

インテリジェント配信

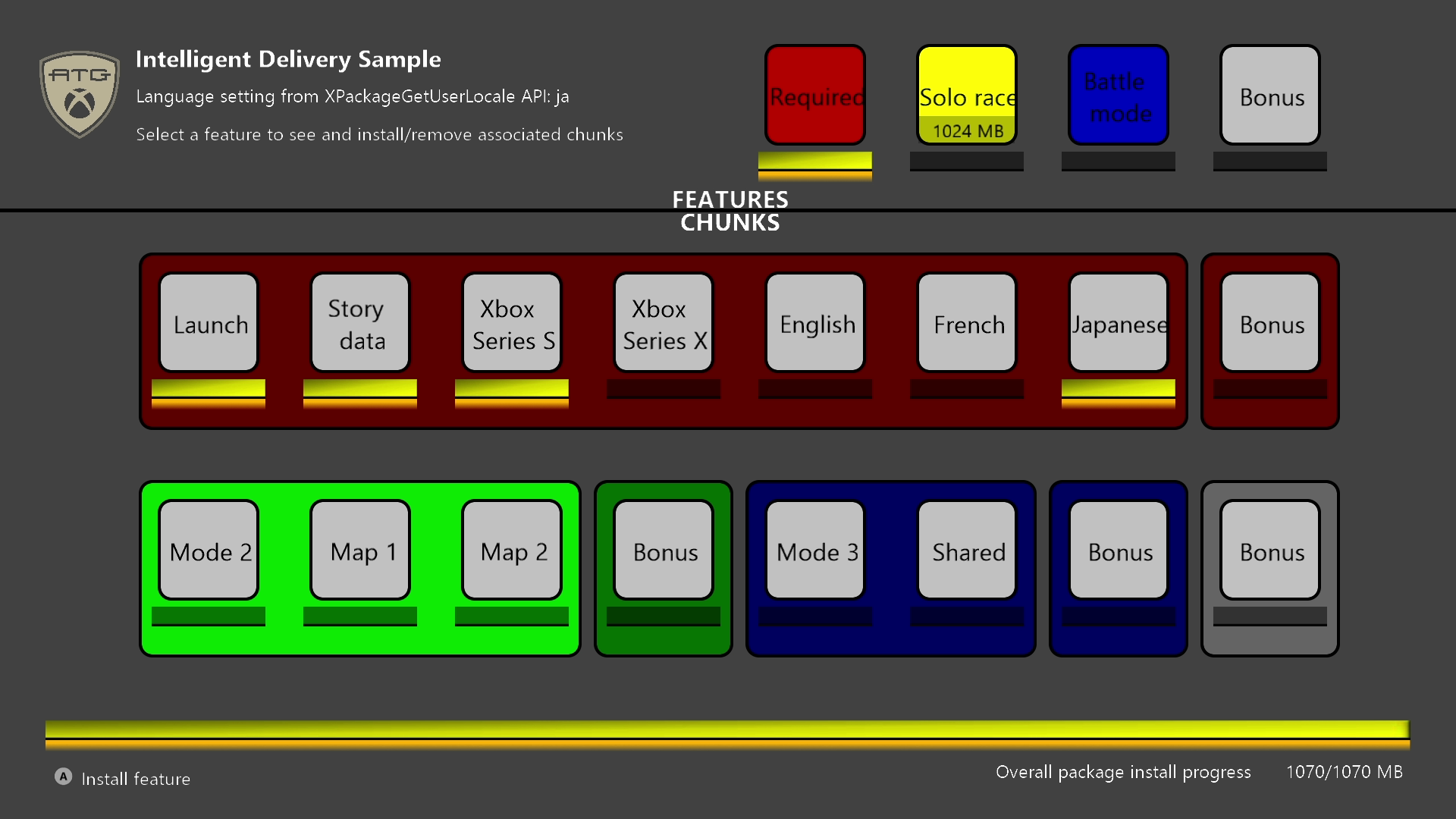
*(旧 StreamingInstall)*

*このサンプルは Microsoft Game Development Kit (2020 年 8 月) と互換性があります。*

# 説明

このサンプルでは、インテリジェント配信 **API** をデモンストレーションします。起動チャンクのインストールを完了すると、サンプルを実行して残りのチャンクのインストールの進行状況を視覚化できます。

また、Feature (機能) と Recipe (レシピ)、ローカライズ済みチャンク、ハードウェア固有のチャンクなどのインテリジェント インストール機能についても説明します。ターゲット ハードウェアのターゲット SKU (Xbox Series ファミリーの Lockhart/Anaconda、Xbox One ファミリーの Durango/Xbox One X)、デバイスの言語/ローカル設定に応じて、異なるコンテンツがインストールされます。その後、ユーザーはサンプル実行中にインストールまたは削除する複数の Feature を選択して、インストール結果と進捗状況の結果を確認できます。



このサンプルは、Recipe と Feature の機能の動作を紹介した旧 *StreamingInstall* サンプルをもとに構築されています。レイアウト ファイル (Chunks\_[XboxOne/Scarlett].xml) は下記のようになっています:

<Recipes>

<Recipe Id="default" IncludedFeatures="feature1" />

</Recipes>

<Features>

<Feature Id="feature1" Hidden="true" Tags="mode1" />

<Feature Id="feature2" DisplayName="Mode 2" Tags="mode2;maps"/>

<Feature Id="feature3" DisplayName="Mode 3" Tags="mode3"/>

<Feature Id="feature4" DisplayName="Bonus Content" Tags="bonus"/>

</Features>

パッケージは、feature1 のみをインストールする既定 (default) のアクティブなレシピでビルドされます。インストールできる合計 4 つの Feature があり、通常グリッド内では異なる色のアイテムで表示されます。各 Feature をインストールすると、それらのタグがアクティブにマークされ、それらのタグを使用するチャンクがインストールされます。サンプルで Feature を選択すると、Feature の定義でタグ付けされているその Feature に関連付けられたチャンクがインストールまたは削除されます。

さらに複雑な面がいくつかあります:

<Feature Id="feature2" DisplayName="Mode 2" Tags="mode2;maps"/>

feature2 では、mode2 または map でタグ付けされたチャンクがインストールされます。

<Chunk Id="1007" Tags="mode1#bonus">

<FileGroup DestinationPath="\" SourcePath="." Include="filler256mb7.txt"/>

</Chunk>

このチャンクは、mode1 とbonus タグの**両方**がアクティブである場合にのみインストールされます。つまり、feature1 と feature4 の**両方**がインストールされているときです。これらがサンプルに表示されるすべての"Bonus"アイテムで、各機能に 1 つのアイテムが、feature4 にはスタンドアロンのアイテムがあります。

<Chunk Id="3001" Tags="mode3;mode2">

<FileGroup DestinationPath="\" SourcePath="." Include="filler256mb13.txt"/>

</Chunk>

このチャンクは、mode1 **または**mode2 がアクティブな場合にインストールされます。つまり、feature2 **または**feature3 のうちどちらかが最初にインストールされているときにチャンクがインストールされることになります。逆に、両方の機能が削除されている場合にのみ削除できます。これは、サンプルでは "Battle Mode" 機能にグループ化された "Shared" アイテムとして表示されます。

**PC** に対しては、デバイス固有のチャンクは存在しないためこれら Feature とチャンクは少し変更されます。代わりに、feature4 に依存する feature1 があります。

# サンプルのビルド

Scarlett 開発キットを使用している場合は、アクティブなソリューション プラットフォームを Gaming.Xbox.Scarlett.x64 に設定します。

Xbox One 開発キットを使用している場合は、アクティブなソリューション プラットフォームを Gaming.Xbox.XboxOne.x64 に設定します。

PC を使用している場合は、アクティブ なソリューション プラットフォームを Gaming.Desktop.x64 に設定します。

*詳細については、GDK のドキュメントの*「サンプルの実行」*を参照してください。*

# サンプルの使用方法

ほとんどの GDK サンプルとは異なり、*IntelligentDelivery* サンプルは Visual Studio からのビルド、配置には向いていません。Visual Studio から直接配置して実行することができますが、このサンプルでデモンストレーションされるインストール ツールとその処理が一時停止します。このサンプルを使用する手順は次のとおりです。

1. **配置、実行を行わずに**、Visual Studio からサンプルをビルド
2. インストール パッケージのコンテンツを生成
3. インストール パッケージを作成
4. パッケージをインストールしてサンプルを実行

このサンプルでは、コンソール デバイス ファミリごとに個別のパッケージが必要です。異なる .bat ファイルと .config ファイルが用意されており、デバイスごとに適切なパッケージを作成するために使用できます。どのプラットフォームに対しても XPackage コードは GDK で変更されません。*MicrosoftGame.config* ファイルにおいて唯一異なる属性が TargetDeviceFamily であり、これは XboxOne、Scarlett または PC となります。

最初の手順は、Visual Studio からサンプルをビルドするだけです。これによりバイナリとディレクトリ レイアウトが作成され、最終的にパッケージの作成に使用できます。サンプルがビルドされたら、コマンドラインから **CreateInstallPackage\_[XboxOne/Scarlett/PC].bat** を実行します。このバッチ ファイルでは、インストールの進捗状況をリアルタイムに監視するためのコンテンツ ファイルとして機能する複数の大きなファイルの作成を行い、TargetDeviceFamily セットを使用して適切な MicrosoftGame.config をステージングします。

次の手順では、GDK と一緒にインストールされている **makepkg** ツールを使用してインストール パッケージを作成します。**Xbox ゲーム コマンド プロンプト**を開き *IntelligentDelivery.vcxproj* を含むディレクトリに移動します。 開いたコマンド プロンプトから **CreateXVC\_[XboxOne/Scarlett].bat** または **CreateMSIXVC\_PC.bat** を実行するか、直接コマンドを実行できます。

Xbox One デバイス ファミリ:

makepkg.exe pack /v /f Chunks\_XboxOne.xml /d ".\Gaming.Xbox.x64\Layout\Image\Loose" /pd ".\Gaming.Xbox.x64\Layout\Image"

Scarlett デバイス ファミリ:

makepkg.exe pack /v /f Chunks\_Scarlett.xml /d ".\Gaming.Scarlett.x64\Layout\Image\Loose" /pd ".\Gaming.Scarlett.x64\Layout\Image"

PC:

makepkg.exe pack /pc /v /f Chunks\_PC.xml /d .\Gaming.Desktop.x64\Layout\Image\Loose /pd .\Gaming.Desktop.x64\Layout\Image

コマンドを直接実行する場合は、.bat を参照して、すべてのファイルが適切な場所にステージされていること (特に一部が更新された場合) を確認します。

パッケージには容量を埋めるためのいくつかのギガバイトのコンテンツが含まれているため、実行には数分かかる場合があります。実行が完了すると **/pd** オプションの後に指定されている場所に、いくつかのファイルが作成されます。この場合では、 **“.\Gaming.[Xbox/Scarlett/PC].x64\Layout\Image”** ディレクトリとなります。実際のインストール パッケージ ファイルにはパッケージの**パッケージ ファミリ名** が含まれる名前が用いられ、これは *MicrosoftGame.config* に由来します。コンソールの場合、**.xvc** ファイルは TargetDeviceFamily によるサフィックスを用いて作成されます (Xbox One の場合は**\_x**、Scarlett の場合は**\_xs**)。PC の場合、**.msixvc** ファイルが作成されます。

最後の手順では、パッケージをデバイスにインストールします。**Xbox** の場合、テストするシナリオに応じて、インストールするコンテンツを決定できるオプションがいくつかあります。ストリーミング インストールの実装をテストする場合は、次のコマンドを使用できます。

xbapp install [/l] [/w] [/i] <パッケージ名>

|  |  |
| --- | --- |
| オプション | 詳細 |
| <パッケージ名> | MakePkg コマンドから作成したインストール パッケージ。 |
| /l | 起動チャンクのみをインストールするように指定します。起動チャンクがインストールされればゲームを実行できますが、レイアウト ファイルで "Launch" とマークされたチャンクでリストされているコンテンツだけが使用できます。 |
| /w | インストール時にパイプを開いたままにするように指定します。これにより、最初にインストールされていない機能と追加の言語チャンクの動的インストールをテストできます。 |
| /i | 対話型モードを指定します。これにより、対話型のインストールと削除のためにパッケージに定義されている利用可能な機能を表示するインストール管理のシステム UI が表示されます。 |
| <完全なパッケージ名> | コンソールにインストールされた後のゲームの名前。パッケージ ファミリ名と混同しないようにしてください。コマンド プロンプトから **xbapp list** コマンドを実行すると、コンソールにインストール済みパッケージの一覧を取得できます。 |

/l および /w を使用すると、ゲームのインストールを異なるシナリオでテストできます。/l のみを使用すると、システムでは初期のチャンクのみをインストールします。/l および /w を使用すると、システムによって初期のチャンクがインストールされ、アプリ起動後、残りの部分がインストールされます。/w のみを使用すると、ゲームでレイアウト ファイルで定義された有効なレシピに基づいてベース パッケージ全体がインストールされ、パイプを開いたままにできます。これらのパラメーターを使用しない場合、ベース パッケージがインストールされるだけで、サンプルで追加のインストールは実行できません。

/w はリテール動作に最も似ており、開いたままとなっているパイプを通常 CDN でホストされているパッケージ ソースの可用性と捉えることができます。

いくつかのチャンクは、ユーザーのシステムと言語に基づいて自動的にインストールされます。たとえば、言語を "en-US" に設定し、Scarlett ファミリ パッケージを使用して、Scarlett コンソールに配置する場合、他の言語またはデバイス "Xbox-Lockhart" でタグ付けされたチャンクはインストールされません。Durango ファミリ パッケージでは、ターゲット デバイス (または開発キットのデバッグ設定)に基づいて、Durango コンテンツまたは Scorpio コンテンツがインストールされます。

xbapp installコマンドにオプションの引数を指定して使用することで、他のチャンクのインストールを強制することができます。

|  |  |
| --- | --- |
| オプション | 詳細 |
| /Languages | セミコロンで区切った言語の一覧でインストールする言語を指定します。コンソール設定を変更した後でインストール コマンドを再実行する必要なく、複数の言語でローカライズされたコンテンツをテストするために使用できます。 |
| /Devices | インストールする必要があるハードウェア固有のチャンクを指定します。セミコロンで区切られた、含めるさまざまなハードウェア SKUの一覧です。 |
| /Recipes | インストールする必要のあるレシピを指定します。セミコロンで区切られた一覧で、レイアウト ファイルで定義されている Recipe 指定子です。 |
| /Features | インストールする必要がある機能を指定します。セミコロンで区切られた一覧で、レイアウト ファイルで定義されている Feature 指定子です。 |
| /AllChunks | コンソール ハードウェアまたは設定に関係なく、インストールする必要があるすべてのチャンクを指定します。 |

これらは **xbapp install** コマンドで使用できるオプションを詳しく説明することを目的とした表ではありません。包括的なオプションの一覧については、GDK のドキュメントを参照してください。

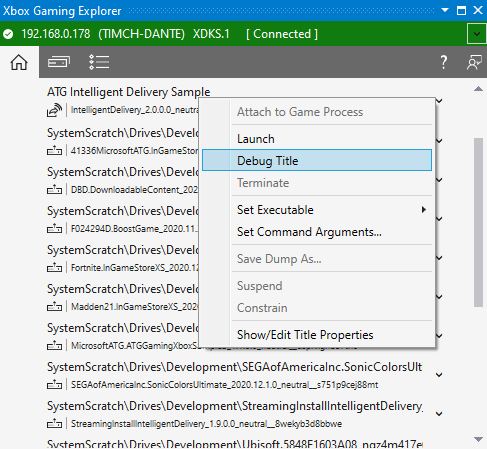
**PC** の場合、代わりに wdapp コマンドが使用できます。起動チャンクをインストールする /l フラグのみ使用できます。サンプルはパッケージ ソースをホストするデバイスと同じデバイスで実行されるので、プロセスをアクティブにすることなく、インテリジェント配信の操作を実行できます。

wdapp install [/l] <パッケージ名>

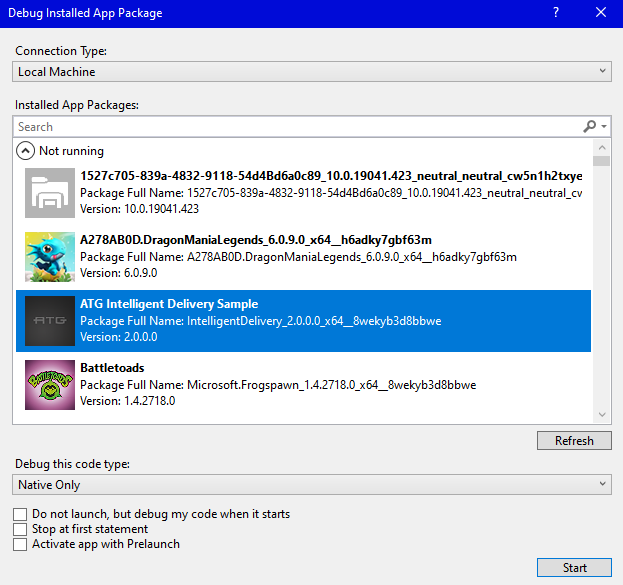
# サンプルのデバッグ

このサンプルは通常 Visual Studio から起動するだけではデバッガーで実行できないので、他の方法を使用してパッケージ化されたビルドをデバッグする必要があります。

**Xbox** の場合、Xbox Gaming Explorer を使用して、登録タイトルの一覧にあるサンプルのデバッグ セッションを開始できます。



**PC** の場合、[デバッグ] > [その他のデバッグ ターゲット] > [インストールされているアプリケーション パッケージのデバッグ] を使用します。



このサンプルでは、次の操作を使用できます。

|  |  |  |  |
| --- | --- | --- | --- |
| 操作 | ゲームパッド コントロール | マウスとキーボード | |
| グリッド メニュー内を移動 | 方向パッド | 方向キー |
| グリッド メニューの項目を選択  インストールできる場合、Geature をインストール  インストールできる場合、言語チャンクをインストール  複数が保留中の場合、最初にアイテムをインストール  現在のアイテムがインストール後であり削除可能な場合にはそれをアンインストール | A ボタン | 左クリック / Enter キー |
| すべての Feature とチャンクをリストし、それらの XPackageNkAvailability 値をデバッグ テキストで表示 | Y ボタン | Y キー |
| 画面のデバッグ テキストを切り替え | メニュー ボタン | チルダ ( ` ) |
| 既存のサンプル | ビュー ボタン | Esc |

# 更新履歴

2019 年 4 月、サンプルの最初のリリース。

2020 年 3 月、サンプルを更新し、XboxOne デバイス ファミリと Scarlett デバイス ファミリの構成を分離。

2021 年 2 月、サンプル名を IntelligentDelivery に変更、Recipe と Feature のサポートを修正。

2021 年 3 月、サポートできるプラットフォームとして PC を追加。

# プライバシーに関する声明

サンプルをコンパイルして実行する場合、サンプルの使用状況を追跡するために、サンプル実行ファイルのファイル名が Microsoft に送信されます。このデータ コレクションからオプトアウトするには、Main.cpp の「Sample Usage Telemetry」というラベルの付いたコードのブロックを削除します。

全般的な Microsoft のプライバシー ポリシーの詳細については、「[Microsoft のプライバシーに関する声明](https://privacy.microsoft.com/en-us/privacystatement/)」を参照してください。