Autodesk®
Scaleform®

携帯ゲームキットの概要

本書は、Scaleform モバイルプレーヤーを使用するモバイルおよびタブレットハードウェア用にデザインさたクロスプラットフォーム、タッチ/ジェスチャーベースのゲームデモ「Scaleform 4.2 携帯ゲーム用キット」のアーキテクチャ、ソースコード、コンテンツについて説明するものです。

著者： Nate Mitchell

バージョン： 1.01

変更日： 2012 年 6 月 5 日

Autodesk®
GAMEWARE

著作権表示

Autodesk® Scaleform® 4.2

MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE REGARDING THESE MATERIALS.

How to Contact Autodesk Scaleform:

Document	Mobile Game Kit Overview
Address	Autodesk Scaleform Corporation
	6305 Ivy Lane, Suite 310
	Greenbelt, MD 20770, USA
Website	www.scaleform.com
Email	info@scaleform.com
Direct	(301) 446-3200
Fax	(301) 446-3199

目次

1 はじめに

Scaleform 携帯ゲーム用キットは、Autodesk® Scaleform® をエンジンおよびレンダラーに使用するモバイル用タッチベースのゲームビルドのベストプラクティスサンプル実装を提供します。Starforce Battlement は、Scaleform の市販用モバイルプレーヤーをクロスプラットフォームにして完全に ActionScript 3 で記述されています。実装の詳細は、本書の概要に紹介されています。このゲームはモバイルデバイス用にデザインされていますが、マウスとタッチ入力をサポートする PC、Mac、Linux、iOS、Android などのプラットフォームでプレーできます。

このキットには、多重解像度サポート、サウンド再生、デバイスへの保存とロード、成果、および iOS ゲームセンター統合などのサンプルフレームワークを含むこのゲームの完全 ActionScript 3 ソースコードおよびメインメニューが含まれています。また、ゲームに使用されたすべての Adobe® Flash®コンテンツおよびアセットも含まれます。このキットのすべてのリソースは、開発者が再利用したり、アーキテクチャのベストプラクティスサンプルとして活用したり、Scaleform を使用した独自のゲームを実装したりするために使用できます。

このゲームは、モバイルデバイス再生用 Scaleform を活用するようデザインされていますが、Adobe Flash Player を使用してウェブブラウザでも再生し、Scaleform のメモリーフットプリントまたは各種プラットフォームの性能のベンチマークとして活用することができます。

Starforce Battlement を開発および実行時間の観点から見た主要構成の簡単な内訳

1. インタラクティブなアート、アニメーション、UI レイアウト、レベル編集に Flash Professional を使用
2. ゲームのロジックは ActionScript 3 (FlashDevelop 4 をエディタに推奨) で記述
3. エクスポートした .SWF ファイルはデバイスの Scaleform で再生
4. Scaleform のレンダラーは再生結果をデバイスの GPU に渡して効果的に画面に表示

1.1 Starforce Battlement

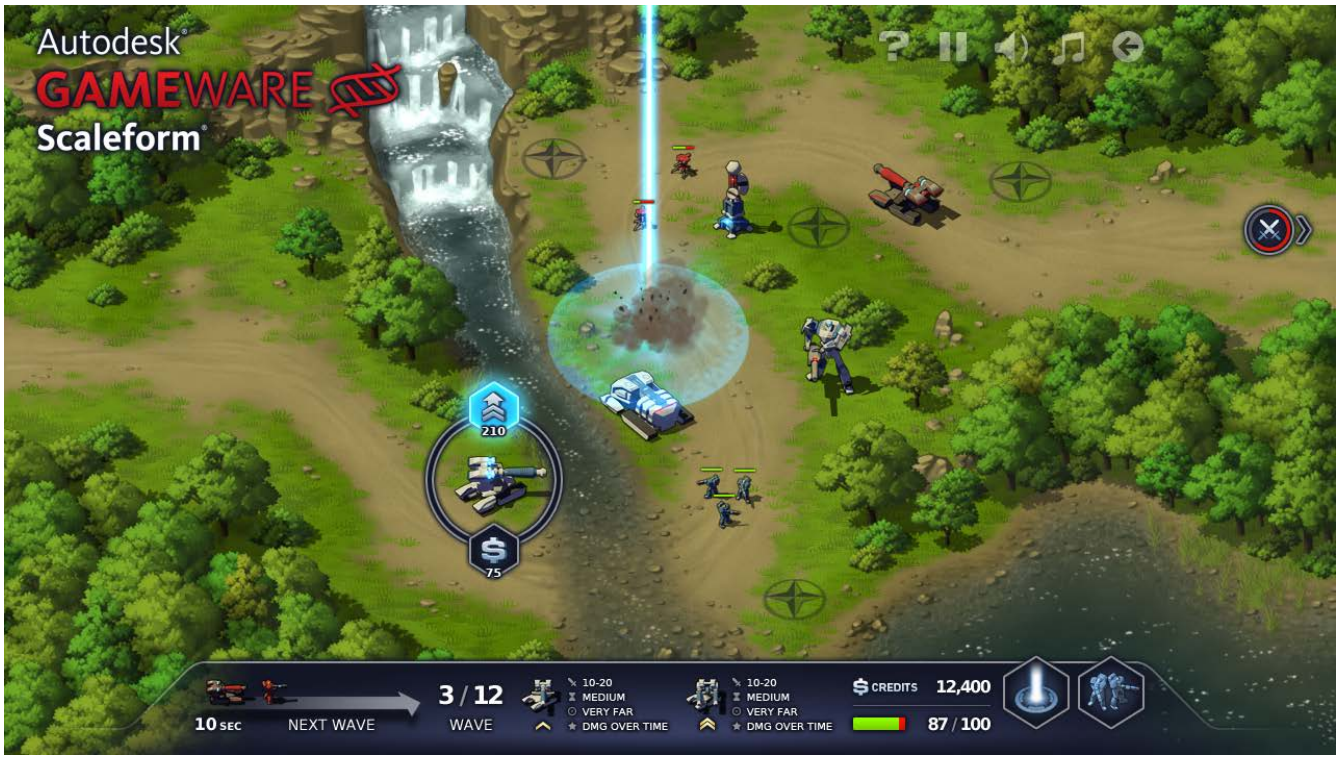

図 1：Starforce – レベル 1 のスクリーンショット

Starforce Battlement は、ロールプレイ式の防衛ゲームです。各レベルの主な目標は、タワーに攻撃してくる敵の波からタワーを防御することです。敵がが目的地（タワー）に到達するたびに、プレーヤーの命が 1 ポイント減ります。十分な敵がタワーに到達して、プレーヤーの命がゼロになるとゲームーが終了します。

プレーヤーは、地図上に場所を指定して兵隊やタワーを建て、攻撃してくる敵に反撃し、敵が目的地に到達しないよう防御できます。敵を排除すると、プレーヤーのポイントが加算され、そのポイントで新しいタワーを構築したり、タワーを強化したりできます。すべての敵を倒したところで、そのレベルを完了します。

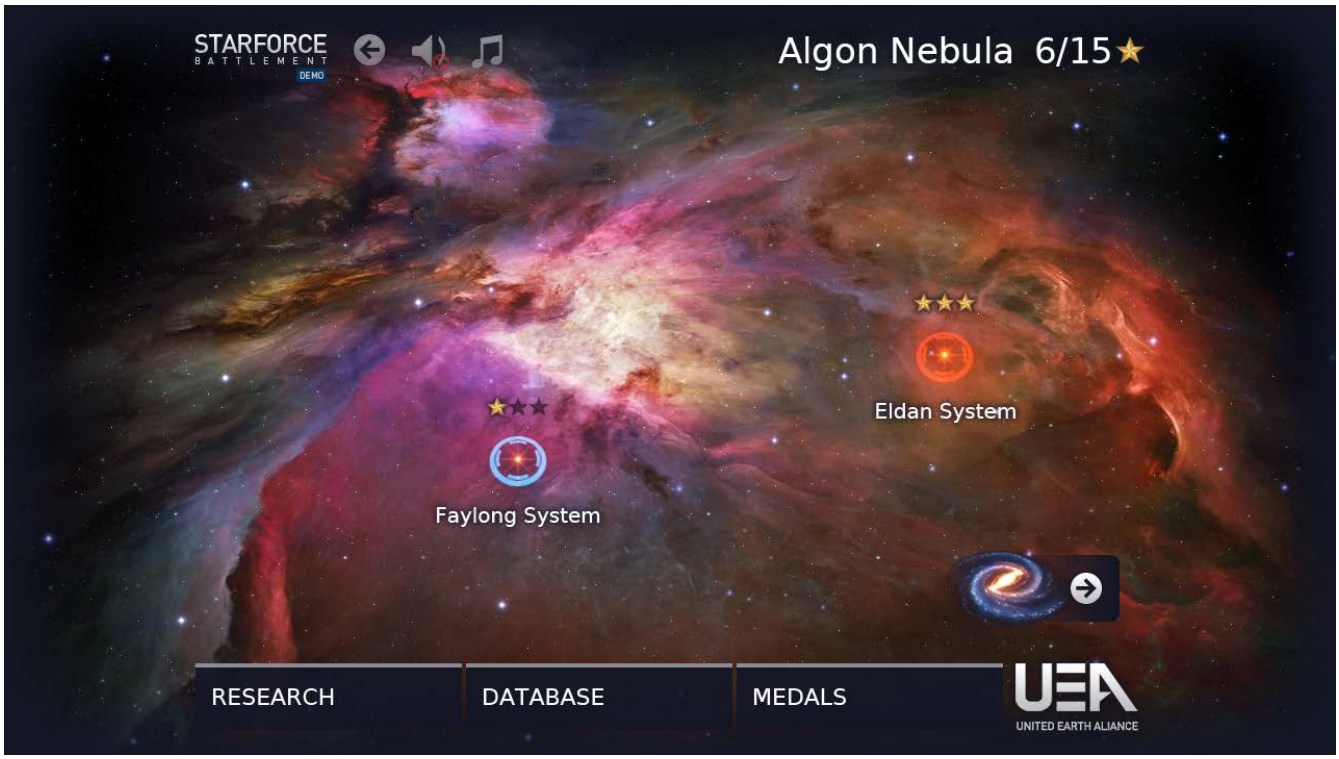

図 2 : Starforce - メインメニュー

Starforce Battlement のフルゲームデモでは、レベルを選択できるメインメニュー、チュートリアル、プレーヤーの統計/実績の表示などが装備され、すべてを開発者が自分の名前で再利用できるようになっています。

Starforce Battlement は、2012 年 5 月から iOS App Store で無料ダウンロードを開始します。開発者の方々には、iOS デバイスにダウンロードして、Scaleform が iOS でどのように機能するか試していただきたいと思っています。

1.2 機能

Starforce Battlement では、以下の Scaleform モバイル機能が使えます。

1. クロスプラットフォームモバイルサポート
2. GPU ベクターおよびビットマップレンダリング
3. ActionScript 3 完全サポート
4. マルチタッチおよびジェスチャー入力
5. データの保存とロード
6. サウンドおよび音楽再生
7. iOS ゲームセンター統合
8. オリエンテーションロック、オリエンテーションハンドリング

以下の ActionScript 3 フレームワークサンプルも提供しています。

1. プレーヤーの得点表示
2. サウンドおよび音楽再生
3. iOS ゲームセンター統合
4. データの保存とロード

2 概要

2.1 ファイルの場所とビルドについて

このデモに関連するファイルは、以下の場所に保存されます。

- *Bin/Data/AS3/Kits/StarforceTD/* - ActionScript 3 ソースコード、Flash コンテンツ、その他ゲーム（アイコンやサウンドなど）に使用されるリソース
- *Projects/Win32 /{Msvc80, Msvc90,* または *Msvc10}/GFx 4.0 SDK /* – Windows で FxPlayer Visual Studio 2005/2008/2010 を構築およびデバッグするための Visual Studio プロジェクト。
- このアプリケーションは Starforce の再生を自動設定しません。
- *Projects/iPhone/Xcode4/GFx 4.0 iPhone SDK* – iOS の市販用モバイルプレーヤー用 Xcode 4 プロジェクトこのアプリケーションは Starforce の再生を自動設定しません。
- *LocalApps¥StarforceTD¥Android* – 初めて Scaleform を構築した際に作られるが、このディレクトリには、市販用モバイルプレーヤー用に正しく設定された Eclipse プロジェクトが Android の StarforceTD の再生用に予め設定されている。

Windows 用に構築されているデモ版「StarforceTD.exe」は、*Bin/Data/AS3/Kits/ StarforceTD* に保存されます。これは、Windows のスタートメニューから、または Scaleform SDK ブラウザからもアクセス可能です。

FxPlayer または FxMobilePlayer アプリケーション用プロジェクトおよびソリューションは、携帯ゲーム用キットを別のプラットフォームでコンパイル、展開、実行するために使用されます。

Windows の場合、デバッグ用の「作業ディレクトリ」が *Bin/Data/AS3/Kits/StarforceTD* に設定され、そのコマンドライン属性は、「*StarforceTD.swf*」に設定されています。

iOS の場合、Xcode でデバイス用のアプリケーションを構築および展開した後、SFW ファイルおよびサブディレクトリ（オーディオ、アイコン）を iTunes を使用してアプリケーションにコピーします。

2.2 デモの使用

2.2.1メインメニュー

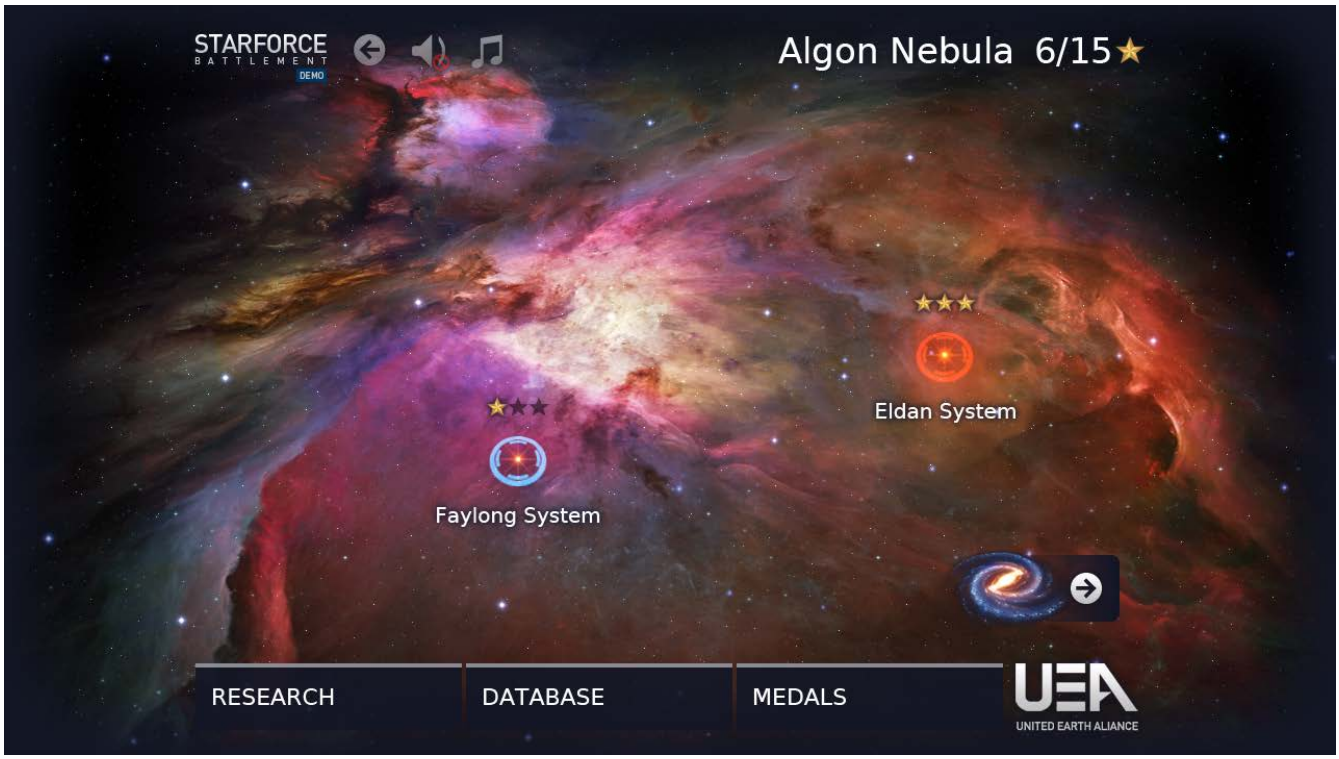

図 3 : Starforce - メインメニュー

StarforceTD.as はゲームへのエントリーポイント（入口）です。これは、StarforceTD.fla のドキュメントクラスで、初期化すると、MainMainState を初期化し、StarforceMenu.swf をロードします。StarforceMenu.swf にはメインメニューのアートアセットとメインメニューを動かす ActionScript 3 クラス定義が含まれています。

メインメニューでは、基本的にミッションを選択ます。また、プレーヤーのテクノロジーツリーや表示設定を変更するオプション、プレーヤーの得点履歴や成績を見ることができます。メインメニューの左上にあるボタンは、プレーヤーが音楽やサウンド効果を切り替えたり、ゲームを終了するためのボタンです。

レベル選択は、ユーザーが選択可能な銀河/星雲マップにインジケータを表示します。各星雲には、線形矢印にロックされていないミッションデータが表示されます。プレーヤーは、マップを 2D スペースにドラッグして、ミッションを選択する前にオプションを見ることができます。

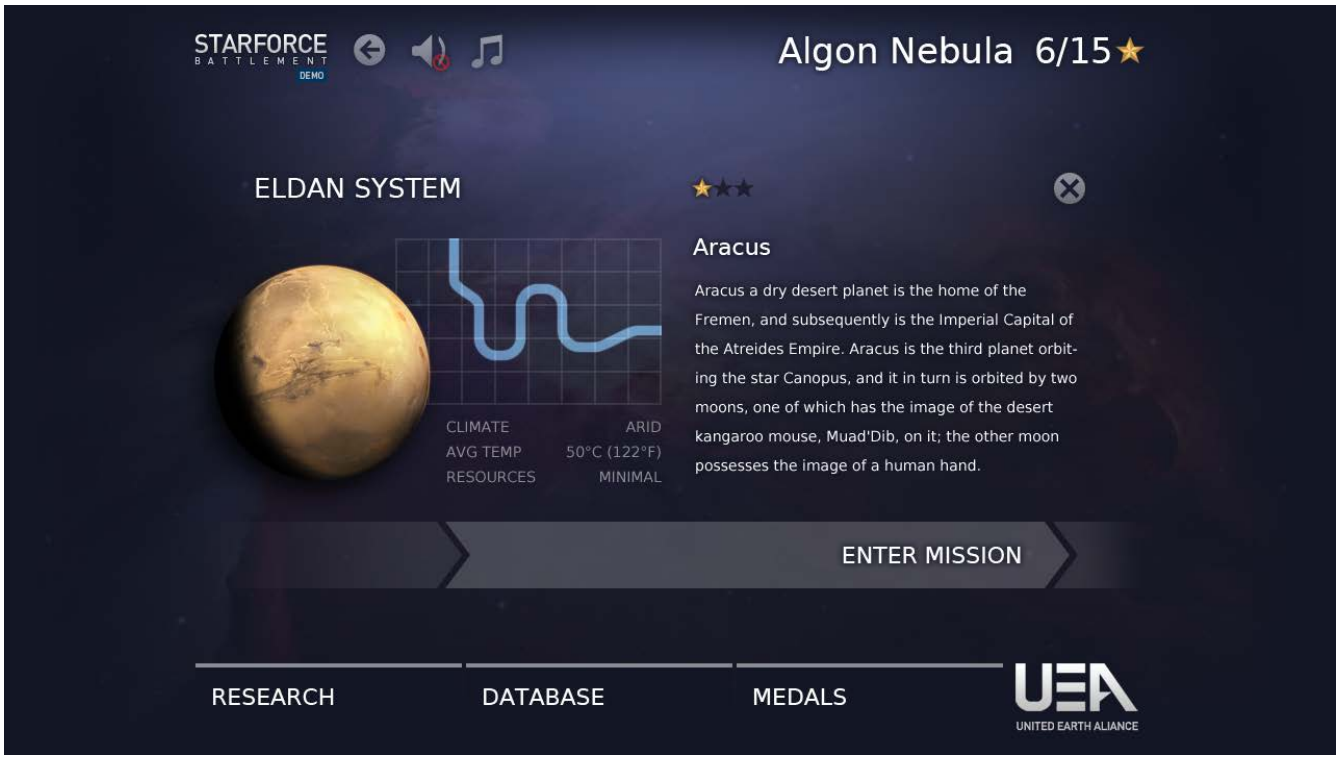

図 4 : Starforce – ミッションデータ

プレーヤーがミッションを選択すると、ミッションデータビュー（MissionDataView.as）が表示されます。ミッションデータビューではミッションが詳しく説明されています。このビューから、プレーヤーは、「Enter Mission」ボタンを押してゲームを開始するか、画面右上の「X」ボタンを選択してミッション選択ビューに戻ることができます。

2.2.2ゲーム開始と HUD

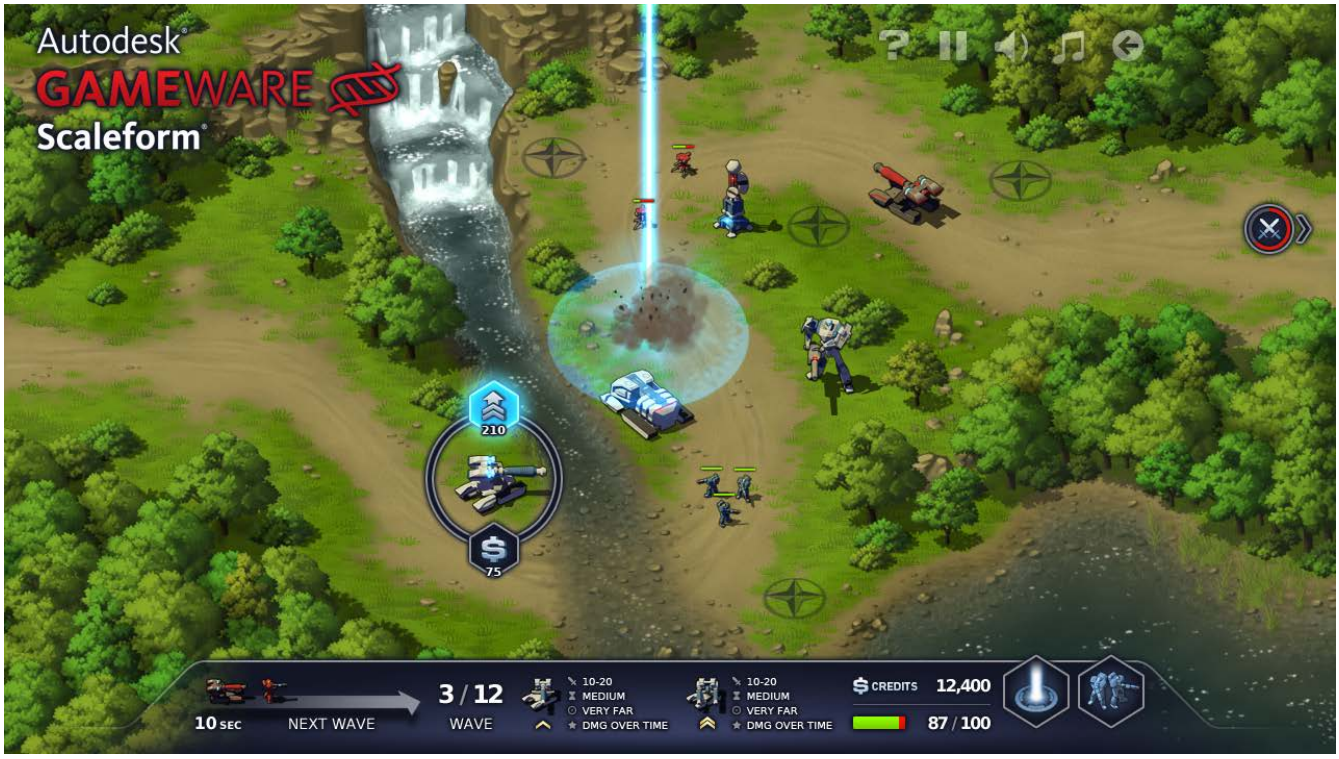

図 5 : Starforce – レベル 1 のゲーム画面と UI

Starforce Battlement のレベルとゲーム本体は StarforceTD.swf にパックされています。ゲームのレベルに関するすべてのデータは、StarforceTD.swf を初めて実行したときにメモリーにロードされますが、レベルは LevelGameState を作成するまで初期化されません。ゲームはそれぞれ独特な 5 つのミッション/レベルに分かれています。

最初の敵の攻撃(ファーストウェーブ)は「Rush Wave」ボタン（上のスクリーンショット右側の交差する剣のアイコン）をクリックするまで開始されません。最初のウェーブが始まると、すべての敵を倒すか、プレーヤーが負けるまで各レベルで設定されている間隔で敵が次々と現れます。現在のウェーブの回数とそのレベルに設定されているウェーブの数は、HUD の左隅のボタンに表示されます。ここでは、次のウェーブが来るタイミングも見ることができます。

プレーヤーがすべての敵を倒すと、タワーに到達した敵の数によってプレーヤーに 1～3 個の星で評価がもらえます。この星は、プレーヤーのテクノロジーツリーの展開に使えます。

タワーを建てる場所は、黒い丸の中にダイアモンドが付いた印が付いています。これは、すべてのレベルで散らばっています。選択すると、その場所の上にメニューが表示され、新しいタワーの建設を選択できます。同じ場所には 1 つのタワーしか建てることができません。タワーは空いている有効な場所にしか建てることができません。タワーの建設には十分なクレジットが必要です。プレーヤーの現在のクレジットは、HUD の右側（プレーヤーのエネルギー表示の上）に表示されています。

タワーが完成すると、最大 2 倍までアップグレードできます。現在のタワーレベルと次のレベルは、タワーを選択すると、HUD の中央に表示されます。

プレーヤーには、「Orbital Strike」と「Deploy Mercenaries」のアビリティが与えられています。このアビリティは、HUD の右側にあるそれぞれのアビリティボタンをクリックしていつでも使えます。この機能を有効にしたらと、プレーヤーは、機能を起動または誘導する場所をマップ上に選択します。Orbital Strike は、マウスまたはタッチでインプットを押すまたはドラッグして発射後にドラッグすることができます。アビリティを使った後は、一定時間の「クールダウン」が必要です。クールダウン中は使えません。クールダウンの時間は、アビリティボタンが暗い色になって表示され、時間経過と共に明るくなります。

HUD の右上にある一連のボタンでは、ユーザーがチュートリアルを見たり、ゲームをポーズ/再開したり、サウンド効果の切り替え、音楽の切り替え、現在のミッションのやり直し/終了などが行えます。

3 アーキテクチャ

3.1 Flash ファイル

このゲームのアートアセットの大部分は、Flash です。FLA ファイルには、レベル、エンティティ、UI、および大部分のゲームアニメーションが含まれてます。Starforce Battlement は、兵隊およびタワーのモデルは 3D で作られているため主にビットマップを使用しています。

ゲームのアート、UI 用のアイコンのサブセットは、FLS ファイルの外に保存されています。これらの外部画像は、実行時に ActionScript でロードされます。

Starforce Battlement は以下の 3 つの FLS ファイルで構成されています。

1. **StarforceTD.fla** – ゲームの中核となる.FLA ファイル。このファイルには、ActionScript コードと Starforce Battlement の全レベル、UI、エンティティのアートアセットが含まれています。また、メインメニューやローディング画面を適切にロード/アンロードすることでゲームを管理する役割も担っています。
2. **StarforceMenu.fla** – このファイルには、ActionScript コードとメインメニューおよびミッション選択のアートアセットが含まれています。StarforceTD.swf と一部の UI クラス定義を共有していますが、内蔵型のため、スタンドアローンでテストすることが可能です。
3. **LoadingView.swf** – このファイルには ActionScript コードと、ゲーム中に表示されるロード画面のアートアセットが含まれています。

3.2 ActionScript コード

3.2.1実装サマリー

Starforce Battlement のゲームプレイおよび UI のロジックは、すべて ActionScript 3 で記述されています。このため、ゲームプレイコードは移動性が高く、Starforce は iOS ゲームセンターとの統合以外に Starforce デバイスには依存していません。

3.2.1.1 エントリーポイント

ゲームのエントリーポイントは、StarforceTD.as クラスで、Flash ドキュメントクラスプロパティを使って StarforceTD.swf と関連しています。ドキュメントクラスに関する詳細は、Adobe のオンライン Flash 説明書を参照してください。

3.2.1.2 ゲームステータス

このゲームは、3 つの「GameStates」に分けられています。それぞれがゲームの異なる状態を表しています。一度にアクティブにできる GameStates は 1 つで、アクティブな GameStates は StarforceTD クラスが保有管理します。3 つのゲームステータスは以下のとおりです。

1. MenuGameState – メインメニューの GameState
2. LoadingGameState – ロード画面の GameState
3. LevelGameState – レベルとゲームプレイの GameState

StarforceTD のコントラクターはサブシステム（データ、サウンド、マルチタッチ）を初期化し、StarforceMenu.swf MainMenuView.as が定義するメインメニューをロードします。

3.2.1.3 メインメニューの実装

メインメニューは最初 com.scaleform.std.menu パッケージが実装します。（パッケージの詳細は下を参照）メインメニューは、ActionScript 3 のネイティブイベントシステムでイベントがさまざまなメインメニュービュー間に渡されるイベントドリブンアーキテクチャの上に構築されます。ユーザーの選択に従って、イベントをディスパッチし、処理して、ウィンドウの開閉、データの読み込み、全く新しいステータス（ミッションの開始など）へのトランジションを行います。

ミッションを開始すると、GameEvent がメインメニューからディスパッチされます。
StarforceTD.as がこのイベントを聞き、LoadingGameState のロード、MenuGameState のアンロード、LevelGameState のロードを実施して即座にレベルに移行します。
LevelGameState は選択したレベルをロードして、レベルのロードが完了するとすぐにゲームを開始します。

LoadingGameState は、描かれるレベルコンテンツはすでに StarforceTD.swf のメモリにロードされているため、技術的な観点から見ると必要ありませんが、ここでは LoadingGameState トランジションを使用してレベルと HUD の初期化をユーザーに見えないようにして、ゲームへの移行をスムーズにしています。

3.2.1.4 ゲームプレイの実装

Level.as はゲームの実装の中核部分です。このクラスは、すべてのゲームレベルのベースクラスとして機能します。Level.as クラスは、レベル、HUD、敵、タワー、ユーザーアビリティを含むゲームを構成するすべてのエレメントを初期化および管理します。Level はすべてのレベルのベースクラスです。各レベルには、独自のサブクラスがあり、敵を定義したり、その他の行動および変数をオーバーライドしてレベルをより独特にすることができます。（金の量、ウェーブの間隔、敵のタイプ、ウェーブの数などを変える）

Level は、すべてのエンティティの製造、破壊、管理を担います。通常、ゲームのエンティティは、タワーと敵です。すべてのエンティティはレベル別に行動するため更新する必要があります。（地図上を動く、敵を探す、攻撃するなど）

Flash Professional は、ActionScript クラスを画像、アニメーション、ActionScript で構成される記号と簡単にバインドできます。これらの記号は、実行時に ActionScript から参照およびインスタント化できます。Starforce Battlement はこの方法でゲームおよびメニューシステムの画像コンテンツのほとんどをインスタント化しています。記号を ActionScript にバインドする、または実行時にインスタンスを作ることに関する詳細は、Adobe の Flash 説明書を参照してください。

たとえば、TowerMech 記号は StarforceTD.fla の Flash ライブラリにあります。TowerMech 記号のプロパティでは、この記号が com.scaleform.std.entities.TowerMech ActionScript クラスにバインドされていることが分かります。このクラスが ActionScript でインスタント化されると、この画像記号のインスタンスが作られ、コードで操作することができます。

Level.as クラスは、イベントドリブンフレームワークでゲームを管理します。このクラスは、エンティティ、UI、またはその他のシステムのフレームワーク（サウンド、データ、アチーブメント）からディスパッチされるイベントを聞き、それに合わせてゲームと HUD を更新します。たとえば、タワーが兵隊を攻撃するとき、攻撃している兵隊のヘルスゲージを直接変更するのではなく、タワーが AttackEvent をディスパッチします。Level は、AttackEvents をグローバルに聞き、AttackEvent を受信して、アタッカーとディフェンダーを処理し、ディフェンダーが受けたダメージを計算して、ダメージをディフェンダーに適用します。

ほとんどの相互運用は、さまざまなクラスを要約する方法でレベルを通じてゲームエンティティ間で渡されます。これには、エンティティのスコープ以外のデータも計算されます。（プレーヤーのテクニカルツリーはディフェンダーのダメージ量に影響される場合があるなど）このデザインパターンは、冗長コードを取り除き、エンティティクラスを独立させるためゲーム全体の変更を簡単にします。

3.2.1.5 ゲームデータ

すべてのタワー、ミッション、アビリティ、敵に対する統計を含むゲームのデータの大部分は、com.scaleform.std.data パッケージで定義されます。このデータは、GameData 静的型クラスを使って実装からアクセスできます。これにより、外部変数が影響を与えている場合に、GameData クラスでデータを戻してクラスを要求する前にデータを変更できるようになります。（たとえば、プレーヤーのテクノロジーツリーが普通のタワー攻撃で変化した場合など）

ゲームのデフォルトデータは、ActionScript に保存せず XML または JSON などの外部のデータ定義フォーマットに保存して実行時にロードするのが理想です。残念ながら、XML は Scaleform 4.0 をサポートしていないため、最も速い方法という点で、クラス定義にデータを保存する方法が採用されています。これは、将来的にベストプラクティスを反映して変更される可能性があります。

3.2.1.6 インプット処理

ゲームのインプットは、ActionScript イベントリスナーでスクリプトされています。これにより、インプットロジックがクロスプラットフォームで機能し、Scaleform Player がプラットフォーム全体のマウス/タッチのインプットを要約して、ActionScript に渡します。

ゲームは、ネイティブ Scaleform アプリケーションがタッチインプットをサポートしているかどうかを判断してマウスまたはタッチのインプットを聞きます。FxPlayer が正しく設定されていると仮定して、アプリケーションは、MultitouchInputState が MovieView にインストールされているかどうか、またはタッチに関連した適切なオペレーティングシステム機能が利用可能かどうかに基づいて、実行時にタッチがサポートされているかを識別します。

Starforce のインタラクティブエレメントには、TouchEvents と MouseEvents の双方のロジックがありますが、アプリケーションの設定に基づいていずれか片方のセットしか使用することはできません。TouchEvents と MouseEvents のタイプは異なりますが、双方とも良く似たタイプを多く持って

います。たとえば、TouchEvent.TOUCH_TAP は、MouseEvent.MOUSE_CLICK と同様に使用できます。また、TouchEvent.PRESS は MouseEvent.MOUSE_DOWN と同様に使用できます。

TouchEvents と MouseEvents の双方で 1 つの機能宣言をリスナーとして使用するには、機能のパラメータを必ず Event タイプにします。*target* などのイベントに詳しい情報が必要である、またはインプットイベントを操作スル場合は、イベントを正しくキャストする必要があります。実行時にタイプを識別する一般的な方法のひとつは、「*is*」演算子を使うことです。（以下参照）

```
protected function onSetRallyClick( e:Event ):void {
    var point:Point;
    if (e is MouseEvent) {
        var me:MouseEvent = e as MouseEvent;
        point = new Point( me.stageX, me.stageY );
    }
    else {
        var te:TouchEvent = e as TouchEvent;
        point = new Point( te.stageX, te.stageY );
    }
}
```

3.2.2ActionScript パッケージ

ゲーム用の ActionScript コードは、クラスの機能および目的に基づいて ActionScript パッケージに区分されています。

以下のパッケージはすべて「com.scaleform.std」の前に置きます。

1. **abilities** – プレーヤーのアビリティ (Orbital Strike, Hire Mercenaries) のクラス定義およびゲームプレイロジック
2. **controls** – 再利用する UI エレメントおよびコンポーネントのクラス定義
3. **core** – ステートフローおよびデータ書込み/読出しを行う基本コンポーネントのクラス定義
4. **data** – プレーヤー、エンティティ、レベル、アビリティを含むゲームのデータ定義

5. **entities** – プレーヤー、エンティティ、レベル、アビリティを含むゲームのデータ定義
6. **events**– すべてのゲーム、UI、ステートイベントのイベント定義
7. **fx** – ゲームの特殊効果（爆発や戦闘のアニメーション）のクラス定義
8. **hud**– 内蔵 HUD およびメニューのクラス定義
9. **levels** – 各レベルのクラス定義。Level クラスはゲームのプライマリクラスの 1 つです。
10. **loading** – ロード画面のクラス定義
11. **menu** – すべての UI エレメント、ミッション選択、インプット処理を含むメインメニューのクラス定義
12. **system** – サウンド再生、保存/ローディング、オリエンテーション処理などのタスクが ActionScript と相互作用するためのフレームワーク
13. **utils** – コードベース全体で使われるユーティリティクラス、機能、定義。特定の派 k-時に限定されない。

3.2.3フレームワーク

3.2.3.1 保存/ロードフレームワーク

com.scaleform.std.system.SaveSystem.as に定義される保存およびロードのフレームワークは、ディスクにデータを保管して読み出すために実装するサンプルです。プレーヤーの進捗を保存するために使うコードは単純ですが、より複雑な実装のための基盤となるものです。

この機能のための ActionScript インタフェースには、SharedObject クラスを使います。
SharedObject は、開発者がディスクの ActionScript オブジェクトにデータを保管して読み出せるようにする ActionScript 3 の標準クラスです。Scaleform の SharedObject 実装クラスは、プリミティブ型と配列をサポートしています。

各 SharedObject は、実行時に書き込み/読み出しに使用する固有のストリング識別子で定義されています。下のコードでは、SharedObject 固有のストリング識別子は「sb_player_data」です。

```
// Local reference to the galaxy progress. Populated from the PlayerData.
public static var galaxyProgress:Array = null;
// Constant String for the property of the SharedObject we'll modify.
protected static const GALAXY_PROGRESS_SO_PROP:String = "galaxyProgress";
// Reference to the SharedObject once we retrieve it.
protected static var _playerDataSO:SharedObject = null;

public static function load( pd:PlayerData ):void {
    _playerDataSO = SharedObject.getLocal("sb_player_data");
    galaxyProgress = _playerDataSO.data[ GALAXY_PROGRESS_SO_PROP ];
    pd.setGalaxyProgress( galaxyProgress );
}

public static function save( pd:PlayerData ):void {
    // Write PlayerData into SharedObject.
    _playerDataSO = SharedObject.getLocal("sb_player_data");
    galaxyProgress = pd.getGalaxyProgress();
    _playerDataSO.data[ GALAXY_PROGRESS_SO_PROP ] = galaxyProgress;
    _playerDataSO.flush();
}
```

開発者は、1 つの SharedObject インスタンスに制限されているわけではありません。そえぞれに固有の名前が付けられていれば、複数の SharedObjects をディスクに保管することができます。こすることで、セグメントデータをカテゴリに分けることができるため便利です。

SharedObject クラスに関する詳細は、以下のサイトを参照してください
http://help.adobe.com/en_US/FlashPlatform/reference/actionscript/3/flash/net/SharedObject.html

3.2.3.2 サウンドフレームワーク

com.scaleform.std.system.SoundSystem に定義されるサウンドフレームワークは、ゲームのサウンド効果と音楽再生を処理する実装サンプルです。フレームワークは、ゲームからディスパッチされた SoundEvents に基づいてディスクから外部サウンド（この場合 MP3）をロードして再生します。SoundEvents は、ステージを参照するクラスからディスパッチできます。

```
dispatchEvent( new SoundEvent( SoundEvent.PLAY_SOUND, true, true,
"infantry_attack.mp3",
                               1, false ) );
```

サウンドフレームワークは 2 つのサウンドチャンネル（背景と前景）を使用します。SoundEvents は、再生する対象サウンドのチャンネルを指定します。ゲームのサウンド効果はすべて前景チャンネルで再生されます。BGM は背景チャンネルで再生されます。これにより、SoundEvent を使ってそれぞれを別々にミュートしたり、切り替えたりできます。

```
dispatchEvent( new SoundEvent( SoundEvent.MUTE_SOUNDFX, true, true );
```