

# MLE助力APEX全栈开发

公益讲座11:00准时开始,请大家先浏览云技术微信公众号技术文章。资料会在各群同步发布,已入群客户请勿重复入群!



20-21

数据库和云讲座群



甲骨文云技术公众号



B站专家系列课程



## **立即扫码进行1V1 免费咨询**

**2023年10月，MySQL 5.7 将终止官方支持和更新。**  
立刻升级至更快、更稳定、更安全的 MySQL 8.0 /  
MySQL Database Service，获取 300+ 项新特性，  
使开发更加灵活和高效，更好的满足业务发展需求。

**免费咨询热线：  
400-699-8888**

\* 活动最终解释权归甲骨文公司所有

# MLE 助力 APEX 全栈开发

甲骨文技术公益课 - 数据库专场

2023 年 9 月 1 日 11:00

线上直播

Cathy Xing

# Agenda

---

## ➤ Oracle MLE介绍

- MLE、GraalVM是什么
- MLE JavaScript在数据库端实操

## • MLE JavaScript助力APEX

- SQL Workshop中执行JS代码
- APEX应用中使用MLE JS
- 为什么需要在服务端执行JS
- 如何在APEX中使用第三方JS Module

## • MLE Demo 分享

- 校验、二维码生成、Markdown转HTML、HTML过滤器、图片处理、情感分析

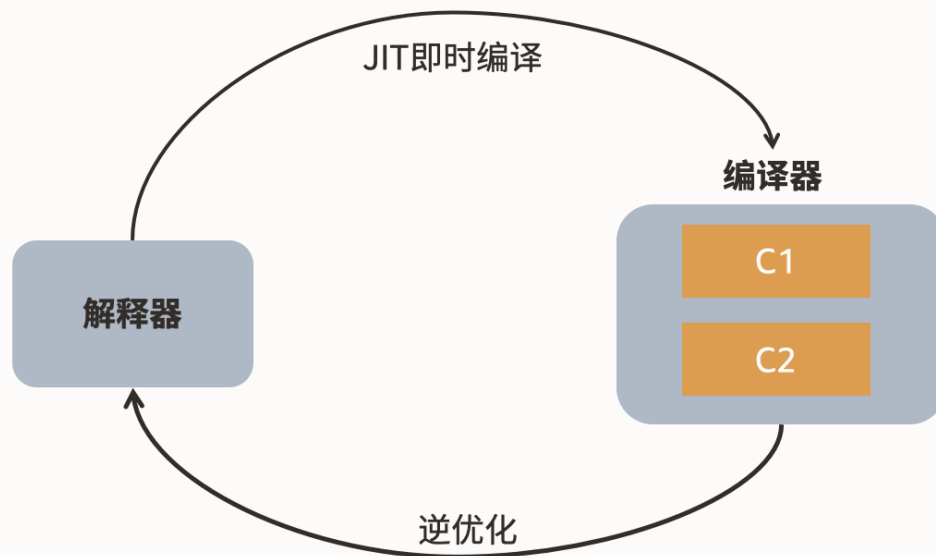
# MLE介绍 – Graal是什么

## Graal是使用Java重新实现的JIT编译器

我们知道Java是一次编写，到处执行，是通过JVM的运行时来实现的，JVM屏蔽了各平台的差异，从而允许在任何平台上运行Java应用。

而JVM使用两种方法运行类文件：**解释器**和**JIT编译器**

- **解释器**：单个语句翻译成机器代码是由语言处理器完成的，并在进入下一行之前立即执行
- **编译器**：编译器将字节码转换为机器语言并执行来提高Java应用程序的性能

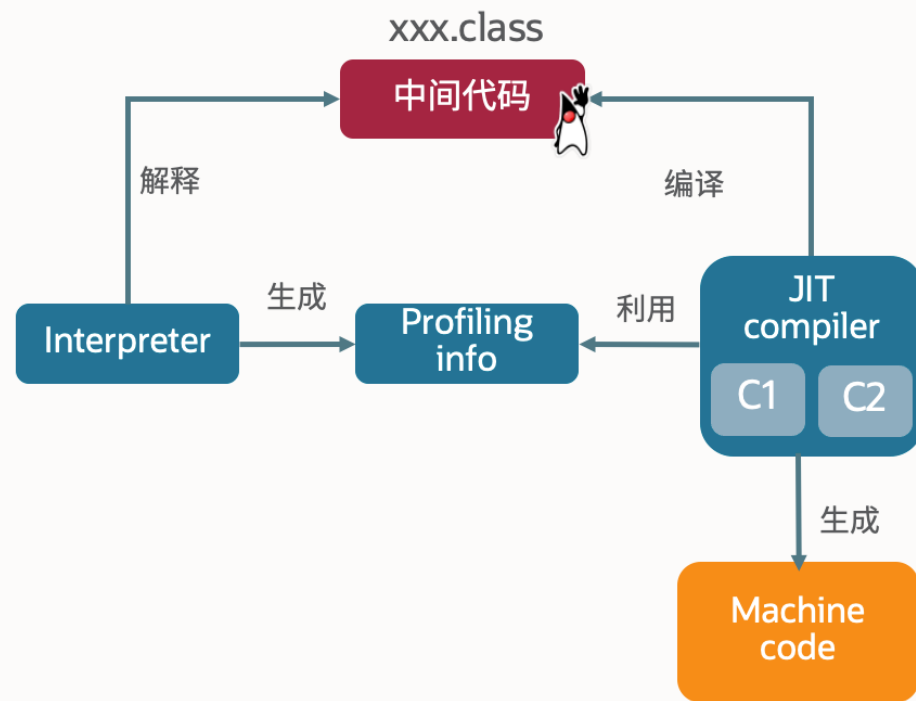


# MLE介绍 – Graal是什么

使用JIT（Just In Time）编译器又分为两种：

- **C1客户端**：快速编译方法，快速启动和较小的内存占用比稳态性能更重要，但生成的机器代码不如服务器编译器优化，更注重启动速度和局部的优化
- **C2服务器**：通常需要更多的时间（和内存）来编译相同的方法，但会生成比C1生成的代码更好的优化机器代码，稳态性能比快速启动更重要

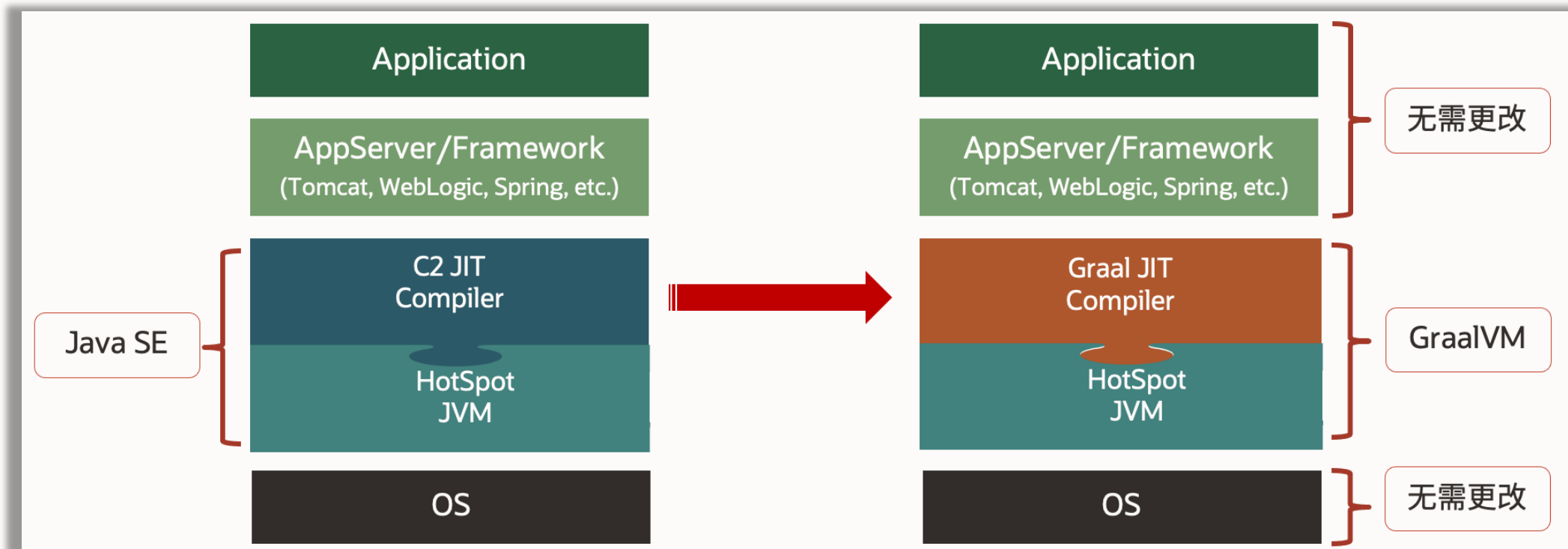
## JVM 编译器的工作原理



# MLE介绍 - GraalVM是什么

GraalVM 起始于 2011 年 Oracle 实验室的一个研究项目。该项目旨在创建一个可以杰出性能运行多种编程语言的运行时平台，如下所述，其核心是高级优化 GraalVM 编译器。GraalVM 编译器可用作 Java 虚拟机的即时 (JIT) 编译器，或帮助 GraalVM本地镜像提前将Java字节码编译为原生机器码

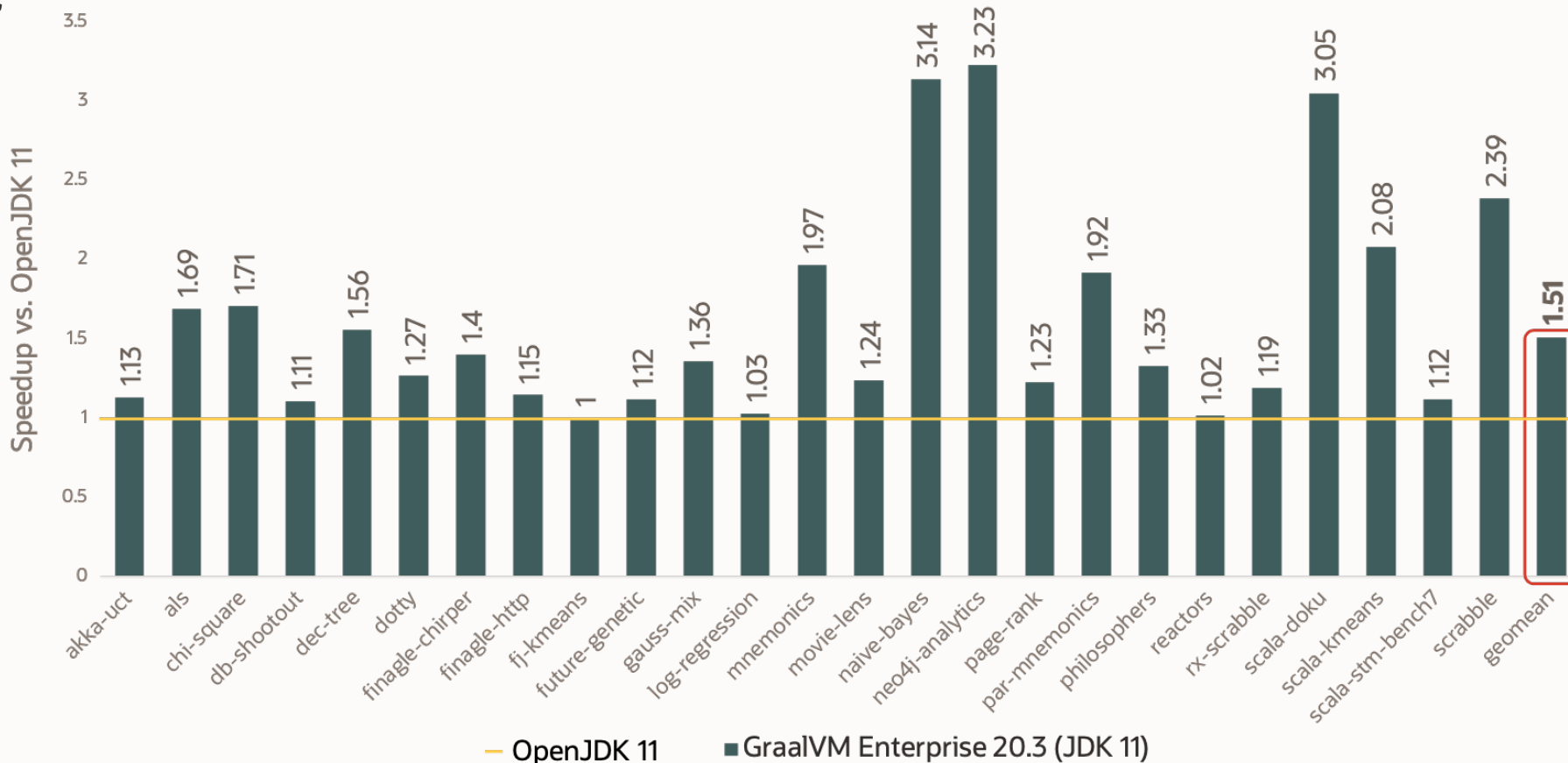
## GraalVM是从JVM内部将C2 JIT编译器替换为新的Graal JIT编译器



# MLE介绍 - GraalVM是什么

对现实世界的应用进行基准测试的结果，性能得到提升

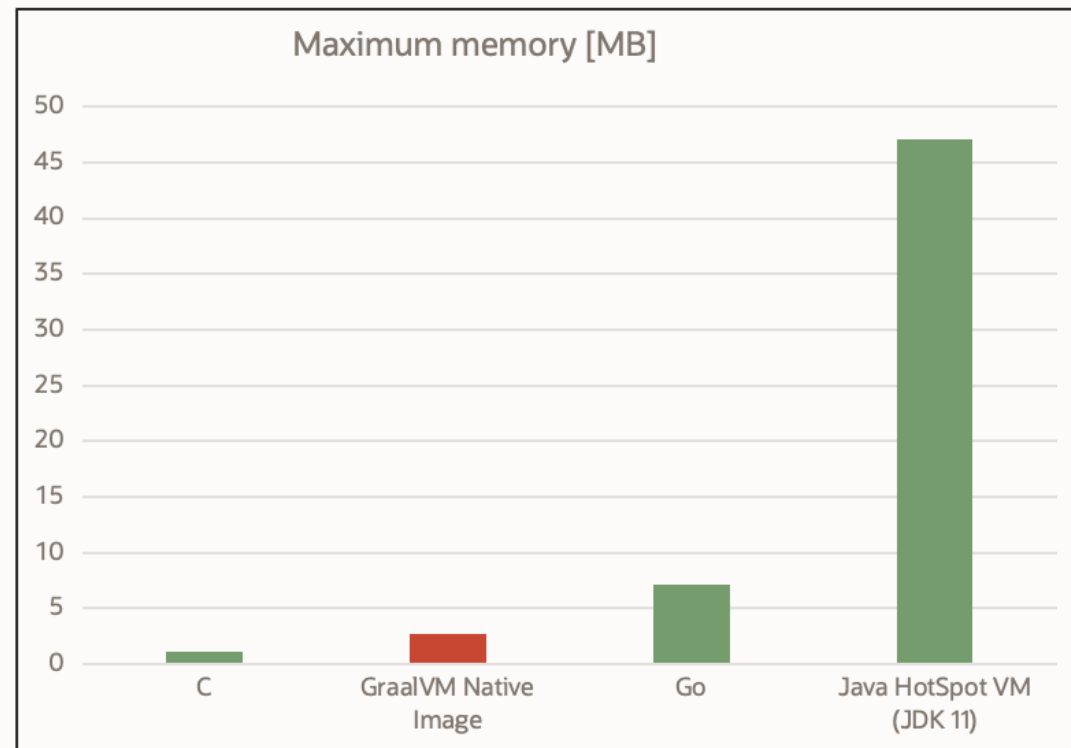
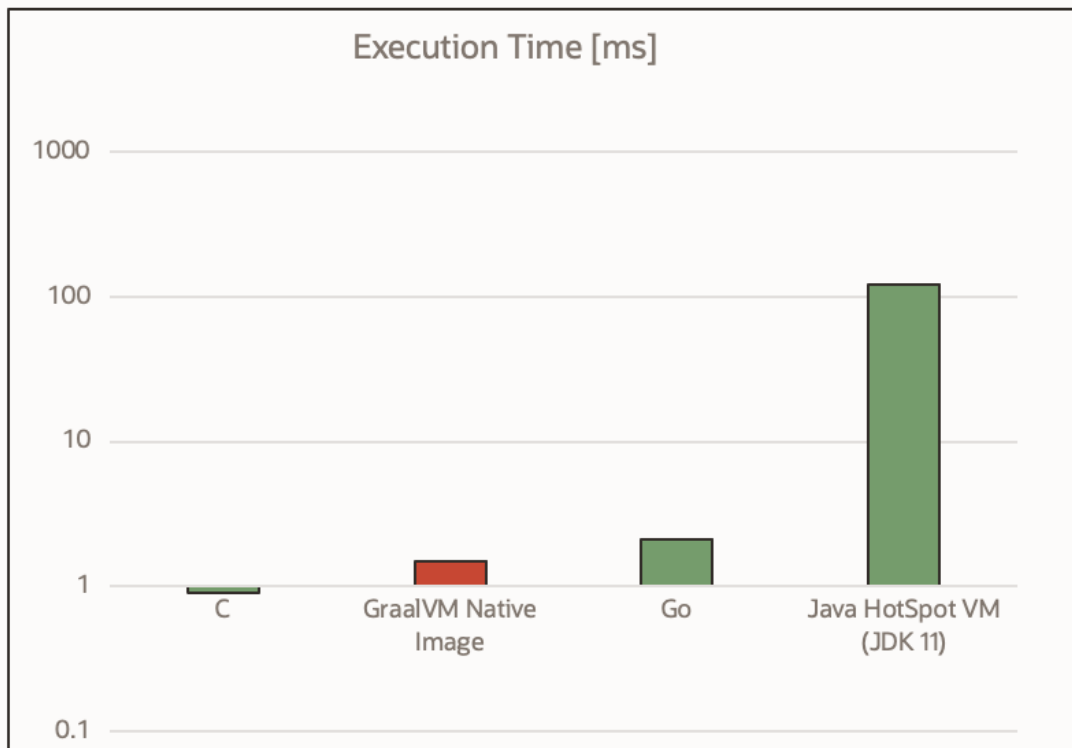
GraalVM 提前将Java 应用程序编译成独立的二进制文件，这些二进制文件可以立即启动，无需预热即可提供峰值性能





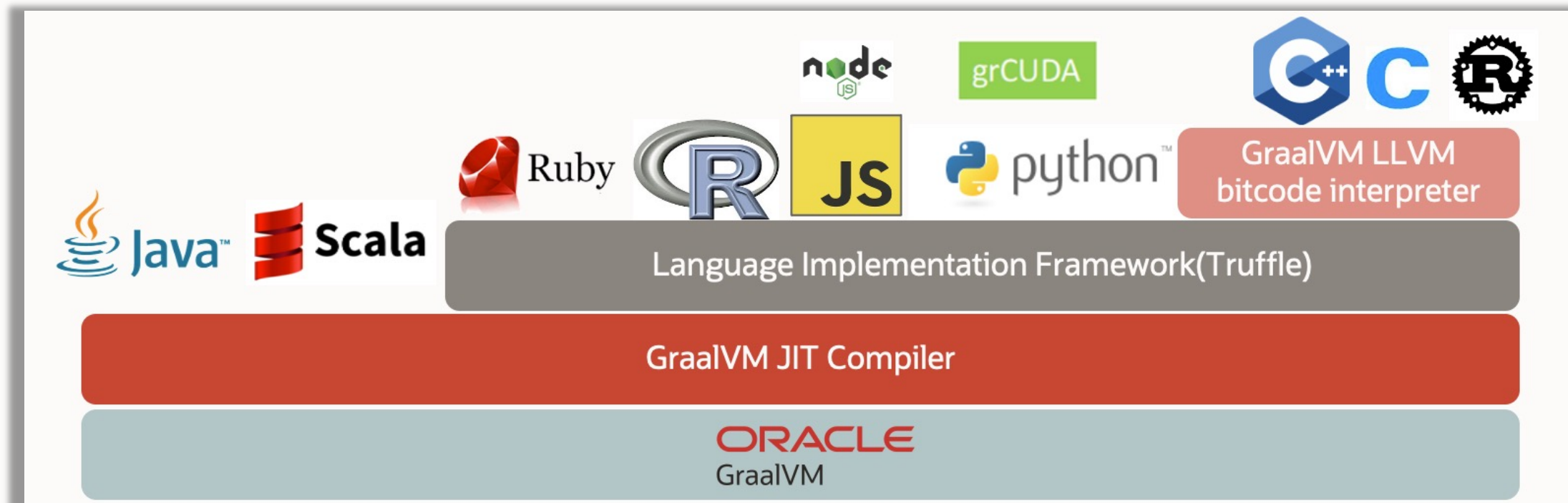
# MLE介绍 - GraalVM是什么

更快、更轻、更省内存



# MLE介绍 – GraalVM是什么

通过Graal编译器与Truffle语言实施框架和Sulong相结合，共同协作，以卓越性能运行JavaScript、Python、R、Ruby 以及LLVM支持的其他等语言



# MLE介绍 – MLE是什么

MLE, Oracle Database Multilingual Engine 数据库多语言引擎

用于在 Oracle 数据库中执行 GraalVM 的运行时

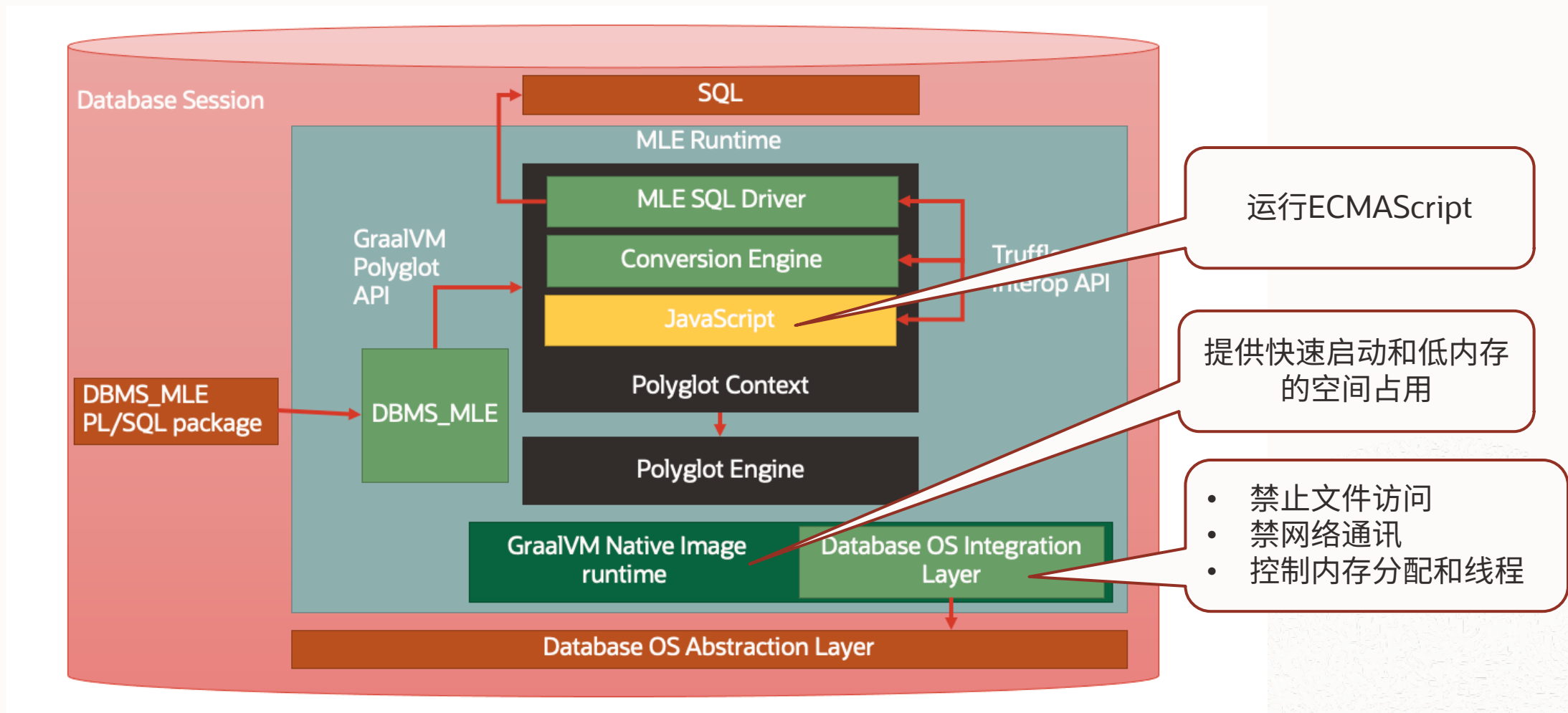
- MLE JavaScript在Oracle 18c中首次作为Beta选项引入
- 21c中发布, 通过**DBMS\_MLE**包来执行JavaScript
  - 作为一项功能提供, 仅用于运行 JavaScript 片段
- 23c中通过扩展了: **持久性MLE Module**模块、从SQL或PL/SQL来调用模块等
  - 允许创建可重用的代码块, 称为模块

目前仅支持MLE JavaScript, 将来GraalVM所能支持其它语言, 也会被加进来



# MLE介绍 - MLE是什么

MLE架构图：基于GraalVM



# Agenda

## ➤ Oracle MLE介绍

✓ MLE、GraalVM是什么

➤ **MLE JavaScript在数据库端实操**

## • MLE JavaScript助力APEX

- SQL Workshop中执行JS代码
- APEX应用中使用MLE JS
- 为什么需要在服务端执行JS
- 如何在APEX中使用第三方JS Module

## • MLE Demo 分享

- SQL Workshop中执行JS代码

# MLE JavaScript 数据库端实操

Hello World

```
1  DECLARE
2      ctx DBMS_MLE.context_handle_t := DBMS_MLE.create_context();
3  BEGIN
4      DBMS_MLE.eval(ctx, 'JAVASCRIPT', q'~console.log("Hello, World!");~');
5      DBMS_MLE.drop_context(ctx);
6  END;
7  /
```

1. 创建JS执行上下文
2. 在当前上下文中，执行JS代码
3. 清除JS上下文

# MLE JavaScript 数据库端实操

## 如何传递参数呢?

### 通过DBMS\_MLE动态执行JavaScript

PL/SQL与JavaScript之间参数互传

PL/SQL

```
DECLARE
  ctx DBMS_MLE.context_handle_t := DBMS_MLE.create_context;
  res number := 0;
BEGIN
  dbms_output.put_line('PL/SQL res: '||res);
  DBMS_MLE.export_to_mle(ctx, 'val', 42);
  DBMS_MLE.eval(ctx, 'JAVASCRIPT', q'~
    let bindings = require("mle-js-bindings");
    let val = bindings.importValue("val"); // 42
    bindings.exportValue("res", val + 7);
    console.log("js val: "+val);
  ~');
  -- res = 49
  DBMS_MLE.import_from_mle(ctx, 'res', res);
  dbms_output.put_line('PL/SQL res: '|| res);
  DBMS_MLE.drop_context(ctx);
END;
```

JS

PL/SQL

#### 在PL/SQL中

```
/* 设置val的值到MLE的JS上下文中 */
DBMS_MLE.export_to_mle(ctx, 'val', 42);
/* 在PL/SQL中, 从JS上下文中导入res的值取出 */
DBMS_MLE.import_from_mle(ctx, 'res', res);
```

#### 在JS中

```
/* 引入val参数 */
bindings.importValue("val");
/* 设置res的值导出到PL/SQL中 */
bindings.exportValue("res", val + 7);
```



# MLE JavaScript 数据库端实操

## DBMS\_MLE - 使用JavaScript SQL Driver

使用JavaScript将数据库表中的数据读取出来：

```
DECLARE
ctx DBMS_MLE.context_handle_t;
user_code clob := q'~
    const oracledb = require("mle-js-oracledb");

    // implicit connection - server-side execution
    const conn = oracledb.defaultConnection();

    // execute query with bind, fetch result rows
    const query = "select empno, ename, sal from emp where sal < :1 order by sal desc";
    const res = conn.execute(query, [3000]);
    for (let row of res.rows) {
        console.log("empno: " + row[0] + ", and the name: " + row[1] + ", sal: " + row[2])
    };
~';
BEGIN
ctx := DBMS_MLE.create_context();
DBMS_MLE.eval(ctx, 'JAVASCRIPT', user_code);
DBMS_MLE.drop_context(ctx);
END;
```



# 为什么在服务端来执行JavaScript?

- **服务端执行JavaScript更安全!**

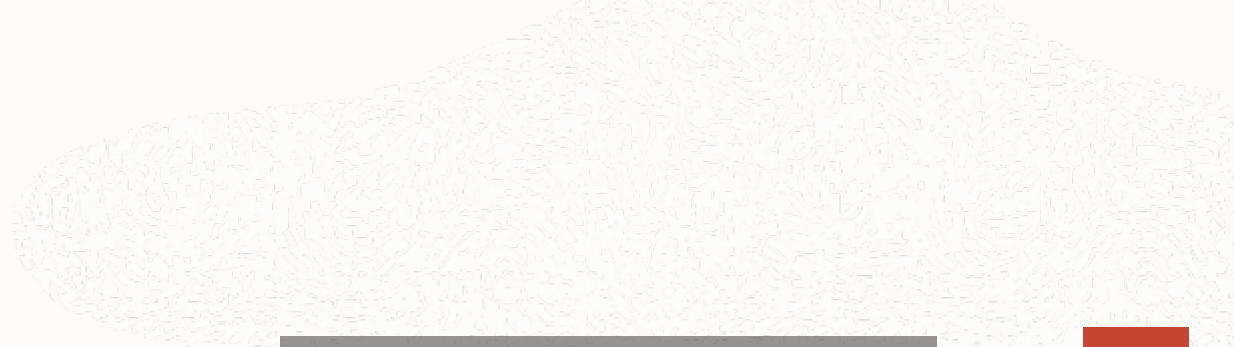
JavaScript在客户端运行的安全性一直是JS的弱项一般的存在，而如果能在服务端执行JS，安全性大大提高，弥补了长久以来的安全性弱点

- **解锁了更多的使用场景**

比如，在服务端生成二维码时，就可以实现在文档中嵌入二维码、在服务端发邮件时发送二维码图片等等。那些仅凭客户端无法做到的

- **与其它平台对接选择更多了**

在做一些集成的时候，一般可能会提供一些常见语言的SDK，JS往往是其中之一，但一般不会提供PL/SQL的SDK，于是常常要在客户端来做这些实际是基于服务端的集成，并不合适也不安全。再比如，一些加密算法，也存在会提供主流开发语言的，但可能没有PL/SQL的，而现在这些就都可以直接在服务端运行了，集成变得更容易了



# Agenda

## ✓ Oracle MLE介绍

- ✓ *MLE、GraalVM是什么*
- ✓ *MLE JavaScript在数据库端实操*

## ➤ MLE JavaScript助力APEX

- *SQL Workshop中执行JS代码*
- *APEX应用中使用MLE JS*
- *为什么需要在服务端执行JS*
- *如何在APEX中使用第三方JS Module*

## • MLE Demo 分享

- *校验、二维码生成、Markdown转HTML、HTML过滤器、图片处理、情感分析*

# APEX

## APEX官网

- 永久免费
- 低代码

### 关于对APEX的评价 Gartner Peer Insights



以 4.8 分的高分  
获得 95% 认为它是值得推荐的低代码平台  
拥有 70% 以上 5 星记录的厂商

其它竞争者有:

- Microsoft PowerApps
- Salesforce
- ServiceNow
- OutSystems
- Mendix
- Appian
- Pega

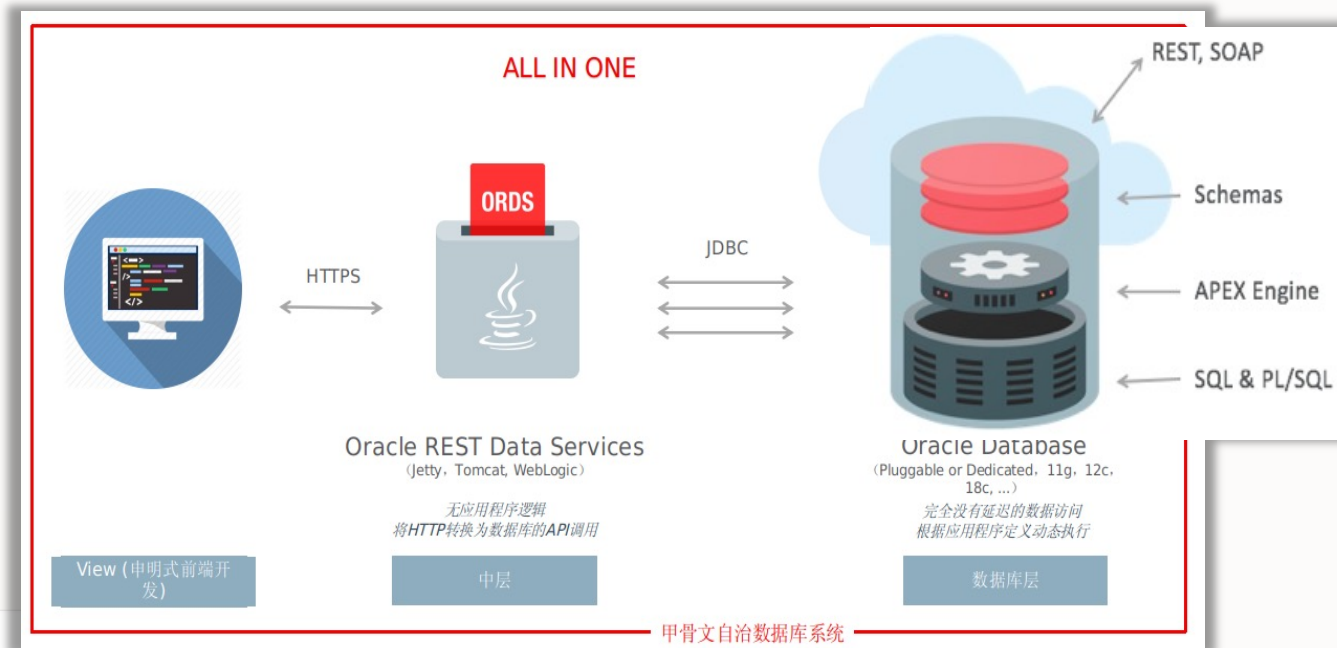


5.0 ★★★★★ Dec 22, 2021

**Oracle APEX, a low-code great tool**

Reviewer Role: Enterprise Architecture and Technology Innovation    Company Size: 3B - 10B USD    Industry: Services Industry

Several low-code platforms are in the market nowadays, our company explored some of them and, decided for Oracle APEX, basically because it is a very mature platform that has been evolving with a very clear vision and purpose over the years and, is close to our data infrastructure, thus, ...

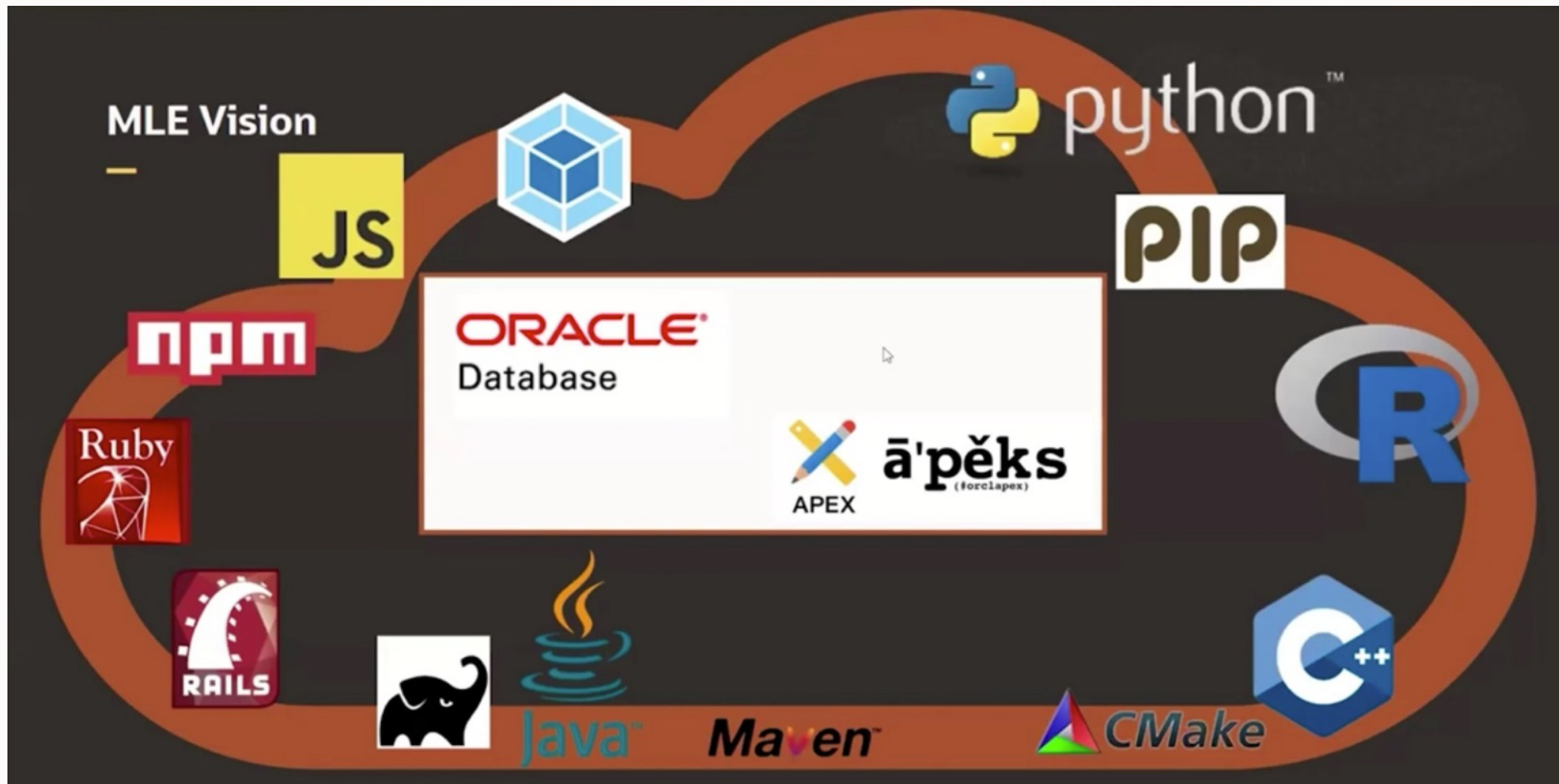


- 500,000+ 开发者
- 2+ Million 应用
- 6,000+ 每天新创建应用
- 50,000+ 客户
- 150+ 合作伙伴



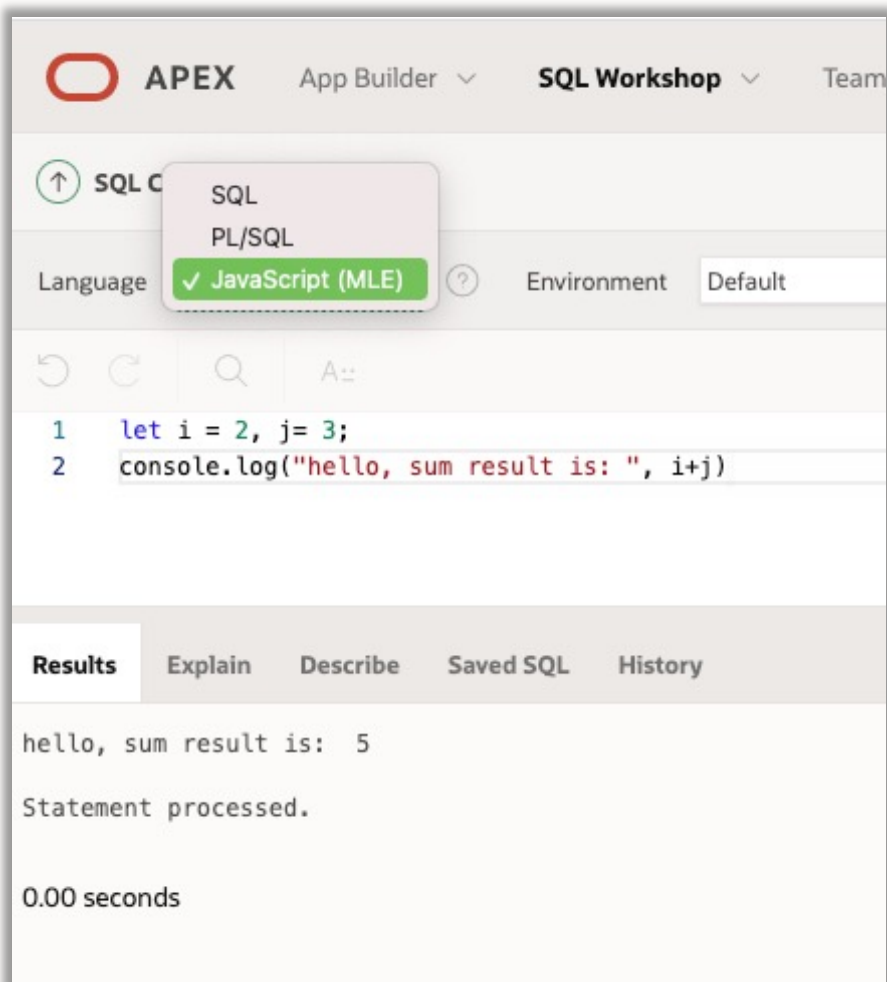
# MLE跟APEX又是什么关系?

APEX中直接原生支持JavaScript!!



# MLE JavaScript助力APEX – SQL Commands初体验

借助MLE, APEX变身为多语言

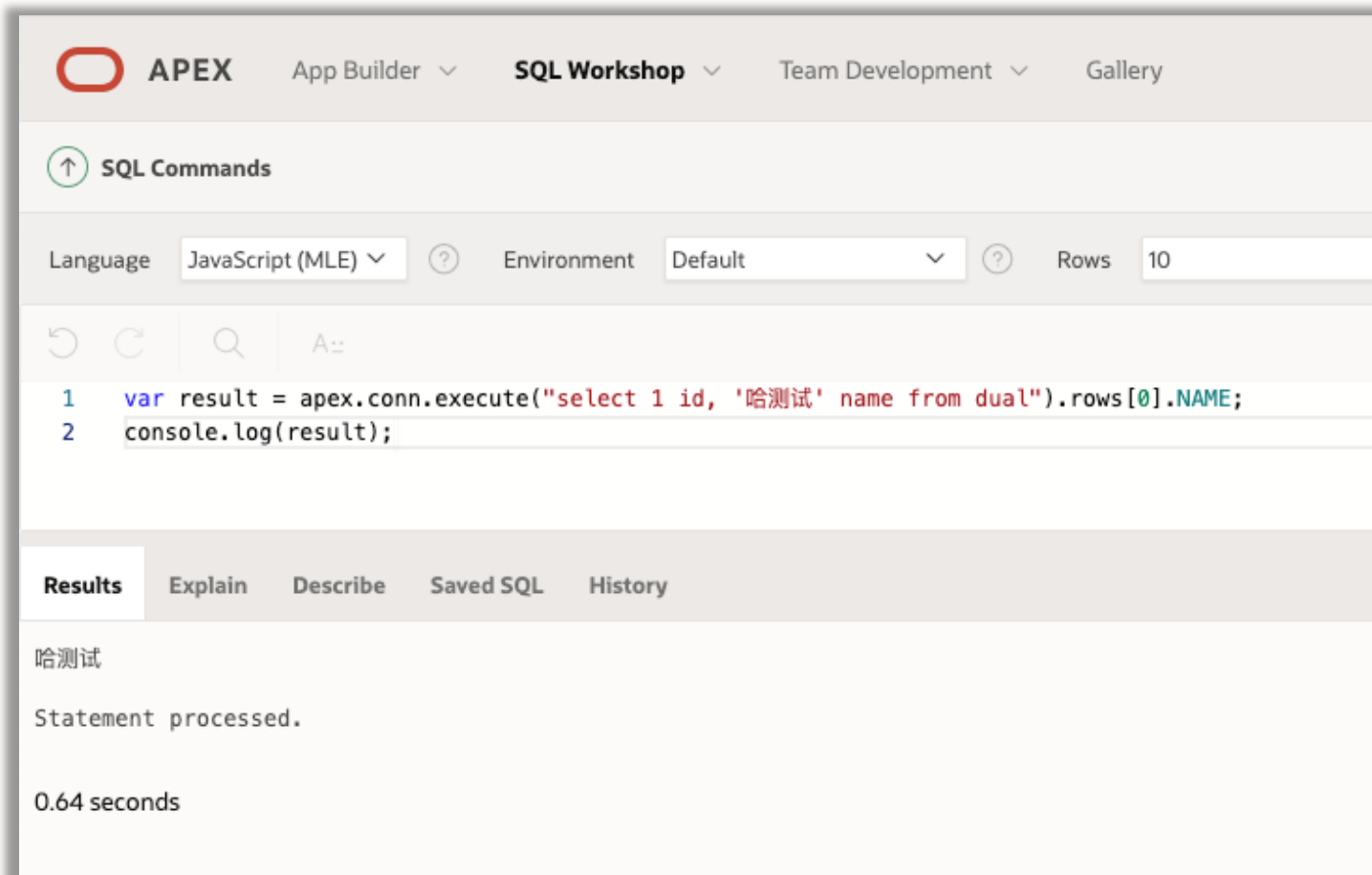


The screenshot shows the APEX SQL Workshop interface. The language is set to JavaScript (MLE). The code being executed is:

```
1 let i = 2, j = 3;  
2 console.log("hello, sum result is: ", i+j)
```

The results section shows the output: "hello, sum result is: 5". Below the output, it says "Statement processed." and "0.00 seconds".

JS使用数据库查询的返回



The screenshot shows the APEX SQL Workshop interface. The language is set to JavaScript (MLE). The code being executed is:

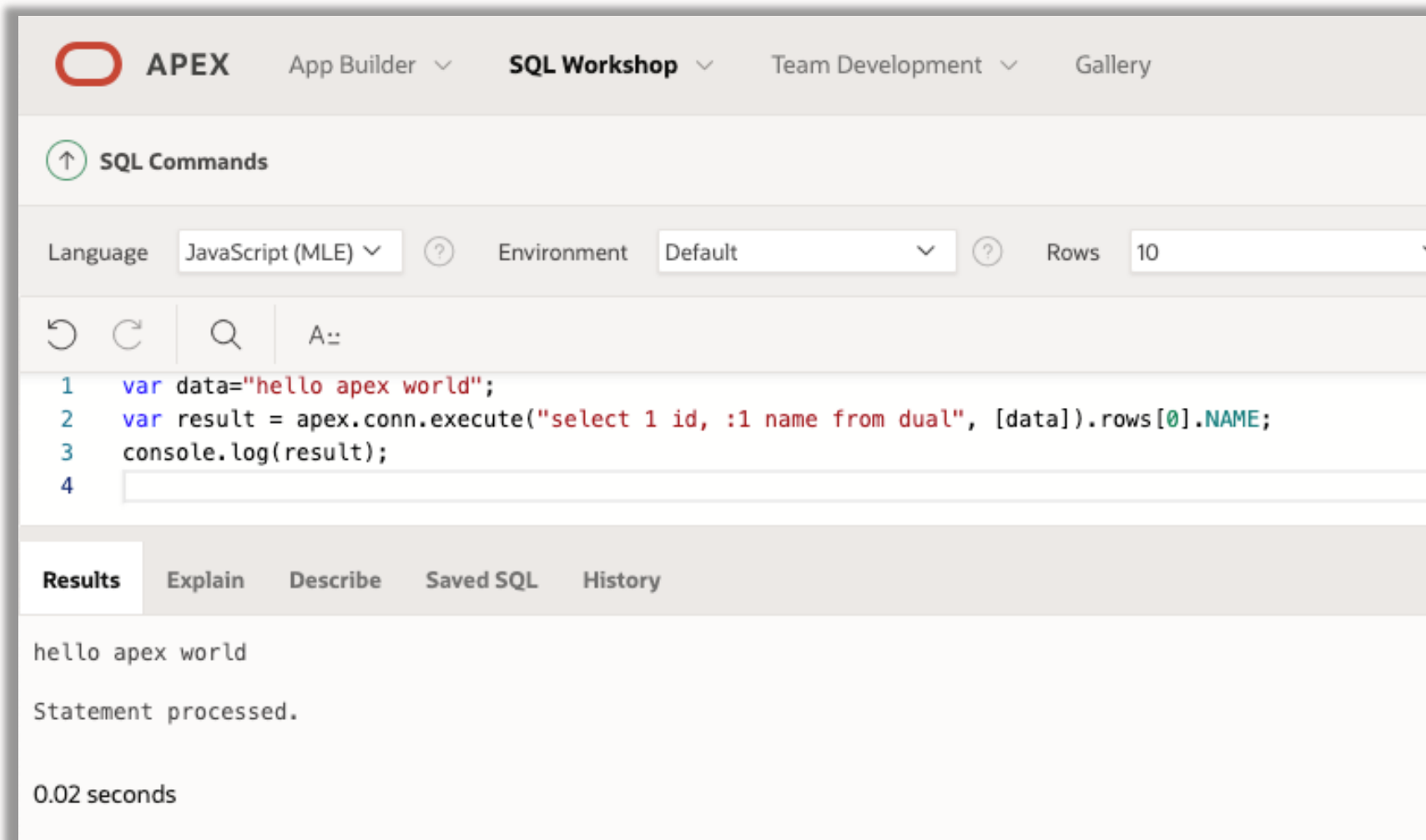
```
1 var result = apex.conn.execute("select 1 id, '哈测试' name from dual").rows[0].NAME;  
2 console.log(result);
```

The results section shows the output: "哈测试". Below the output, it says "Statement processed." and "0.64 seconds".



# MLE JavaScript助力APEX – SQL Commands初体验

如何将MLE中的JS的值  
作为参数传过去



The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are navigation links: APEX, App Builder, SQL Workshop, Team Development, and Gallery. Below this is a section for 'SQL Commands' with a refresh icon. The 'Language' dropdown is set to 'JavaScript (MLE)', 'Environment' is 'Default', and 'Rows' is '10'. The code editor contains the following JavaScript code:

```
1 var data="hello apex world";
2 var result = apex.conn.execute("select 1 id, :1 name from dual", [data]).rows[0].NAME;
3 console.log(result);
4
```

Below the code editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, showing the output: 'hello apex world', 'Statement processed.', and '0.02 seconds'.

# MLE JavaScript助力APEX – 应用中使用MLE JavaScript

APEX通过使用**apex.env**来大大简化了与MLE JavaScript之间的参数传递等工作

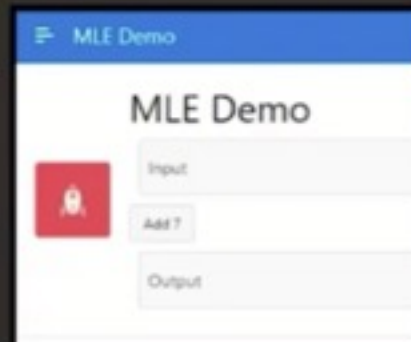
在数据库端通过DBMS\_MLE包来独立运行MLE JS的代码

通过APEX使用DBMS\_MLE

```
DECLARE
  ctx dbms_mle.context_handle_t;
  user_code varchar2(1024) := q'~
    let bindings = require("mle-js-bindings");
    let input = bindings.importValue("P1_INPUT");
    bindings.exportValue("P2_OUTPUT", input + 7);
  ~';
BEGIN
  ctx := dbms_mle.create_context();

  dbms_mle.export_to_mle(ctx, 'P1_INPUT', P1_INPUT);
  dbms_mle.eval(ctx, 'JAVASCRIPT', user_code);
  dbms_mle.import_from_mle(ctx, 'P2_OUTPUT', P2_OUTPUT);

  dbms_mle.drop_context(ctx);
END;
/
```



```
apex.env.P2_OUTPUT = apex.env.P1_INPUT + 7;
```

# MLE JavaScript助力APEX – 如何在APEX中使用第三方JS Module

可以使用数据库原生提供的持久化MLE Module来管理Module。 [官方文档见>> 引入MLE JavaScript Modules](#)

## 1. 数据库端通过PL/SQL创建MLE Module 比如叫`named_exports_module`

```
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};
/
```

如果创建的module要引入其它依赖的module时，可以用import引入。

比如，引入的`namedExports`，是在后面[第2步](#)中定义的环境变量的JS引入名，后面步骤中也会看到如何映射这个`namedExports`与其所对应的MLE Module

```
create OR REPLACE MLE module jsmodule LANGUAGE JAVASCRIPT
As
    import * as myMath from "namedExports";

    export function mySum(a, b) {return myMath.sum(a,b+10);}
    export function myDiff(a, b) {return myMath.difference(a,b);}
/
```



# MLE JavaScript助力APEX – 如何在APEX中使用第三方JS Module

官方文档见>> [引入MLE JavaScript Modules](#)

## 2. 创建MLE Module的环境变量

建立上一步所创建的Module和在JS要引入名之间的映射关系，一个环境变量可以包含有多个Modules

查看MLE环境变量

使用说明

```
import { export1, export2 } from "utilFunctions";  
const { export1, export2 } = await import( "utilFunctions" );
```

Module Owner	MLE Module名	JS端的引入名
XH	NAMED_EXPORTS_MODULE	namedExports
XH	JSMODULE	jsModule

```
CREATE OR REPLACE MLE ENV xh_exports_env  
imports ('namedExports' MODULE named_exports_module,  
'jsModule' MODULE JSMODULE);
```

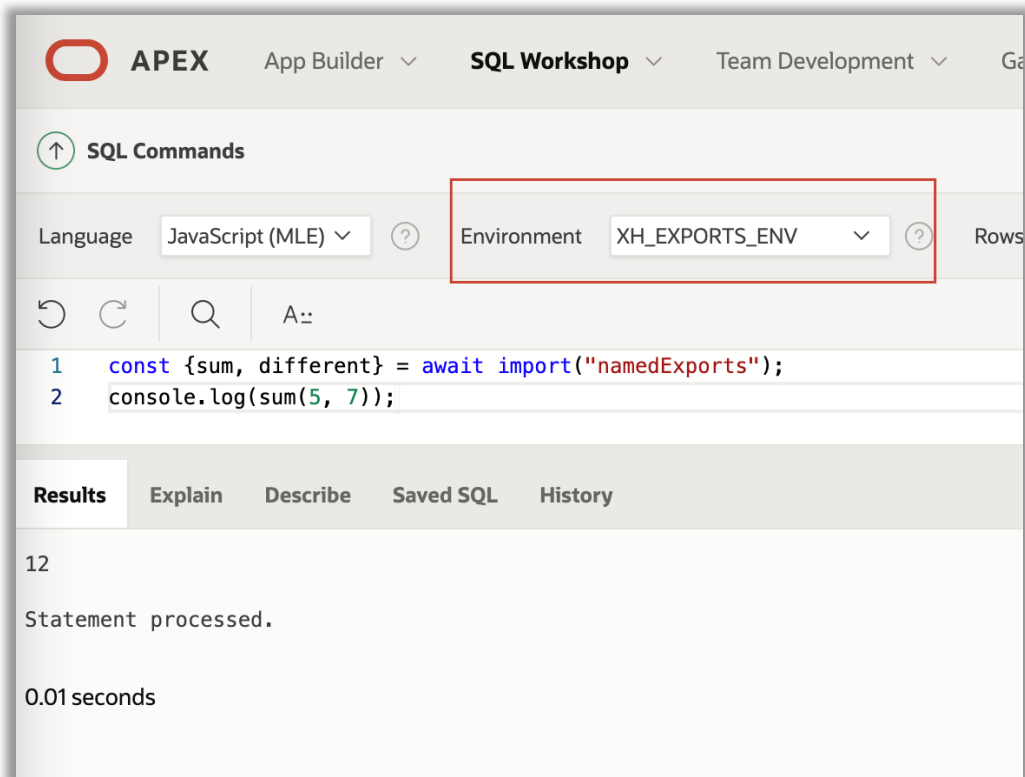


# MLE JavaScript助力APEX – 如何在APEX中使用第三方JS Module

官方文档见>> [引入MLE JavaScript Modules](#)

## 3. JS代码引入MLE Module, 并指定是使用哪个#2步中所创建的MLE Module的环境变量

APEX的SQL Commands中

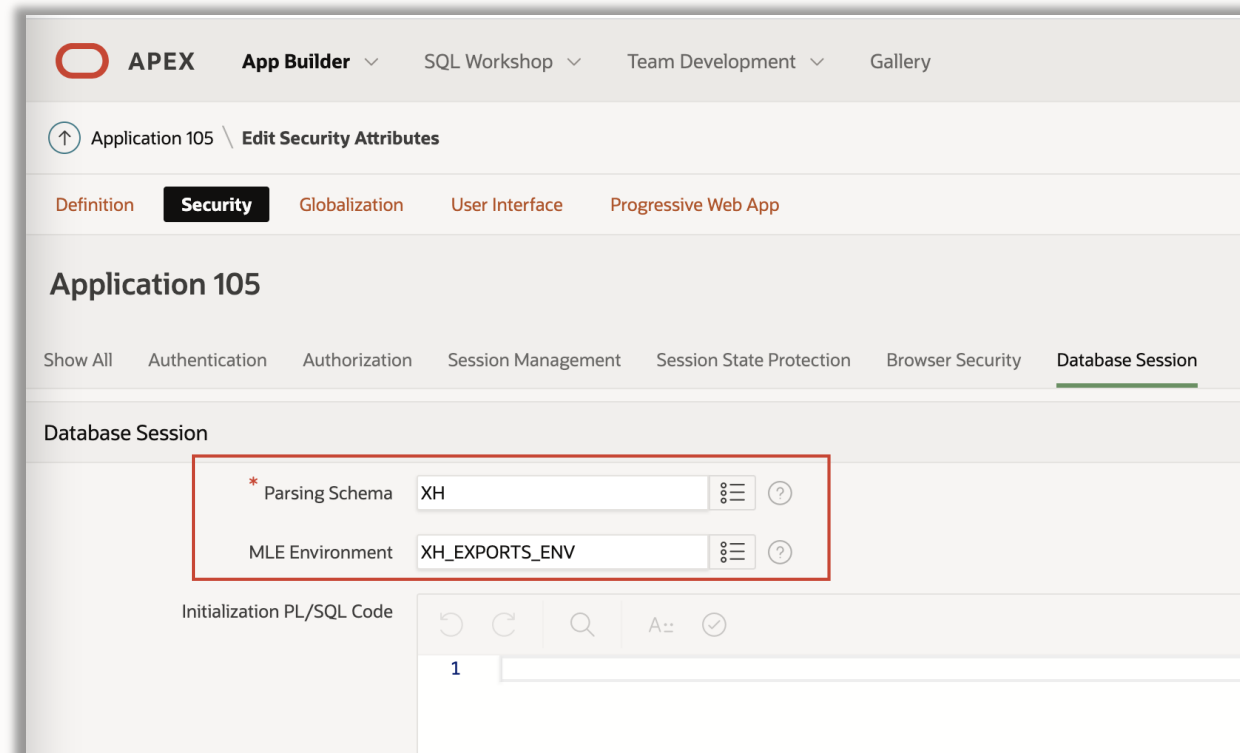


The screenshot shows the APEX SQL Workshop interface. The 'Language' dropdown is set to 'JavaScript (MLE)'. The 'Environment' dropdown is set to 'XH\_EXPORTS\_ENV'. The code editor contains the following JavaScript code:

```
1 const {sum, different} = await import("namedExports");
2 console.log(sum(5, 7));
```

The 'Results' tab shows the output: 'Statement processed.' and '0.01 seconds'.

APEX的应用的Process中使用



The screenshot shows the APEX Application 105 'Edit Security Attributes' page. The 'Security' tab is selected. Under the 'Database Session' section, the 'Parsing Schema' is set to 'XH' and the 'MLE Environment' is set to 'XH\_EXPORTS\_ENV'.

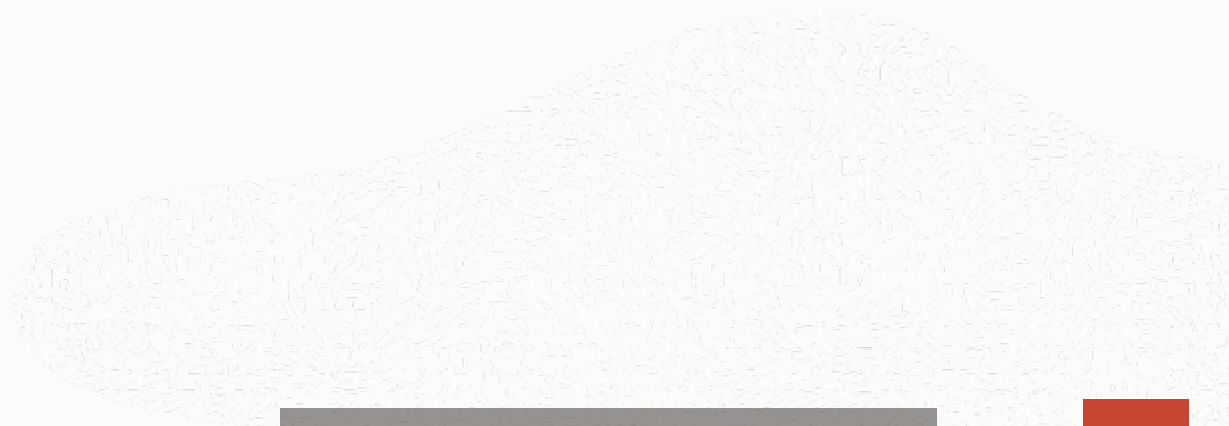


# MLE JavaScript助力APEX – 如何在APEX中使用第三方JS Module

官方文档见>> [引入MLE JavaScript Modules](#)

## 4. 最后就可以直接在Process里面引入了

```
const {sum, different} = await import("namedExports");  
apex.env.P4_JS_OUTPUT = sum(7, 8);
```



# Agenda

- ✓ Oracle MLE介绍

- ✓ MLE、GraalVM是什么
- ✓ MLE JavaScript在数据库端实操

- ✓ MLE JavaScript助力APEX

- ✓ SQL Workshop中执行JS代码
- ✓ APEX应用中使用MLE JS
- ✓ 为什么需要在服务端执行JS
- ✓ 如何在APEX中使用第三方JS Module

- MLE Demo 分享

- 校验、二维码生成、Markdown转HTML、HTML过滤器、图片处理、情感分析

# APEX + MLE Demo

- <https://github.com/stefandobre/apex-mle-demo>

## Examples

Form Validation



QR Code Generation



Markdown to HTML



HTML Sanitization



Image Manipulation



Sentiment Analysis



- [APEX + Server-Side JavaScript hands-on资料](#)  
直接用原生的持久化MLE Module来实现

# MLE Demo分享

The screenshot shows the 'QR Code Generation' module. The 'Input' field contains the URL 'apex.oracle.com'. A 'Create' button is visible. The 'Output' section displays a QR code.

The screenshot shows the 'Analyze Content' module. The 'Content' field contains the text 'I think APEX + MLE is not good enough'. The 'Score' is displayed as -3. An 'Analyze' button is at the bottom right.

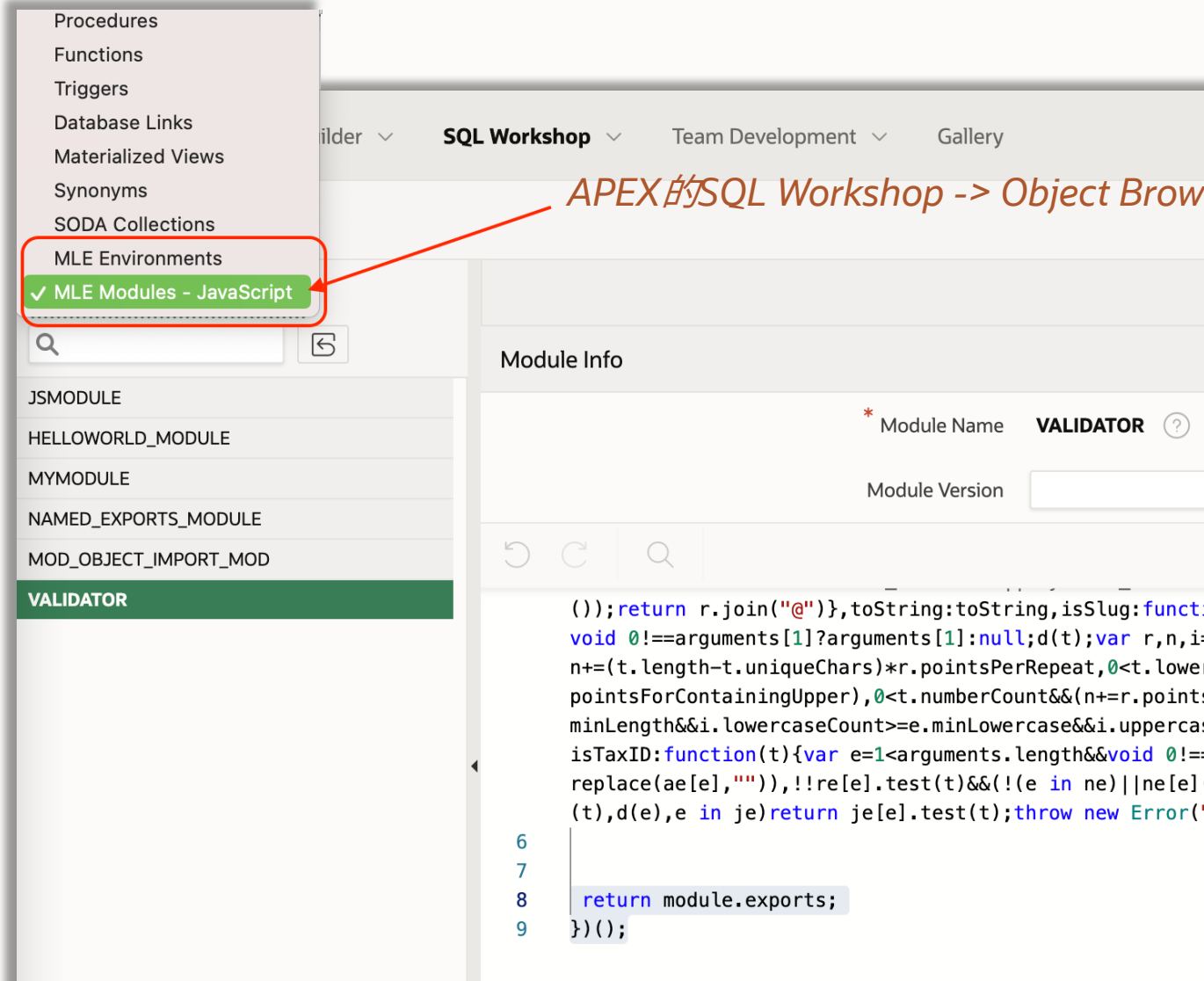
The screenshot shows the 'Markdown to HTML' module. The 'Input' field contains the following markdown: `## title  
### my part  
  
**bold**  
*italic*`. The 'Output' section shows the rendered HTML: `<h2>title</h2>  
<h3>my part</h3>  
<p><strong>bold</strong>  
<em>italic</em></p>`. Below the output, the text is displayed as: **title**, **my part**, and **bold italic**. A 'Convert' button is at the bottom.

The screenshot shows the 'Image Manipulation' module. The 'Input' section has a 'Base Image' field with a 'Drag and Drop' area. The 'Crop' section has fields for X, Y, Width, and Height. The 'More' section has a 'Flip' option with 'Horizontal' checked and 'Vertical' unchecked. The 'Output' section shows the result of the image manipulation. A red arrow points from the input image to the output image, with a red circle highlighting the watermark. A red text label below the arrow reads: '按logo加水印后, 并进行水平翻转'.



# MLE Demo分享

## 建MLE Module



APEX的SQL Workshop -> Object Browser -> 左边的下拉列表 里选择

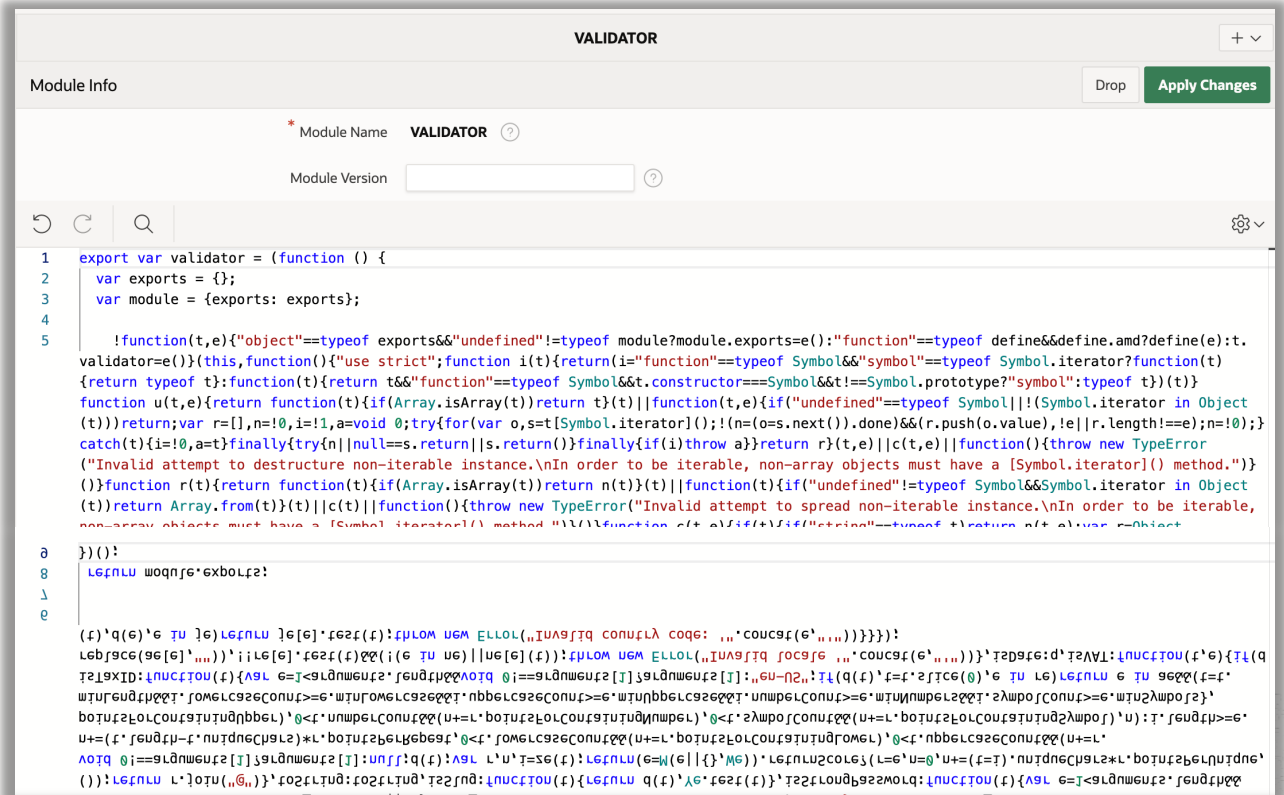


# MLE Demo分享

将代码包裹成标准的js module形式

```
export var validator = (function () {  
  var exports = {};  
  var module = {exports: exports};  
  // ...第三方JS库，此处则为validator.min.js的内容...  
  return module.exports;  
})();
```

以validator为例，最后生成好的validator的MLE Module





# MLE Demo分享

以validator为例

```
requireModule('validator').isEmail(apex.env.P15_EMAIL)
```

改为

```
(await import("validator")).validator.isEmail(apex.env.P15_EMAIL)
```



## **立即扫码进行1V1 免费咨询**

**2023年10月，MySQL 5.7 将终止官方支持和更新。**  
立刻升级至更快、更稳定、更安全的 MySQL 8.0 /  
MySQL Database Service，获取 300+ 项新特性，  
使开发更加灵活和高效，更好的满足业务发展需求。

**免费咨询热线：  
400-699-8888**

\* 活动最终解释权归甲骨文公司所有

# 深入了解Oracle Key Vault 密钥保险箱(OKV)

## 数据安全实战演练系列(四)



### 钟远

- 资深数据安全专家
- 10年以上系统安全运维和管理库管理经验
- 电信行业背景，在智能运维与数据安全架构方面经验丰富

### 内容简介

深入了解Oracle Key Vault密钥保险箱(OKV)，理解OKV的原理、架构、功能以及如何部署使用等。通过现场DEMO演示掌握OKV的使用，内容主要包括：

- OKV概述
- OKV应用场景
- OKV动手演示
- OKV界面以及概念的介绍
- OKV管理TDE Master key
- OKV管理数据库密码



Zoom直播

直播时间：9月8日 11:00 - 12:00

扫描二维码进入直播

Zoom ID: 957 9669 6723

密码：20212023



微信扫一扫预约



数据库和云讲座群



甲骨文云技术公众号



技术专家1V1深入交流