Date : 22/09/2020

Spring Boot 7AM

Mr. RAGHU

===================================================================

**Scheduling using Spring Boot**

Maven Session:

https://www.youtube.com/c/NareshIT/search?query=raghu+maven

fixedDelay : An Exact time gap given between first method finish time to next method start time

fixedRate : Max time gap given to start next method call,

including (previous) method execution time.

Q) What is the difference between fixedDelay and fixedRate?

A) fixedDelay provides exact time between two methods.

fixedRate provides max time gap including method execution time.

--Ex--

package in.nareshit.raghu.service;

import java.util.Date;

import org.springframework.scheduling.annotation.Scheduled;

import org.springframework.stereotype.Component;

@Component

public class ExportData {

@Scheduled(fixedRate = 2000) // 2sec =2000 mill sec

public void execute() {

System.out.println("DONE " + new Date());

}

}

**Scheduling using cron expression**

===================================================================

=> cron is a expression used to indicate date and time in Unix based Operating System

=> cron looks like:

SEC MIN HRS DAY MONTH WEEKDAY

0-59 0-59 0-23 1-31 1-12 SUN-SAT

=> Symbols used in cron expression

\* = any / all

- = range

, = possible values

/ = period of time

? = any or all applied only for DAY or WEEK DAY when month is provided

=> By default it indicates PointOfTime, we can also use as period of time.

----Example--------

1. 0 0 9 \* \* \*

=> Execute given task in a loop every day 9:00:00 AM

ex: 22nd Sep - 9AM --- method is called

Next 23rd Sep - 9AM --- method is called

Next 24th Sep - 9AM --- method is called

2. 0 0 18 \* \* \*

=> Execute given task in a loop every day 6PM

3. 0 0 6,18 \* \* \*

=> Execute given task in a loop every day 2 times 6AM and 6PM

4. 0 0 10,14 \* \* \*

=> Execute given task in a loop every day 2 times 10AM and 2PM

5. 0 30 8 \* \* \*

=> Execute given task in a loop every day 1 time : 8:30:00 AM

6. 0 10 \* \* \* \*

=> Every hrs 10th min task executed. (It is not 10 mins gap)

Task started at : 9:10:00 AM

Next : 10:10:00 AM (next hr 10th min)

Next : 11:10:00 AM

7. 10 \* \* \* \* \*

@Component

public class ExportData {

@Scheduled(cron = "10 \* \* \* \* \*")

public void execute() {

System.out.println("DONE " + new Date());

}

}

=> Every Minute 10th sec (it is not 10 sec gap)

Task Started at : 9:00:10 AM

Next : 9:01:10 AM (next min 10th sec)

Next : 9:02:10 AM (next min 10th sec)

8. \* \* \* \* \* \*

=> Every sec execut task.

@Component

public class ExportData {

@Scheduled(cron = "\* \* \* \* \* \*")

public void execute() {

System.out.println("DONE " + new Date());

}

}

9. 0 \* 9 \* \* \*

(Invaild Expression)

=> If we provide hrs then must provide mins and sec too

=> If we provide mins then must provide sec too.

10. 0 0 9 1 \* \*

=> Every month 1st - 9:00:00 AM

11. 0 0 8-11 \* \* \*

0 0 8,9,10,11 \* \* \*

=> Every day 4 times executed

8:00:00 AM

9:00:00 AM

10:00:00 AM

11:00:00 AM

12. 0 0 9 1 1 SUN

=> Execute on 1st JAN 9:00:00 AM if given day is SUNDAY only.

Next execution is on: 1st JAN 9:00:00 AM 2023 (SUN)

13. 0 0 9 \* 8 \*

(INVAILD EXPRESSION)

=> WHEN MONTH IS PROVIDED, WE MUST PROVIDE DAY AND WEEK DAY BOTH

=> ELSE DO YOU WANT TO INDICATE EVERY DAY AND EVERY WEEK DAY THEN USE ?

Vaild format is : 0 0 9 ? 8 ?

14. 59 59 23 31 12 ?

=> Every Year, Dec-31st - 11:59:59 PM

------------------------- Period of time (/ = gap)-------------------------

use Symbol / at any position that indicates period (not at WEEK DAY)

15. 10 \* \* \* \* \*

=> Every min 10th sec

\*/10 \* \* \* \* \*

=> Every 10 sec gap

--Example--

@Component

public class ExportData {

@Scheduled(cron = "\*/10 \* \* \* \* \*")

public void execute() {

System.out.println("DONE " + new Date());

}

}

16. 0 0/10 9 \* \* \*

=> Start 9:00:00 AM Every day

=> gap time added +10 mins

Next: 9:10:00 AM

Next: 9:20:00 AM

Next: 9:30:00 AM

Next: 9:40:00 AM

Next: 9:50:00 AM

Q) What if method takes more then given time?

A) Consider at given point next method execution starts, it will never check about last

method (finished or not?). A new thread is created.

17. 0 0 9/3 1 \* \*

=> Start 1st of every month 9:00:00AM

Add Gap of +3 hrs

Next: 12 PM, Next: 3PM, next: 6PM, Next: 9PM

If we take next 12 AM comes under 2nd Date. So, not considered.

\*)https://docs.spring.io/spring/docs/current/javadoc-api/org/springframework/scheduling/support/CronSequenceGenerator.html

"0 0 \* \* \* \*" = the top of every hour of every day.

"\*/10 \* \* \* \* \*" = every ten seconds.

"0 0 8-10 \* \* \*" = 8, 9 and 10 o'clock of every day.

"0 0 6,19 \* \* \*" = 6:00 AM and 7:00 PM every day.

"0 0/30 8-10 \* \* \*" = 8:00, 8:30, 9:00, 9:30, 10:00 and 10:30 every day.

"0 0 9-17 \* \* MON-FRI" = on the hour nine-to-five weekdays

"0 0 0 25 12 ?" = every Christmas Day at midnight