記事
Toshihiko Minamoto · 2021年2月23日 6m read

Java Business Host から PEX への移行

Java Business Host から PEX への移行

InterSystems IRIS 2020.1 および InterSystems IRIS for Health 2020.1 で PEX がリリースされ、Java Business Host を使うよりも優れたかたちで Java プログラムをプロダクション環境に取り込めるようになりました。 PEX は、相互運用性のコンポーネントを構築するための API をすべて提供するほか、Java と .NET の両方で使用できます。 Java Business Host は非推奨となり、今後のリリースで廃止される予定です。

PEX を使うメリット

- デベロッパーはプロダクションのすべてのコンポーネントを、Java と .Net のどちらでも構築できる
- コンポーネント間で一層複雑なメッセージ構造の受け渡しができる
- シンプルな設定
- 開発のワークフローがシンプルな上に、ObjectScript を使う必要がない

ここからは、既存の Java Business Host のコードを PEX に移行する方法に注目します。

概要

PEX で使用されるクラスとインターフェースは、Java Business Host (JBH) のものとは異なります。
本記事では、その相違点を要約して解説しますが、詳細は完全なドキュメントをご覧ください。

- ビジネスサービスの変換
- ビジネスオペレーションの変換
- 設定
- メッセージ
- ログ

ビジネスサービスを Java Business Host から PEX に変換する

PEX のビジネスサービスを構築するには、com.intersystems.gateway.bh.BusinessService の代わりに、com.intersystems.enslib.pex.BusinessService を実装する必要があります。

PEX で使用されるビジネスサービスの設計パターンは、サービスがスレッドを開始してメッセージを作成するというものから、サービスが定期的に呼び出される関数を実装してメッセージを作成するというものに一変しました。

JBH では、以下のようなコードが使われています。

```
@Override
public boolean OnInit(Production p) throws Exception {
  production = p;

  if (messageThread == null) {
    Messager messager = new Messager();
```

```
      messageThread = new Thread(messager);
      messageThread.start();
    }

    return true;
  }
```

一方の PEX では、関数を 3 つ実装するだけで OK です。

```
  public void OnInit() throws Exception {
    // ???
    return;
  }

  public Object OnProcessInput(Object messageInput) throws Exception {
    // ??? SendMessage() ? SendMessageAsync() ???????

    return null;
  }

  public void OnTearDown() throws Exception {
    // ???????
    return;
  }
```

また、設定はどのように使用され、メッセージはどのように配信され、ログはどのように記録されるのかも変更する必要があります。 詳しくは後ほど解説します。

ビジネスオペレーションを Java Business Host から PEX に変換する

PEX のビジネスオペレーションを構築するには、com.intersystems.gateway.bh.BusinessOperation の代わりに com.intersystems.enslib.pex.BusinessOperation を実装する必要があります。

ビジネスオペレーションの設計パターンは、JBH も PEX も構造的には同じですが、2 つのメインエントリーポイントへのパラメーターが変更されています。

OnInit() の変更点

PEX の OnInit() はパラメーターを受け取りません。

OnMessage() の変更点

PEX の場合、OnMessage() には、JBH で使用される String の代わりに、ジェネリック型の Object が与えられます。 これにより、プロダクションの作成者は好きなメッセージを渡すことができます。

JBH では、アプリケーションに以下のようなコードが使われていたのではないでしょうか

```
  public boolean OnMessage(String message) throws Exception {
    // ??????????
    return true;
  }
```

PEX では、パラメーターにジェネリック型の Java Object
が使用されます。適切にキャストする必要がありますが、String
を使った場合よりも一層複雑なメッセージを送信できます。
以下は、ファイルストリームであるリクエストを抽出する方法を示した例です。

```
  public Object OnMessage(Object request) throws Exception {
    com.intersystems.jdbc.IRISObject streamContainer = (com.intersystems.jdbc.IRISObj
ect)request;
    com.intersystems.jdbc.IRISObject str = (com.intersystems.jdbc.IRISObject)streamCo
ntainer.get("Stream");
    String originalFilename = (String)streamContainer.get("OriginalFilename");

    Long contentSize = (Long)str.get("Size");
    String content = (String)str.invoke("Read", contentSize);

    // ??????????

    return null;
  }
```

また、設定が使用される方法、メッセージが配信される方法、ログが記録される方法も変更する必要があります。
詳しくは後ほど解説します。

設定

設定の宣言が簡単になりました。

JBH では、設定は SETTINGS 文字列を使って宣言され、以下のようなコードで取り込まれていました。

```
  String setting = production.GetSetting("Min");
  if (!setting.isEmpty()) {
    min = Integer.parseInt(setting);
  }
```

PEX の場合、設定は単純に public メンバフィールドとなります。
これらは、クラスがインスタンス化されるときに自動的に設定されます。

```
  public int Min = 0;
```

public メンバーフィールドは、Java の基本データ型 (String や int など)
であれば、何でもプロダクション環境で設定できます。

メッセージ

メッセージはよりパワフルに送信できます。 JBH では、メッセージは文字列として送信されます。 一方の PEX を使うと、メッセージは、 ObjectScript で定義されるオブジェクトの場合であれは、オブジェクト (IRISObject) として送信され、Java で定義されているクラスの場合であれば、com.intersystems.enslib.pex.Message のサブクラスとして送信されます。

JBH の場合は、以下のようなコードが使われます

```
production.SendRequest(value.toString());
```

PEX の場合だと、以下のようなコードが使われます

```
MyExampleMessageClass req = new MyExampleMessageClass("message to send");
SendRequestAsync(Target, req);
```

ログ

ログ機能はすべて似たようなものですが、名前だけが違います。

PEX で情報メッセージをログするときは、LOGINFO() を使います。

```
LOGINFO("Received message");
```

オブジェクトのゲートウェイ

Java Business Host では、専用のゲートウェイが必要でしたが、 PEX では、Java ゲートウェイ 1 つで、Java のすべてのニーズに対応できます。 また、ゲートウェイはたくさん使うこともできます。
これはあなた次第です。 こちらの Java ゲートウェイの手引き がおすすめです。

結論とフィードバック

まだ PEX を試していないという方は、ぜひぜひお試しください。 PEX を使えば、少ないコードで解決できるビジネスの問題の幅をぐっと広げることができます。また、今はすべての作業を .NET で実行できるようにもなりました。

JBH のアプリケーションを PEX へ移行させることに関するご質問や問題は、私か WRC までご連絡ください。

#.NET #Java #相互運用性 #InterSystems IRIS #InterSystems IRIS for Health

---

ソースURL:https://jp.community.intersystems.com/post/java-business-host-%E3%81%8B%E3%82%89-pex-%E3%81%B8%E3%81%AE%E7%A7%BB%E8%A1%8C