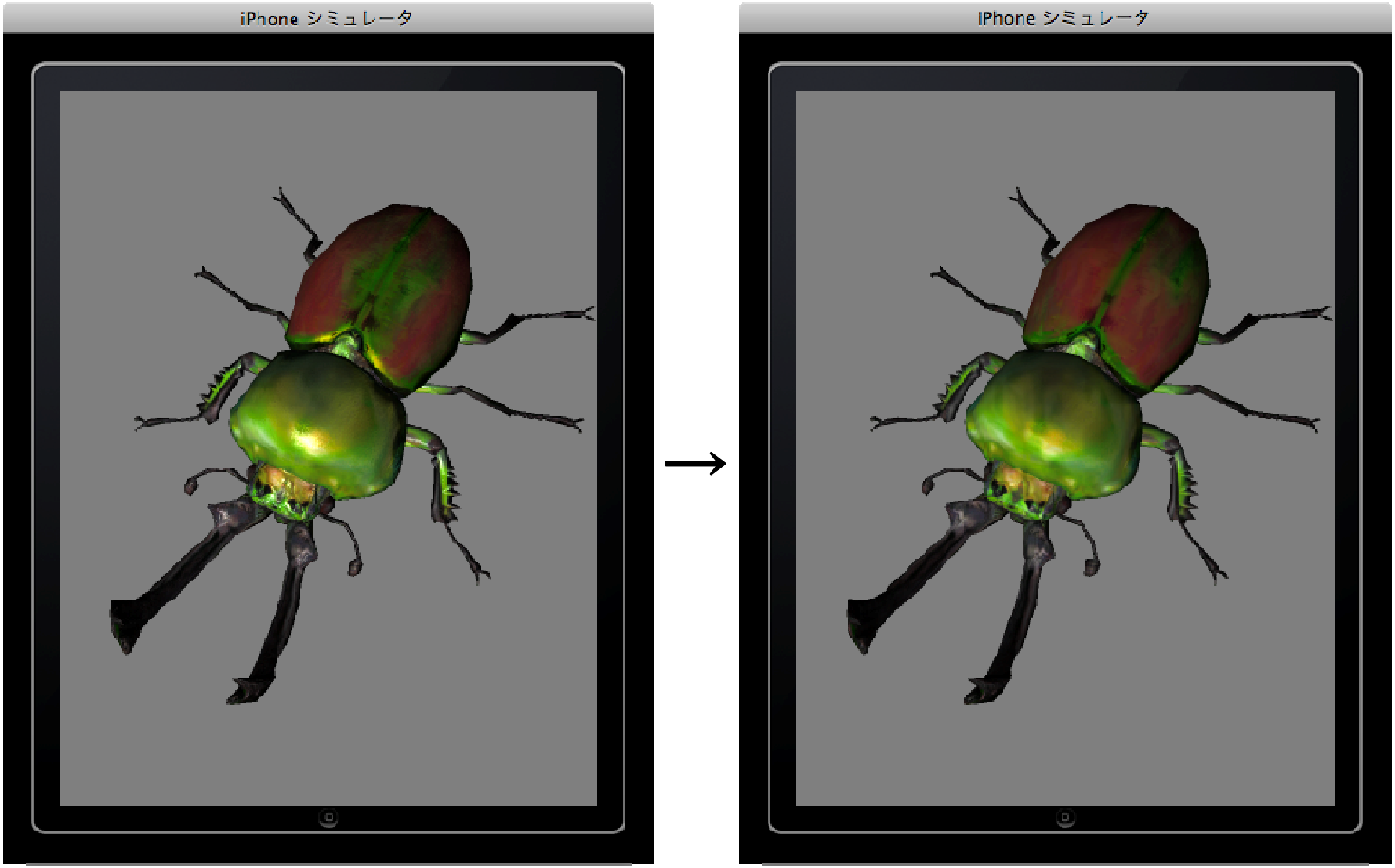
シェーディングモデルの軽量化

- シェーディングモデル
  - フラグメントライティング
  - 頂点ライティング
    - 法線マップオフ
    - 接線従法線オフ
  - なし
    - スペキュラマップもオフ
    - 法線オフ



- シェーダコンフィグでなく描画設定で

シェーディングモデルの変更（その１）



シェーディングモデルの変更（その２）

- 頂点単位
  シェーディング
  - 法線マップなし
  - スペキュラマップ
  - 環境マップ

8FPS → 30～ FPS

シェーディングモデルの変更（その３）

- シェーディングなし
  - カラーマップ
  - 環境マップ

  8FPS → 30～60 FPS

- テクスチャも個別にオンオフ
  - 環境マップもなし

    60～ FPS

アートワーク最適化

- マテリアル数を減らす
  - 第一原則
- しかし不透明とそれ以外（パンチスルー＆半透明）を分ける
  - ETC1圧縮は不透明のみ
  - アルファテストは大きな性能低下を招く

アートワーク汎用化

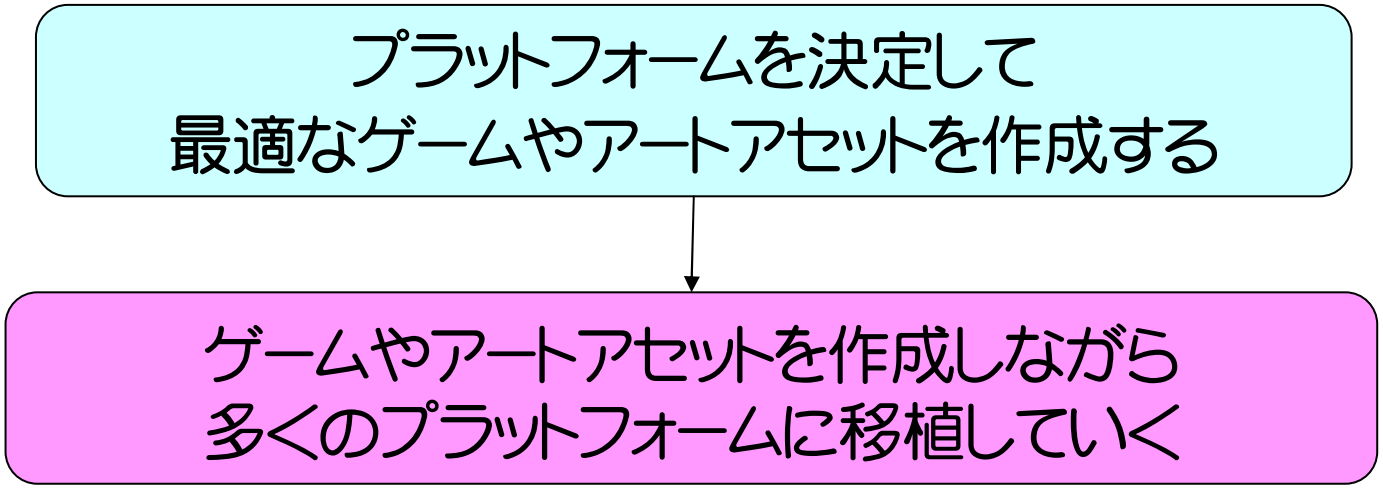
- 機能の有無に両対応
  - 法線マップなし
  - スペキュラなし
  - ライティングなし
- 低精度対応
  - 大きなものと小さなものを１つのモデルに入れない
  - テクスチャのリピートをしすぎない
- 拡張機能を使用しない
  - テクスチャサイズは２のべき乗

まとめ（その１）

- OpenGL ES2.0 はプラットフォームではない
  - ES2.0 の使用できる多くのプラットフォーム
  - しかし同時対応は可能
- スケーラビリティ
  - プログラマブルシェーダのもう１つの役割

まとめ（その２）

- 少量多品種
  - １プラットフォームで５年のコンシューマ
  - １年に５プラットフォームのOpenGL ES2.0
- ビジネスや開発ペースのギアチェンジ
  - 何が勝つかは分からない



その他の話題

- ポストエフェクト
- 描画解像度
- ミドルウェア

ポストエフェクト（その１）

- シェーダ時代の必須表現

- 苦手
  - タイルベースレンダリングのメリットが得られない

- 回数を減らす
  - バックバッファレンダリングからフロントバッファへコピーするときに

ポストエフェクト（その２）

- レンダーテクスチャの仕様にも幅がある
  - COMPLETEになる条件
    例：２のべき乗サイズ
    例：カラーとデプスのビット数を合わせる
  - iPhoneはアンチエイリアスがかけられる

描画解像度

- 解像度を下げてみる
  - 画面サイズが分かっている
  - 画面サイズが小さいと違和感は少ない
- 処理が重いかどうかを動的に判断する
  - 意外に違和感がない
  - 元々フレームレートが不安定なので可変フレームレートに対応する

ミドルウェア（PC, コンシューマ系）

- Unity
  - iPhone
  - Windows, Mac, Web, Wii
- Torque Game Engine
  - iPhone
  - Windows, Mac, Linux, Xbox360, Wii
- UnrealEngine3

ミドルウェア（モバイル系）

- Airplay
  - iPhone, Android
  - Symbian, Brew, Windows Mobile
- MascotCapsule eruption
  - iPhone, Android
  - Symbian, Brew, Windows Mobile, Java系

Q&A

連絡先：hayashih@soj.sega.co.jp