WebSphere Liberty Profile での EAR 配置

Maven を使用して `mvn package` で EAR（Enterprise Archive）ファイルを成功裏に構築したようです。ビルドプロセスはエラーなく完了しましたが、この EAR を WebSphere Liberty Profile（WLP）にデプロイし、サーバーを起動すると、「no module found」というエラーが発生します。これは、スムーズなビルドが成功したにもかかわらず、サーバーがアプリケーションを認識しないという状況が謎です。この問題を分析し、解決策を見つけるために、原因を特定し、対処方法を検討します。

問題の理解

EAR ファイルは、Java EE アーカイブの標準形式で、通常、WAR ファイル（ウェブアプリケーション用）、JAR ファイル（EJB またはライブラリ用）、デプロイメント記述子（`application.xml`）を含むアプリケーションをパッケージ化します。EAR を WLP にデプロイすると、サーバーはこれらのモジュールを検出し、アプリケーションを起動するはずです。「no module found」というメッセージは、WLP が EAR 内のモジュールを見つけられないか、認識できないことを示しています。

Maven のビルドが成功したため（「すべてがスムーズ」）、問題は以下の 3 つの領域のいずれかにある可能性が高いです：1. **EAR ファイルの内容**：EAR には期待されるモジュールが含まれていないか、デプロイメント記述子が欠けているか、不正確です。2. **デプロイメントプロセス**：EAR を WLP にデプロイする方法が、サーバーが見つけ、処理する方法と一致していないかもしれません。3. **サーバーの設定**：WLP が EAR 内のモジュールを認識できないのは、必要な機能が欠けているか、設定が不正確であるためです。

これらの可能性を検討し、診断と修正のための具体的な手順を提供します。

可能な原因と解決策

1. EAR ファイルが空か、不正確にパッケージ化されている可能性　ビルドが成功したにもかかわらず、EAR にモジュール（例えば、WAR や JAR ファイル）が含まれていないか、`application.xml` ファイルが欠けているか、不正確である可能性があります。

- **なぜ起こるか**：Maven の EAR プロジェクトでは、`maven-ear-plugin` が EAR を組み立てる責任を持ちます。これは、`pom.xml` の設定や依存関係に基づいてモジュールを含めます。モジュールが指定されていないか、依存関係（例えば、WAR）が正しく定義されていないか解決されていない場合、EAR は空か、適切な `application.xml` が欠けている可能性があります。
- **確認方法**：
  - EAR ファイル（ZIP アーカイブ）を開くためにツール（例えば、`unzip`）を使用するか、ターミナルで `jar tf myApp.ear` を実行して内容をリストアップします。
  - 確認する項目：
    * モジュールファイル（例えば、`my-web.war`、`my-ejb.jar`）が EAR のルートにあります。

    * `META-INF/application.xml` ファイルが存在します。
  - `application.xml` 内には、モジュールを定義するエントリが含まれているはずです、例えば：

```xml
<?xml version="1.0" encoding="UTF-8"?>
<application>
    <module>
        <web>
            <web-uri>my-web.war</web-uri>
            <context-root>/myapp</context-root>
        </web>
    </module>
</application>
```

- **修正方法**：
  - EAR モジュールの `pom.xml` を確認し、含めるモジュールの依存関係を指定していますか、例えば：

```xml
<dependencies>
    <dependency>
        <groupId>com.example</groupId>
        <artifactId>my-web</artifactId>
        <type>war</type>
    </dependency>
</dependencies>
```

  - 必要に応じて `maven-ear-plugin` を設定します：

```xml
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-ear-plugin</artifactId>
    <version>3.3.0</version>
    <configuration>
        <modules>
            <webModule>
                <groupId>com.example</groupId>
                <artifactId>my-web</artifactId>
                <contextRoot>/myapp</contextRoot>
            </webModule>
        </modules>
    </configuration>
</plugin>
```

  - `mvn clean package` で再ビルドし、EAR の内容を再確認します。

EAR が空か、`application.xml` が欠けているか不正確である場合、これが原因です。Maven の設定を修正することで解決できます。

---

2. デプロイメント方法の問題　EAR を WLP にデプロイする方法にも問題があるかもしれません。WLP は、主に `dropins` ディレクトリと `server.xml` での明示的な設定の 2 つの主要なデプロイメント方法をサポートしています。

- **`dropins` ディレクトリを使用する**：
  - EAR を `wlp/usr/servers/<serverName>/dropins/`ディレクトリに置くと、WLP は自動的に検出し、デプロイします。
  - ただし、EAR ファイルの場合、自動デプロイは期待通りに動作しないことがあります。特に、追加の設定（例えば、コンテキストルート）が必要な場合です。
- **`server.xml` を使用する**：
  - EAR ファイルの場合、`wlp/usr/servers/<serverName>/server.xml` でアプリケーションを明示的に設定する方が良いです：

```
<server>
    <featureManager>
        <feature>servlet-3.1</feature> <!-- 必要な機能が有効になっていることを確認します -->
    </featureManager>
    <application id="myApp" name="myApp" type="ear" location="${server.config.dir}/apps/myApp.ear"/>
</server>
```

  - EAR を `wlp/usr/servers/<serverName>/apps/`に置くか（または `location` パスを適切に調整します）。
- **確認方法**：
  - EAR の場所とサーバーの起動方法（例えば、`./bin/server run <serverName>`）を確認します。
  - サーバーのログ（例えば、`wlp/usr/servers/<serverName>/logs/console.log` または `messages.log`）を確認し、デプロイメントメッセージを確認します。
- **修正方法**：
  - 上記のように `server.xml` で EAR を設定するか、`dropins` を使用するのをやめてください。
  - 変更を加えた後、サーバーを再起動します：`./bin/server stop <serverName>` の後に`./bin/server start <serverName>`。

EAR がサーバーに適切に登録されていない場合、これがエラーの原因です。

---

3. サーバー機能の欠如　WLP は、`server.xml` で有効にする機能のみを読み込む軽量サーバーです。EAR に特定の機能（例えば、サーブレット、EJB）が必要なモジュールを含まれている場合、その機能が有効になっていないと、WLP はモジュールを認識できないか、読み込めないかもしれません。

- **なぜ起こるか**：例えば、WAR ファイルには `servlet-3.1` 機能（またはそれ以上）が必要です。EJB モジュールには `ejbLite-3.2` が必要です。これらがないと、サーバーはモジュールを処理できないかもしれません。
- **確認方法**：
  - `server.xml` の `<featureManager>` セクションを確認します。
  - 一般的な機能には以下があります：
    * `<feature>servlet-3.1</feature>` はウェブモジュール用です。
    * `<feature>ejbLite-3.2</feature>` は EJB モジュール用です。
  - サーバーログを確認し、機能に関するメッセージ（例えば、「必要な機能がインストールされていません」）を確認します。
- **修正方法**：
  - アプリケーションの要件に基づいて、`server.xml` に必要な機能を追加します：

```
<featureManager>
    <feature>servlet-3.1</feature>
    <!-- 必要に応じて他の機能を追加します -->
</featureManager>
```

  - 変更を適用するためにサーバーを再起動します。

機能が欠けている場合、これを有効にすることで WLP がモジュールを認識できるようになります。

診断手順

問題を特定するために、以下の手順を実行します：

1. **EAR ファイルの確認**：
   - `jar tf myApp.ear` を実行するか、解凍します。
   - モジュール（例えば、`my-web.war`）と有効な `META-INF/application.xml` が含まれていることを確認します。
2. **Maven ビルドの確認**：
   - EAR モジュールの `pom.xml` を確認し、依存関係と `maven-ear-plugin` の設定を確認します。
   - `mvn clean package` を再実行し、モジュールを含めるメッセージ（例えば、「モジュール my-web.war を追加」）をビルド出力で確認します。
3. **デプロイメントの確認**：
   - EAR の場所（例えば、`dropins` または `apps`）を確認します。
   - `dropins` を使用している場合、`apps` に移動し、`server.xml` に追加してみてください。

4. **サーバーログの確認**：
    - サーバーを起動し、`console.log` または `messages.log` で「no module found」より詳細なエラーメッセージを確認します。
    - 「アプリケーションにはモジュールが含まれていません」や機能に関するエラーのヒントを探します。
5. **シンプルな EAR でテスト**：
    - 知っている動作する `pom.xml` を使用して、単一の WAR ファイルを含む基本的な EAR を作成し、WLP にデプロイします。動作する場合、問題は元の EAR のパッケージングに特有です。

最も有力な原因

ビルドがスムーズだったにもかかわらず WLP が「no module found」と報告しているため、最も有力な原因は**EAR ファイルにモジュールが含まれていないか、`application.xml` が欠けているか不正確である**ことです。これは、`pom.xml` が含めるモジュールを正しく指定していないために起こることがあります。ただし、EAR が正しい場合、問題はデプロイメント（例えば、`server.xml` で設定されていない）またはサーバー機能の欠如にあるかもしれません。

最終的な推奨

- EAR の内容（`jar tf myApp.ear`）と `pom.xml` の設定を確認してください。
- EAR が空か、`application.xml` が欠けているか不正確である場合、Maven の設定を修正し、再ビルドします。
- EAR が正しい場合、`server.xml` で設定し、必要な機能が有効になっていることを確認し、再デプロイします。
- サーバーログを確認し、追加のコンテキストを確認します。

これらの手順を実行することで、問題がビルド、デプロイメント、またはサーバー設定にあるかを特定し、WLP でアプリケーションを正常に実行できるようになります。特定の設定に関するヘルプが必要な場合はお知らせください！