# 日日是Oracle APEX

Oracle APEXを使った作業をしていて、気の付いたところを忘れないようにメモをとります。

2024年5月28日火曜日

## Oracle Database 23aiのJavaScriptでのモジュールの呼び出し方を調べる

Oracle Database 23aiのMLE - Multilingual Engineで、JavaScriptのモジュールを呼び出す方法について確認してみました。

以下の資料を主に参照しています。

**Oracle Database Release 23**
**JavaScript Developer's Guide**
4 Overview of Importing MLE JavaScript Modules

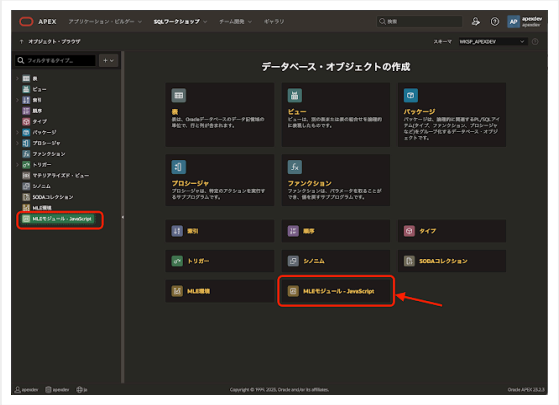
実装のサンプルとしては、以下のLiveLabsも参考にしています。
APEX + Server-Side JavaScript (MLE)
**Lab 4: Using External Modules**

JavaScriptの実行には、Always FreeのOracle Autonomous Database 23aiのOracle APEX 23.2を使います。

Example 4-2のFunction Export using Named Exportsに記載されているMLEモジュール**named_exports_module**を作成します。モジュールに含まれるファンクションsumとdifferenceがエクスポートされます。

```
CREATE OR REPLACE MLE MODULE named_exports_module LANGUAGE JAVASCRIPT AS

function sum(a, b) {
  return a + b;
}

function difference(a, b) {
  return a - b;
}

export {sum, difference};
/
```

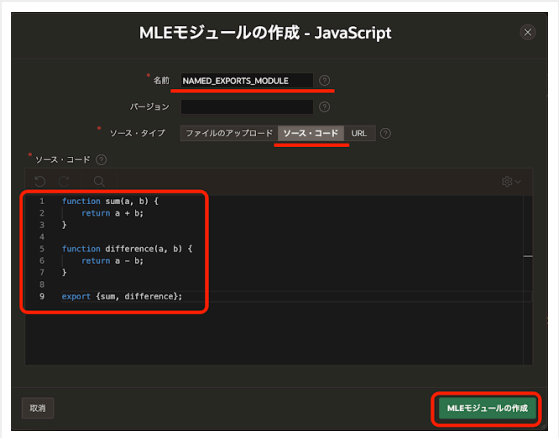
**SQLワークショップ**の**オブジェクト・ブラウザ**より、**MLEモジュール**を作成します。



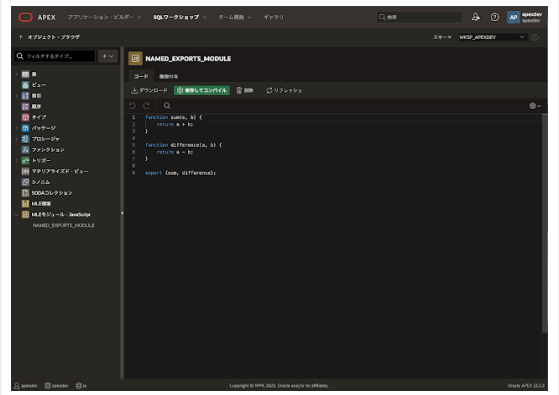

**名前**は**NAMED_EXPORTS_MODULE**です。APEXからMLEモジュールを作成する際は名前が大文字になりますが、コードで作成してもダブル・クォーテーションで囲まなければ、ディクショナリには大文字で登録されます。

**ソース・タイプ**に**ソース・コード**を選択し、**ソース・コード**（JAVASCRIPT AS 以下）を記述します。

**MLEモジュールの作成**をクリックします。



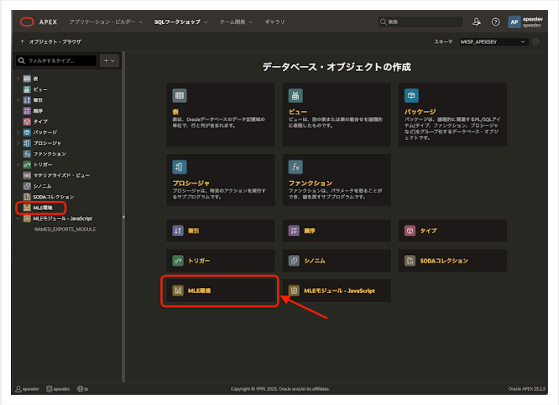

**MLEモジュール**として**NAMED_EXPORTS_MODULE**が作成されます。




作成した**MLEモジュール**に含まれる、エクスポートされたファンクション**sum**および**diffrence**を呼び出してみます。

**MLE環境**として**NAMED_EXPORTS_ENV**を作成します。

```
CREATE MLE ENV named_exports_env;
```

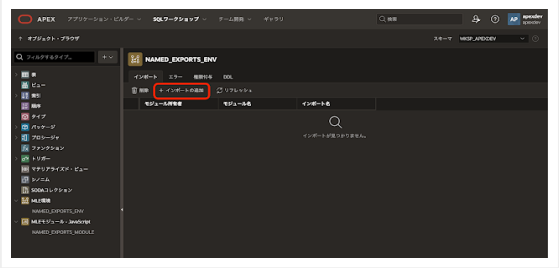



**環境名**は**NAMED_EXPORTS_ENV**とします。

**MLE環境の作成**をクリックします。


**MLE環境**として**NAMED_EXPORTS_ENV**が作成されます。**インポート**・タブを開き、**インポートの追加**をクリックします。

先ほど作成したMLEモジュール**NAMED_EXPORTS_MODULE**を、このMLE環境**NAMED_EXPORTS_ENV**にインポート可能なモジュールとして登録します。



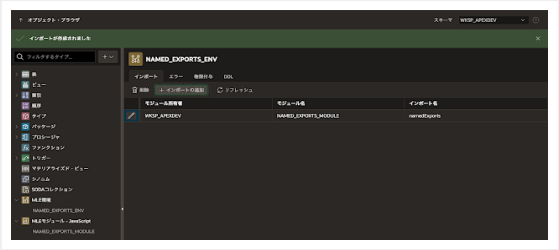

内部的には以下のスクリプトが実行されます。

```
alter MLE ENV named_exports_env
   add imports('namedExports' module named_exports_module);
```

**モジュール所有者**は**APEXのワークスペース・スキーマ**、**モジュール名**は**NAMED_EXPORTS_MODULE**です。**インポート名**は**namedExports**とします。インポート名は大文字小文字が区別されます。

**作成**をクリックします。


MLE環境**NAMED_EXPORTS_ENV**にインポート可能なMLEモジュールとして、**NAMED_EXPORTS_MODULE**が追加されます。JavaScriptのコードから、このモジュールをインポートする際には、インポート名**namedExports**を指定します。




**SQLワークショップ**の**SQLコマンド**から、モジュール**NAMED_EXPORTS_MODULE**でエクスポートされているファンクション**sum**と**difference**を呼び出してみます。

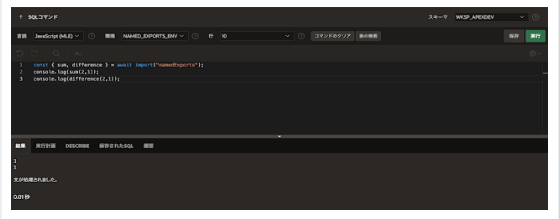
**言語**として**JavaScript(MLE)**を選択し、**環境**として**NAMED_EXPORTS_ENV**を選択します。MLEモジュール**namedExports**のインポートは、以下の文で実行します。

**const { sum, difference } = await import("namedExports");**

以下のスクリプトを実行します。

```
const { sum, difference } = await import("namedExports");
console.log(sum(2,1));
console.log(difference(2,1));
```

sumとdifferenceが呼び出せます。

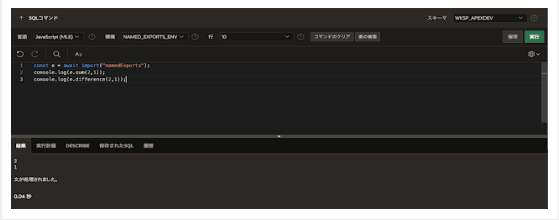


以下のスクリプトでも確認します。

**const e = await import("namedExports");**

結果は同じになります。

```
const e = await import("namedExports");
console.log(e.sum(2,1));
console.log(e.difference(2,1));
```



MLEモジュール**NAMED_EXPORTS_MODULE**の**sum**と**difference**を呼び出すために、このモジュールをインポートするMLEモジュール**NAMED_IMPORTS_MODULE**を作成します。

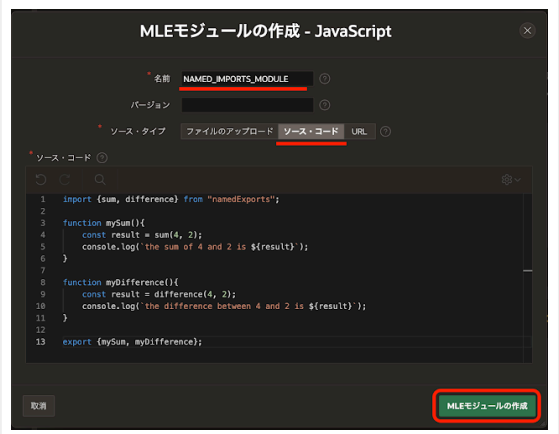
**Example 4-7 Named Imports Using Specified Identifiers**として記載されているコードを実行します。

```
CREATE OR REPLACE MLE MODULE named_imports_module
LANGUAGE JAVASCRIPT AS

import {sum, difference} from "namedExports";

function mySum(){
   const result = sum(4, 2);
   console.log(`the sum of 4 and 2 is ${result}`);
}

function myDifference(){
   const result = difference(4, 2);
   console.log(`the difference between 4 and 2 is ${result}`);
}

export {mySum, myDifference};
/
```

先ほどと同じ手順でMLEモジュールの作成を呼び出します。

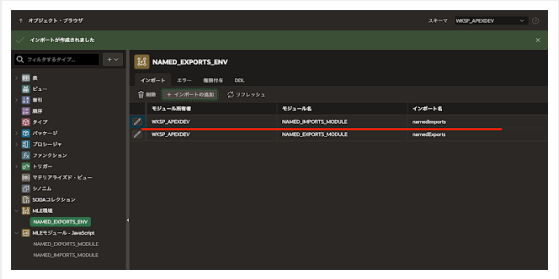
**名前**は**NAMED_IMPORTS_MODULE**、**ソース・タイプ**は**ソース・コード**です。MLEモジュール**NAMED_EXPORTS_MODULE**のインポートは、以下の文で実行します。

**import {sum, difference} from "namedExports";**




作成したMLEモジュールをMLE環境**NAMED_EXPORTS_ENV**のインポートに追加します。**インポート名**は**namedImports**とします。


MLE環境**NAMED_EXPORTS_ENV**にインポート名**namedImports**が追加されます。




SQLコマンドよりインポート名**namedImports**を指定してインポートし、ファンクション**mySum**と**myDifference**を呼び出してみます。

```
const { mySum, myDifference } = await import("namedImports");
mySum();
myDifference();
```

**namedImports**としてインポートされたMLEモジュール**NAMED_IMPORTS_MODULE**から、MLEモジュール**NAMED_EXPORTS_MODULE**のファンクションが参照できていることが確認できます。

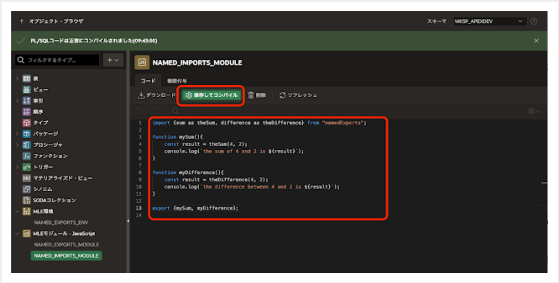
Example 4-6 Module Object Definitionに記載されているモジュール本文は以下です。この内容でモジュールNAMED_IMPORTS_MODULEを置き換えても、動作は変わりません。

```
import * as myMath from "namedExports"

function mySum(){
   const result = myMath.sum(4, 2);
   console.log(`the sum of 4 and 2 is ${result}`);
}

function myDifference(){
   const result = myMath.difference(4, 2);
   console.log(`the difference between 4 and 2 is ${result}`);
}

export {mySum, myDifference};
```

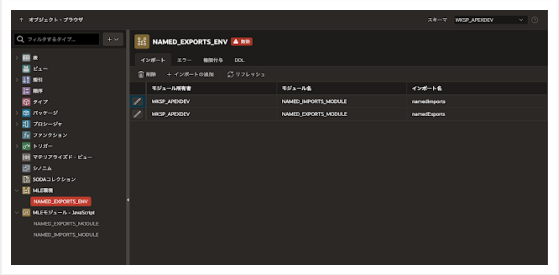
Example 4-8 Named Imports with Aliasesの本文も同様です。

```
import {sum as theSum, difference as theDifference} from "namedExports";

function mySum(){
   const result = theSum(4, 2);
   console.log(`the sum of 4 and 2 is ${result}`);
}

function myDifference(){
   const result = theDifference(4, 2);
   console.log(`the difference between 4 and 2 is ${result}`);
}

export {mySum, myDifference};
```

実際にモジュール**NAMED_IMPORTS_MODULE**の本文を置き変え、**保存してコンパイル**を行います



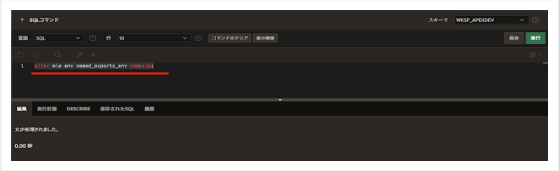
コンパイル後にオブジェクト・ブラウザをリロードすると、MLE環境NAMED_EXPORTS_ENVが無効になっていることが確認できます。



現在のところ、オブジェクト・ブラウザにはMLE環境をバリッドに戻す方法が提供されていません。

**SQLコマンド**を開き、**言語**を**SQL**にして以下のALTER文を実行し、MLE環境をコンパイルします。

```
alter mle env named_exports_env compile;
```



今までは簡単なコードを書いてMLEモジュールを作成していました。

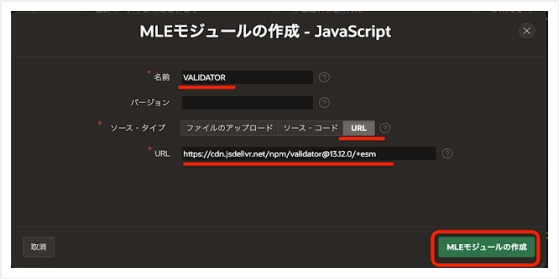
実際はESモジュールからMLEモジュールを作成することができます。

文字列の検証を行うvalidator.jsをMLEモジュールとして作成してみます。MLEモジュールのソースとして以下のURLを指定します。

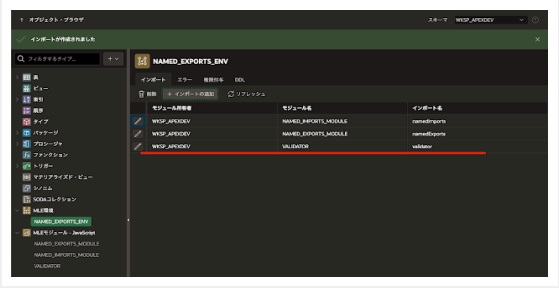
**https://cdn.jsdelivr.net/npm/validator@13.12.0/+esm**

MLEモジュールの作成画面を開きます。

**名前**は**VALIDATOR**、**ソース・タイプ**として**URL**を選択し、**URL**にESモジュールのソースを指すURLを指定します。




作成したMLEモジュール**VALIDATOR**をMLE環境**NAMED_EXPORTS_ENV**に**インポート名validator**として追加します。

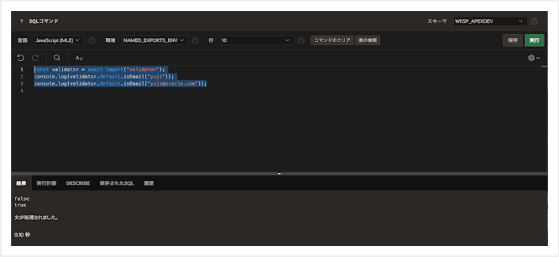



**SQLコマンド**から、**validator.js**に含まれているファンクション**isEmail**を呼び出してみます。

以下のコードを実行します。

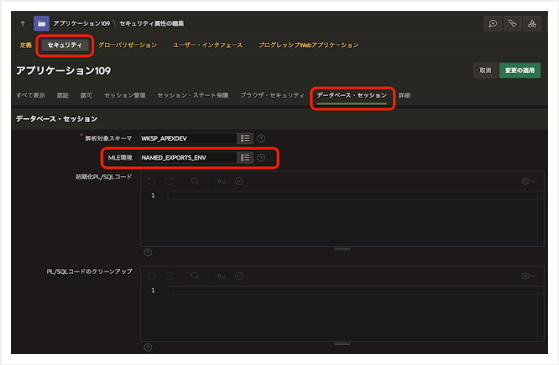
```
const validator = await import("validator");
console.log(validator.default.isEmail("yuji"));
console.log(validator.default.isEmail("yuji@oracle.com"));
```

isEmailの最初の呼び出しはfalse、次の呼び出しはtrueが返されることが確認できます。



Oracle DatabaseのMLEではモジュールがDatabaseに保存されているため、Node.jsやブラウザとは扱い方が若干異なります。そのため、Oracle Database 23aiのMLE上でJavaScriptを実行するにあたって、モジュールがどのように扱われるかを確認してみました。

ちなみにOracle APEXのアプリケーションで参照する**MLE環境**は、**アプリケーション定義**の**セキュリティ**の**MLE環境**で設定します。




今回の記事は以上になります。

**完**

Yuji N. 時刻: 19:20

共有

‹ ホーム ›

ウェブ バージョンを表示

**自己紹介**

**Yuji N.**

日本オラクル株式会社に勤務していて、Oracle APEXのGroundbreaker Advocateを拝命しました。
こちらの記事につきましては、免責事項の参照をお願いいたします。

詳細プロフィールを表示