Java-Success.com

Industrial strength Java/JEE Career Companion to open more doors

Register | Login | Logout | Contact Us

search here … Go

Home | Java FAQs | 600+ Java Q&As | Career | Tutorials | Member | Why?

Can u Debug? | Java 8 ready? | Top X | Productivity Tools | Judging Experience?

Home › Interview › Java Architecture Interview Q&A › 1. Asynchronous processing in Java real life examples – part-1

1. Asynchronous processing in Java real life examples – part-1

Posted on September 2, 2015 by Arulkumaran Kumaraswamipillai

The scenario based questions are very popular with the job interviewers, and some scenario based questions are related to decoupling and **asynchronous** (aka **non-blocking**) processing in Java with message queues/topics, database tables where one process inserts data and another process consumes the inserted data, and **Future** objects in Java multi-threading.

Q1. Can you describe instances where you used asynchronous processing in your application?
A1. The asynchronous processing is very handy for decoupling systems and for boosting performance and scalability. Here are a few examples.

Example 1: Performing time consuming tasks asynchronously

An online application with a requirement to produce time consuming reports or a business process (e.g. rebalancing accounts, aggregating hierachical information, producing compliance reports that runs through hundreds of rules, etc) could benefit from making these long running operations asynchronous. Once the reports or the long running business process is completed, the outcome can be communicated to the user via emails or asynchronously refreshing the web page via techniques known as "**server push**" (i.e via Servlet 3.0 asynchronous) or "**client pull**" (i.e. meta refresh tag). A typical example would be

a) A user makes a request for an aggregate report or a business process like rebalancing his/her portfolios.

b) The user input can be saved to a database table for a separate process to periodically pick it up and process it asynchronously.

c) The user could now continue to perform other functionalities of the website without being blocked.

d) A separate process running on the same machine or different machine can periodically scan the table for any entries and produce the necessary reports or execute the relevant business process. This could be a scheduled job that runs once during off-peak or every 10 minutes. This depends on the business requirement.

e) Once the report or the process is completed, notify the user via emails or making the report available online to be downloaded.

Example 2: RESTFul Web Service utilizing store, play and replay asynchronously mechanism

Define two RESTful web services that take payloads and process them asynchronously. This means the web service requests are not blocked.

1) The “submitCreateCase” stores the payload in a database table for auditing and replaying on failure purposes. This service is invoked by a user via a GUI when submitting case details (i.e. submit some data) that needs to be posted via a third-party web service. If invocation of the third-party web service fails, the status gets updated on the table with the payload as “FAILED”, and can be retried via “replayFailedRequests” service.

2) The “replayFailedRequests” service to retry the payload with status being “FAILED” in the database table. This service gets periodically invoked by a scheduled job running say every 15 minutes to re-invoke the third-party RESTFul web service by retrieving the saved payload from the database table.

Here is some sample code snippets:

```
//...package & imports

@Controller
@RequestMapping(value = "/v1/myapp/", produces =
public class MyAppEndpointControllerImpl impleme

    @Inject
    @Named("createCaseExecutor")
    private Executor createCaseExecutor;

    @Inject
    @Named("replayExecutor")
    private Executor replayExecutor;

    @Inject
    private MyAppReplayService replayService;

    @Inject
    private MyAppClientService myAppClientServic

    @Inject
    private UniqueIdentifierGeneratorService uid

    private static final Logger LOG = LoggerFact

    //invoked via a User Interface by a user sub
    @Override
    @ResponseStatus(HttpStatus.ACCEPTED)
    @RequestMapping(value = "/submitCase", metho
```

100+ Java pre-interview coding tests

open all | close all
Can you write code?
♦ Complete the given
Converting from A to I
Designing your classe
Java Data Structures
Passing the unit tests
What is wrong with th
Writing Code Home A
Written Test Core Jav
Written Test JEE (1)

How good are your .....?

open all | close all
Career Making Know-
Job Hunting & Resum

```
    @ResponseBody public CreateCaseResponseAsync
        @RequestBody final CreateCaseRequestAsyn

        final String auditTrackingId = uidGenera

        //spawn a new thread
        createCaseExecutor.execute(new Runnable(
            @Override
            public void run() {
                LOG.debug("submitCreateCase() in
                final CreateCaseRequest createCa
                //store request payload, status,
                //invoke third-party service asy
                myAppClientService.savePayload(c
                callbackProcessor.processCallbac
            }
        });

        // return the DTO including the auditTra
        final CreateCaseResponseAsync createCase
        createCaseResponseAsync.setAuditTracking
        return createCaseResponseAsync;
    }

    //invoked by a scheduled job every 15 minute
    @Override
    @ResponseStatus(HttpStatus.ACCEPTED)
    @RequestMapping(value = "/replay", method = 
    public void replayFailedRequests(
        @RequestParam(required = false) final In


        //the request payload and statuses are s
         //spawn a new thread
        replayExecutor.execute(new Runnable() {
            @Override
            public void run() {
                LOG.info("replayFailedRequests()
                replayService.replayFailedReques
            }
        });
    }
}
```

The executors can be configured via Java based Spring configs.

```

    @Bean
    public ExecutorService replayExecutor() {
        return Executors.newSingleThreadExecutor
    }

    @Bean
    public ExecutorService createCaseExecutor()
       return Executors.newCachedThreadPool();
    }

```

The callbackProcessor that creates a new nested transaction, and makes the actual third-party REST call.

```
@Named
@Transactional
public class MyAppCallbackProcessorImpl implemen
    private static final Logger LOG = LoggerFact

    private final RestTemplate restTemplate;
    private final TransactionService transaction
    private HttpHeaders httpHeaders;


    @Inject
    public MyAppCallbackProcessorImpl(final Rest
        this.restTemplate = restTemplate;
        this.transactionService = transactionSer
    }

    @Override
    public void processCallback(final String cal
        try {
            transactionService.executeInNewTrans
                @Override
                public Void doInTransaction() th
                    sendCreateCaseRequest(callba
                    return null;
                }
            });
        } catch (final Exception ex) {
            LOG.error(String.format("An error ha
                + " '%s': %s. This error is curr
        }
    }

    private void sendCreateCaseRequest(final Str
        if (StringUtils.isNotBlank(callbackUrl))
            LOG.info("Sending PUT request to MyA
            final HttpEntity<CreateCaseRequest>
                new HttpEntity<CreateCaseRequest
            restTemplate.put(URI.create(callback
        }
    }

    @Override
    public void afterPropertiesSet() throws Exce
        httpHeaders = new HttpHeaders();
        httpHeaders.setAccept(ImmutableList.of(A
        httpHeaders.setContentType(APPLICATION_X
    }
}
```

More asynchronous processing in Java examples part 2.

Popular Member Posts

- ♦ 11 Spring boot interview questions & answers

850 views

♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers

768 views

001A: ♦ 7+ Java integration styles & patterns interview questions & answers

399 views

18 Java scenarios based interview Questions and Answers

387 views

♦ 7 Java debugging interview questions & answers

308 views

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

305 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

297 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

293 views

♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers

246 views

001B: ♦ Java architecture & design concepts interview questions & answers

204 views

Bio | **Latest Posts**

Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

About Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

‹ 03: Identifying and fixing NonUniqueObjectException in Hibernate

04: Identifying and fixing LazyInitializationException in Hibernate ›

Posted in Java Architecture Interview Q&A**,** Judging Experience Interview Q&A**,** member-paid

Empowers you to open more doors, and fast-track

Technical Know Hows

☀ Java generics in no time ☀ Top 6 tips to transforming your thinking from OOP to FP ☀ How does a HashMap internally work? What is a hashing function?
☀ 10+ Java String class interview Q&As ☀ Java auto un/boxing benefits & caveats ☀ Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect

Non-Technical Know Hows

☀ 6 Aspects that can motivate you to fast-track your career & go places ☀ Are you reinventing yourself as a Java developer? ☀ 8 tips to safeguard your Java career against offshoring ☀ My top 5 career mistakes

Prepare to succeed

☀ Turn readers of your Java CV go from “Blah blah” to “Wow”? ☀ How to prepare for Java job interviews? ☀ 16 Technical Key Areas ☀ How to choose from multiple Java job offers?

Select Category ▼

© Disclaimer