

# IntelliJ 深色主题设置



我在 `new com.mysql.cj.jdbc.Driver` 的时候自动注册了驱动吗？



当你使用 `new com.mysql.cj.jdbc.Driver()` 创建驱动实例时，实际上会自动注册该驱动。以下是具体的情况：

## 自动注册机制

### 1. 构造函数中的注册：

- 在 `com.mysql.cj.jdbc.Driver` 类的构造函数中，通常会有一段代码调用 `DriverManager.registerDriver(this)`，这会将驱动实例注册到 `DriverManager` 中。
- 这意味着当你创建驱动实例时，它会自动向 `DriverManager` 注册自己。

### 2. 使用 `Class.forName()`：

- 更常用的方式是通过 `Class.forName("com.mysql.cj.jdbc.Driver")` 来加载驱动。这个方法也会触发静态代码块的执行，其中同样会调用 `DriverManager.registerDriver` 来注册驱动。

## 实际效果

- 不需要手动注册：**因为无论是使用 `new` 还是 `Class.forName()`，驱动都已自动注册，你通常不需要显式地调用注册方法。
- 简化代码：**使用 `Class.forName()` 是推荐的做法，它更加简洁并符合 JDBC 4.0 及以上版本的规范，利用了 SPI (Service Provider Interface) 机制，允许驱动在类加载时自动注册。

## 总结

是的，使用 `new com.mysql.cj.jdbc.Driver()` 会自动注册驱动，但通常推荐使用 `Class.forName()` 来实现驱动的加载和注册，这样代码更加清晰和简洁。



`Class.forName()`  
这个有什么用？



使用 `Class.forName()` 的主要作用和好处包括：

## 1. 自动注册驱动

- 调用 `Class.forName("com.mysql.cj.jdbc.Driver")` 时，会自动加载驱动类，并在类的静态代码块中调用 `DriverManager.registerDriver(this)`。这确保了驱动被正确注册，无需手动管理。

## 2. 简化代码

- `Class.forName()` 提供了一种简单的方式来加载类，不需要手动实例化对象。这样可以减少代码量，增强可读性。

## 3. 与 JDBC 4.0 及以上版本的兼容性

- 从 JDBC 4.0 开始，驱动支持 SPI (Service Provider Interface)，允许驱动在 JAR 包中自动注册。虽然仍然可以手动加载驱动，但使用 `Class.forName()` 是一种更符合规范的方法。

## 4. 适应不同的驱动

- 如果你的应用需要支持多种数据库，只需修改 `Class.forName()` 中的类名，而其他代码保持不变。这使得你的应用具有更好的可扩展性和适应性。

## 5. 避免 ClassNotFoundException

- 当使用 `Class.forName()` 加载类时，如果类不存在，会抛出 `ClassNotFoundException`，这可以帮助你应用启动时就检测到驱动是否存在，避免运行时错误。

## 总结

`Class.forName()` 的使用不仅自动处理驱动注册，还使代码更简洁，并提高了与 JDBC 规范的兼容性。它为处理不同类型的数据库提供了灵活性，使得数据库连接的管理变得更加方便和可靠。