Evaluating Java Programming Paradigms

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science
Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA
Learning Objectives in this Part of the Lesson

• Understand the key benefits & principles underlying reactive programming
• Know the Java reactive streams API & popular implementations of this API
• Learn how Java reactive streams maps to key reactive programming principles
• Recognize how reactive programming compares with other Java paradigms
Learning Objectives in this Part of the Lesson

- Understand the key benefits & principles underlying reactive programming
- Know the Java reactive streams API & popular implementations of this API
- Learn how Java reactive streams maps to key reactive programming principles
- Recognize how reactive programming compares with other Java paradigms
- Be aware of the pros & cons of reactive streams platforms
Comparing Reactive Programming with Other Paradigms
Comparing Reactive Programming with Other Paradigms

- Reactive programming is one of several Java programming paradigms
Comparing Reactive Programming with Other Paradigms

• Reactive programming is one of several Java programming paradigms

```java
byte[] downloadContent(URL url) {
    byte[] buf = new byte[BUFSIZ];
    ByteArrayOutputStream os = new ByteArrayOutputStream();
    InputStream is = url.openStream();
    for (int bytes;
         (bytes = is.read(buf)) > 0;)
        os.write(buf, 0, bytes);
    ...}
```
Comparing Reactive Programming with Other Paradigms

- Reactive programming is one of several Java programming paradigms

```
List<Image> filteredImages =
    getInput()
    .parallelStream()
    .filter(not(this::urlCached))
    .map(this::downloadImage)
    .flatMap(this::applyFilters)
    .collect(toList());
```
Comparing Reactive Programming with Other Paradigms

- Reactive programming is one of several Java programming paradigms.

```java
CompletableFuture
  .supplyAsync(reduce)
  .thenApply
    (BigFraction::toMixedString)
  .thenAccept
    (System.out::println);
```
Comparing Reactive Programming with Other Paradigms

- Reactive programming is one of several Java programming paradigms

```java
List<URL> urls = getInput().stream()
    .map(checkUrlCachedAsync)
    .map(downloadImageAsync)
    .flatMap(applyFiltersAsync)
    .collect(toFuture())
    .thenApply(logResults)
    .join(); ...
```
Comparing Reactive Programming with Other Paradigms

- Reactive programming is one of several Java programming paradigms

```java
Flux
    .fromIterable(Options.instance().getUrlList())
    .parallel(parallelism)
    .runOn(scheduler)
    .map(downloadAndStoreImage)
    .sequential()
    .collectList()
    .block();
```
Pros & Cons of Java Reactive Streams Platforms
Pros & Cons of Java Reactive Streams Platforms

- Java reactive streams platforms implement reactive programming principles to achieve several benefits.
Pros & Cons of Java Reactive Streams Platforms

- Java reactive streams platforms implement reactive programming principles to achieve several benefits
- Support concurrency with a minimal number of threads
Java reactive streams platforms implement reactive programming principles to achieve several benefits:

- Support concurrency with a minimal number of threads
- Scale up performance with fewer hardware resources

Pros & Cons of Java Reactive Streams Platforms

- Reactive programming is not appropriate for all situations.

See www.youtube.com/watch?v=z0a0N9OgaAA
End of Evaluating Java Programming Paradigms