Programming with Java Futures

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

• Motivate the need for Java futures by understanding the pros & cons of synchrony & asynchrony

• Know how Java futures provide the foundation for completable futures in Java

• Understand how to program with Java futures

String f1 = "62675744/15668936";
String f2 = "609136/913704";

Callable<BigFraction> call = () -> {
    BigFraction bf1 = new BigFraction(f1);
    BigFraction bf2 = new BigFraction(f2);
    return bf1.multiply(bf2);
};

Future<BigFraction> future = commonPool().submit(call);
...

BigFraction res = future.get();
Programming with Java Futures
Programming with Java Futures

- Example of using Java Future via a Callable & the common fork-join pool

```java
String f1 = "62675744/15668936";
String f2 = "609136/913704";

Callable<BigFraction> call = () -> {
    BigFraction bf1 =
        new BigFraction(f1);
    BigFraction bf2 =
        new BigFraction(f2);
    return bf1.multiply(bf2); }

Future<BigFraction> future =
    commonPool().submit(call);
...
BigFraction result =
    future.get();
```

See github.com/douglascraigschmidt/LiveLessons/tree/master/Java8/ex8
Programming with Java Futures

- Example of using Java Future via a Callable & the common fork-join pool

```java
String f1 = "62675744/15668936";
String f2 = "609136/913704";

Callable<BigFraction> call = () -> {
    BigFraction bf1 =
        new BigFraction(f1);
    BigFraction bf2 =
        new BigFraction(f2);
    return bf1.multiply(bf2);
};

Future<BigFraction> future =
    commonPool().submit(call);
...

BigFraction result =
    future.get();
```

Callable is a two-way task that returns a result via a single method with no arguments

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/Callable.html
Programming with Java Futures

• Example of using Java Future via a Callable & the common fork-join pool

String f1 = "62675744/15668936";
String f2 = "609136/913704";

Callable<BigFraction> call = () -> {
    BigFraction bf1 = 
        new BigFraction(f1);
    BigFraction bf2 = 
        new BigFraction(f2);
    return bf1.multiply(bf2); }

Future<BigFraction> future = 
    commonPool().submit(call);
...
BigFraction result = 
    future.get();

See "Overview of Java Lambda Expressions & Method References"
### Programming with Java Futures

- Example of using Java Future via a Callable & the common fork-join pool

```java
String f1 = "62675744/15668936";
String f2 = "609136/913704";

Callable<BigFraction> call = () -> {
    BigFraction bf1 =
        new BigFraction(f1);
    BigFraction bf2 =
        new BigFraction(f2);
    return bf1.multiply(bf2);
};

Future<BigFraction> future =
    commonPool().submit(call);
...

BigFraction result =
    future.get();
```

*Can pass values to a callable via effectively final variables*

See javarevisited.blogspot.com/2015/03/what-is-effectively-final-variable-of.html
Programming with Java Futures

- Example of using Java Future via a Callable & the common fork-join pool

String f1 = "62675744/15668936";
String f2 = "609136/913704";

Callable<BigFraction> call = () -> {
    BigFraction bf1 =
        new BigFraction(f1);
    BigFraction bf2 =
        new BigFraction(f2);
    return bf1.multiply(bf2);
};

Future<BigFraction> future =
    commonPool().submit(call);
...

BigFraction result =
    future.get();

Submit a two-way task to run in a thread pool (in this case the common fork-join pool..)

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/ForkJoinPool.html
Example of using Java Future via a Callable & the common fork-join pool

String f1 = "62675744/15668936";
String f2 = "609136/913704";

Callable<BigFraction> call = () -> {
    BigFraction bf1 =
        new BigFraction(f1);
    BigFraction bf2 =
        new BigFraction(f2);
    return bf1.multiply(bf2);
};

Future<BigFraction> future =
    commonPool().submit(call);
...

BigFraction result =
    future.get();

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/Future.html
Example of using Java Future via a Callable & the common fork-join pool

String f1 = "62675744/15668936";
String f2 = "609136/913704";

Callable<? extends BigFraction> call = () -> {
    BigFraction bf1 =
        new BigFraction(f1);
    BigFraction bf2 =
        new BigFraction(f2);
    return bf1.multiply(bf2);
};

Future<? extends BigFraction> future =
    commonPool().submit(call);
...

BigFraction result =
    future.get();
Example of using Java Future via a Callable & the common fork-join pool

```java
String f1 = "62675744/15668936";
String f2 = "609136/913704";

Callable<BigFraction> call = () -> {
    BigFraction bf1 =
        new BigFraction(f1);
    BigFraction bf2 =
        new BigFraction(f2);
    return bf1.multiply(bf2);
};

Future<BigFraction> future =
    commonPool().submit(call);
...

BigFraction result =
    future.get();
```

See [docs.oracle.com/javase/8/docs/api/java/util/concurrent/Future.html#get](https://docs.oracle.com/javase/8/docs/api/java/util/concurrent/Future.html#get)
Example of using Java Future via a Callable & the common fork-join pool

```java
String f1 = "62675744/15668936";
String f2 = "609136/913704";

Callable<BigFraction> call = () -> {
    BigFraction bf1 = new BigFraction(f1);
    BigFraction bf2 = new BigFraction(f2);
    return bf1.multiply(bf2);
};

Future<BigFraction> future = commonPool().submit(call);
...
BigFraction result = future.get(n, SECONDS);
```

See [docs.oracle.com/javase/8/docs/api/java/util/concurrent/Future.html#get](https://docs.oracle.com/javase/8/docs/api/java/util/concurrent/Future.html#get)
End of Programming with Java Futures