**Lập lịch với thư viện Quartz Scheduler**

1. **Tổng quan**
2. **Tổng quan**

**Quartz Scheduler**: thư viện mã nguồn mở để lập lịch và quản lý công việc trong các ứng dụng Java.

Cho phép lập lịch và thực hiện công việc vào những thời điểm hoặc khoảng thời gian cụ thể.

Quartz sử dụng cơ chế lập lịch mạnh mẽ và linh hoạt, cho phép lên lịch các công việc dựa trên việc hoàn thành thành công các công việc, hỗ trợ tích hợp với nhiều kho dữ liệu như JDBC và JPA, và cung cấp API để giám sát và quản lý công việc.

1. **Các thành phần chính**
2. **Scheduler**

Quản lý toàn bộ quá trình lên lịch và thực thi các job. Khi đươc bắt đầu, Scheduler tìm và kích hoạt các trigger thõa mãn điều kiện và yêu cầu thực thi các job liên quan.

Hỗ trợ phân cụm (clustering), cho phép các ứng dụng phân tán cùng sử dụng chung một Scheduler mà không bị trùng lặp job.

1. **Job**

Đại diện cho các tác vụ được thực hiện. Các class thực hiện các tác vụ phải triển khai phương thức execute() chứa logic thực hiện công việc.

Một instance của job sẽ được Quartz tạo ra mỗi khi job được thực thi nếu không cấu hình JobDetail.

1. **JobDetail**

Là một đối tượng cấu hình của Quirtz, chứa thông tin chi tiết về job bao gồm thuộc tính và dữ liệu mà job cần để thực thi.

JobDetail liên kết với một Job, chỉ định job sẽ làm gì khi được kích hoạt bởi trigger.

1. **Trigger**

Trigger xác định khi nào và tần suất một job sẽ được thực hiện, là thành phần kích hoạt các job.

Có 2 loại trigger:

* SimpleTrigger: các tác vụ chạy theo khoảng thời gian đơn giản.
* CronTrigger: các tác vụ lên lịch theo biểu thức cron.

1. **Calender**

Cho phép định nghĩa các khoảng thời gian mà job sẽ được loại trừ, đảm bảo job không chạy trong các khoảng thời gian đó.

1. **JobStore**

Xác định nơi lưu trữ trạng thái của job và trigger. Có 2 loại JobStore:

* RAMJobStore: lưu trữ trong bộ nhớ tạm
* JDBCJobStore: lưu trong cơ sở dữ liệu

1. **Listener**

Các thành phần theo dõi và xử lý sự kiện trong vòng đời của job, trigger hoặc toàn bộ scheduler:

* JobListener: Theo dõi các sự kiện của job, ví dụ như khi job được thực thi hoặc khi gặp lỗi.
* TriggerListener: Theo dõi các sự kiện của trigger, ví dụ như khi trigger được kích hoạt hoặc bị bỏ lỡ.
* SchedulerListener: Theo dõi các sự kiện của toàn bộ scheduler, ví dụ như khi scheduler khởi động hoặc dừng.

1. **Đánh giá**

**Ưu điểm:**

* Linh hoạt, mạnh mẽ
* Quản lý tác vụ phức tạp: hỗ trợ phân nhóm, ưu tiên tác vụ, quản lý trạng thái job
* Khả năng lưu trữ và phục hồi
* Hỗ trở cluster

**Nhược điểm:**

* Cấu hình phức tạp
* Overhead có thể cao hơn

1. **So với công cụ khác (Spring Scheduler)**

Spring Scheduler được tích hợp sẵn trong Spring Framework để hỗ trợ các tác vụ lập lịch mà không cần thư viện bên ngoài.

So với tính năng được tích hợp sẵn, Quartz Scheduler hỗ trợ tốt hơn trong các ứng dụng yêu cầu lập lịch phức tạp

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Ưu điểm** | **Nhược điểm** | **Phù hợp** |
| Quartz Scheduler | Mạnh mẽ, hỗ trợ cluster, quản lý trạng thái job,... | Cấu hình phức tạp, overhead cao | Hệ thống lớn, tác vụ phức tạp |
| Spring Scheduler | Dễ cấu hình, tích hợp sẵn | Hạn chế tính năng, không quản lý trạng thái job | Ứng dụng, tác vụ đơn giản |

1. **Sử dụng**
2. **Cấu hình**

- Cấu hình dependency (Gradle): 