記事
Toshihiko Minamoto · 2021年3月10日 9m read

Open Exchange

ZPM モジュールの構造: InterSystems ソリューションのパッケージ化。

デベロッパーの皆さん、こんにちは！

最近、当社は InterSystems Package Manager (ZPM) をリリースしました。 ZPM を開発した理由の 1 つは、ソリューションをパッケージ化して ZPM レジストリに提出することにより、そのデプロイを「install xxx package」のようなコマンドを実行するだけの単純な作業にするためです。

これを行うには、InterSystems IRIS パッケージの中身を説明する module.xml ファイルをリポジトリに導入する必要があります。

この記事では、module.xml ファイルの異なる構成要素を説明し、独自のファイルを作成する方法をご紹介します。

まずは、samples-objectscript パッケージから始めます。以下のコマンドを実行すれば、IRIS に ObjectScript のサンプルアプリケーションがインストールされます。

```
zpm: USER>install samples-objectscript
```

おそらく、これほどシンプルなパッケージはないと思います。以下は、パッケージの中身を説明する module.xml です。

```
<Export generator="Cache" version="25">
  <Document name="samples-objectscript.ZPM">
    <Module>
      <Name>samples-objectscript</Name>
      <Version>1.0.0</Version>
      <Packaging>module</Packaging>
      <SourcesRoot>src</SourcesRoot>
      <Resource Name="ObjectScript.PKG"/>
    </Module>
  </Document>
</Export>
```

それでは、文書の中身を一行ずつ見ていきましょう。

module.xml は、Cache/IRIS の XML 文書のファミリーに属しているため、この行はその関係性を示すもので、内部ライブラリがこの文書を認識できるようにしています。

次のセクションはこちら。

パッケージには名前を付けます。 名前には小文字と「-」の記号を含めることができます。 (この場合なら samples-objectscript)。 パッケージ名は、拡張子に「.ZPM」を使い、Document タグの name

節に入れてください。

Document の内部要素は以下の通りです。

- パッケージ名。 今回は、

```
<Name>samples-objectscript</Name>
```

- パッケージバージョン。 今回は、

```
<Version>1.0.0</Version>
```

モジュール - パッケージの種類。 モジュールパラメーターは以下のように入力します。

```
<Packaging>module</Packaging>
```

- ZPM がインポートする ObjectScript を探すフォルダー。

今回は「/src」フォルダーで ObjectScript を探すよう指示します。

```
<SourcesRoot>src</SourcesRoot>
```

- インポートする ObjectScript の要素。 パッケージやクラス、include、グローバル、dfi などが含まれます。

SourceRoot フォルダー以下の構造は次のようになります。

/cls - Folder=Package 内にある Class=file.cls 形式のすべての ObjectScript クラス。
サブパッケージはサブフォルダーです。

/inc - *file.inc* 形式のすべての include ファイル。

/mac - すべての mac ルーチン。

/int - すべての「中間」ルーチン (別名「他のコード」、mac
コードをコンパイルした結果、またはクラスやマクロを持たない ObjectScirpt)。

/gbl - XML 形式でエクスポートされるすべてのグローバル。

/dfi - XML 形式でエクスポートするすべての DFI ファイル。 各ピボットは pivot.dfi
ファイルに、各ダッシュボードは dashboard.dfi ファイルに作成されます。

例えば、 ObjectScript ページをインポートするとします。 これをうけて、ZPM は /src/cls/ObjectScript
フォルダーの中を見て、そこからすべてのクラスをインポートします。

```
<Resource Name="ObjectScript.PKG"/>
```

なので、 パッケージ化するソリューションを作成するには、ObjectScript
クラスを「/cls」フォルダー内にあるリポジトリのいずれかのフォルダーに入れ、すべてのパッケージとクラスを class=file.cls 形式の package=folder に入れます。

クラスを違うかたちでリポジトリに保管したいけれども、いちいち手作業で ObjectScript 用に適切なフォルダー構造を作成するのは避けたいという場合、それを解決してくれるツールはたくさんあります。例えば、Atelier や VSCode Object Script は、まさにその条件が満たされるかたちでクラスをエクスポートしますし、他にはパッケージ化する準備ができているアーチファクトをネームスペースからすべてエクスポートしてくれる [isc-dev]

ユーティリティなどがあります。

mac ルーチンのパッケージ化

これはクラスの場合とよく似ています。 ルーチンを /mac フォルダーに入れるだけです。 例はこちら。

```
<Export generator="Cache" version="25">
  <Document name="DeepSeeButtons.ZPM">
    <Module>
      <Name>DeepSeeButtons</Name>
      <Version>0.1.7</Version>
      <Packaging>module</Packaging>
      <SourcesRoot>src</SourcesRoot>
      <Resource Name="DeepSeeButtons.mac"/>
    </Module>
  </Document>
</Export>
```

その他の要素

また、以下のような、オプションとして使える要素もあります。

と の要素が含まれる場合があります。

例:

```
<Author>
        <Organization>InterSystems</Organization>
        <CopyrightDate>2019</CopyrightDate>
      </Author>
```

CSP/Web アプリケーションのパッケージ化

ZPM はウェブアプリケーションもデプロイできます。

これを成功させるには、CSPApplication 要素を CSPApplication パラメーターの節と一緒に導入します。

例えば、DeepSeeWeb の module.xml にある CSPApplication タグをご覧ください。

```
<CSPApplication
        Url="/dsw"
        DeployPath="/build"
        SourcePath="${cspdir}/dsw"
        ServeFiles="1"
        Recurse="1"
        CookiePath="/dsw"
       />
```

この設定により、**/dsw** という名前のウェブアプリケーションが作成され、リポジトリの **/build** フォルダーにあるすべてのファイルが IRIS csp ディレクトリの **${cspdir}****/dsw** フォルダーにコピーされます。

REST API アプリケーション

これが REST-API アプリケーションである場合、CSPApplication 要素はディスパッチクラスを含み、MDX2JSON module.xml のように構成される可能性があります。

```
<CSPApplication
    Path="/MDX2JSON"
    Url="/MDX2JSON"
    CookiePath="/MDX2JSON/"
    PasswordAuthEnabled="1"
    UnauthenticatedEnabled="1"
    DispatchClass="MDX2JSON.REST"
    />
```

依存関係

モジュールをインストールする際には、ターゲットシステムに別のモジュールがインストールされている場合があります。 これは、複数の 要素を含み得る 要素の中にある 要素により 説明されている場合があります。
このそれぞれに、 と があり、 また先にインストールされているべき
他のモジュールとそれらのバージョンが記述されています。 こういった状況では、ZPM はモジュールがインストールされているかどうかを確認し、インストールされていなければ、インストールを実行します。

こちらは、DSW モジュールの MDX2JSON モジュールに対する依存関係を示した例です。

```
<Dependencies>
        <ModuleReference>
          <Name>MDX2JSON</Name>
          <Version>2.2.0</Version>
        </ModuleReference>
      </Dependencies>
```

以下も依存関係の例です。ThirdPartyPortlets が Samples BI(holefoods) に依存しています。

```
<Dependencies>
        <ModuleReference>
          <Name>holefoods</Name>
          <Version>0.1.0</Version>
        </ModuleReference>
      </Dependencies>
```

また、任意のコードを実行して、データと環境をセットアップするというオプションもあります。これについては、次回の記事で解説します。

独自のパッケージをビルドする方法

それでは、 module.xml ができたら、パッケージをビルドして、module.xml
の構造が正しいかどうかをテストすることができます。

テストは、ZPM クライアントを使って実行できます。 ZPM を IRIS
システムにインストールし、以下の読み込みコマンドでパッケージのコードを読み込みます。

zpm: NAMESPACE>load path-to-the-project

パスは、パッケージのリソースを含み、かつルートフォルダーに module.xml を持つフォルダーを指しています。

例えば、 パッケージのビルドは、こちらのプロジェクトを使ってテストできます。
チェックアウトしたら、docker-compose-zpm.yml を使ってコンテナをビルドします。

SAMPLES ネームスペースでターミナルを開き、ZPM を呼び出します。

```
zpm: SAMPLES>

zpm: SAMPLES>load /iris/app

[samples-objectscript]  Reload START
[samples-objectscript]  Reload SUCCESS
[samples-objectscript]  Module object refreshed.
[samples-objectscript]  Validate START
[samples-objectscript]  Validate SUCCESS
[samples-objectscript]  Compile START
[samples-objectscript]  Compile SUCCESS
[samples-objectscript]  Activate START
[samples-objectscript]  Configure START
[samples-objectscript]  Configure SUCCESS
[samples-objectscript]  Activate SUCCESS
```

パスが「/iris/app」になっているのは、docker-compose-zpm.yml
の中でプロジェクトのルートをコンテナの「/iris/app」フォルダーにマップすると指定しているためです。
したがって、このパスを使えば、ZPM にプロジェクトの読み込み元を指定することができます。

ついに、 読み込みに成功しました。
つまり、パッケージをデベロッパーコミュニティのリポジトリに提出する際にこの module.xml
を使えるということです。

以上が、アプリケーションに使う適切な module.xml ファイルを作成する方法です。

InterSystems コミュニティのリポジトリにアプリケーションを提出する方法

現時点で、要件は 2 つあります。

1. アプリケーションが Open Exchange に記載されている。

2. Community Package Manager のリポジトリ
にアプリケーションを提出することをご希望の方は、私までダイレクトメッセージをお送りいただくか、
この記事のコメント欄よりお知らせください。

module.xml が正常に動作していることをご確認ください！)

#InterSystems Package Manager (IPM) #チュートリアル #初心者 #InterSystems IRIS #Open Exchange
InterSystems Open Exchangeで関連アプリケーションを確認してください

ソースURL:
https://jp.community.intersystems.com/post/zpm-%E3%83%A2%E3%82%B8%E3%83%A5%E3%83%BC%E3%83%AB%E3%81%AE%E6%A7%8B%E9%80%A0-intersystems-%E3%82%BD%E3%83%AA%E3%83%A5%E3%83%BC%E3%82%B7%E3%83%A7%E3%83%B3%E3%81%AE%E3%83%91%E3%83%83%E3%82%B1%E3%83%BC%E3%82%B8%E5%8C%96%E3%80%82