Overview of Java
Barrier Synchronizers

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

Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA
Learning Objectives in this Lesson

• Understand what barrier synchronization is & know three different ways of using barrier synchronizers
• Note a human known use of barrier synchronization
• Recognize the three types of Java barrier synchronizers
Learning Objectives in this Lesson

• Understand what barrier synchronization is & know three different ways of using barrier synchronizers
• Note a human known use of barrier synchronization
• Recognize the three types of Java barrier synchronizers
• Know how to categorize various type of Java barrier synchronizers

<table>
<thead>
<tr>
<th># of Iterations</th>
<th>Fixed # of Parties</th>
<th>Variable # of Parties</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-Shot</td>
<td>CountDown Latch</td>
<td>Phaser</td>
</tr>
<tr>
<td>Cyclic</td>
<td>CyclicBarrier</td>
<td>Phaser</td>
</tr>
</tbody>
</table>
Types of Java Barrier Synchronizers
Types of Java Barrier Synchronizers

- Java supports 3 types of barrier synchronizers
Types of Java Barrier Synchronizers

- Java supports 3 types of barrier synchronizers
  - **CountDownLatch**
    - Allows one or more threads to wait on the completion of operations in other threads

See [docs.oracle.com/javase/8/docs/api/java/util/concurrent/CountDownLatch.html](docs.oracle.com/javase/8/docs/api/java/util/concurrent/CountDownLatch.html)

![Race Horses at Starting Gate](image)

*e.g., a race can’t begin until all horses are at the starting gate*
Types of Java Barrier Synchronizers

- Java supports 3 types of barrier synchronizers
  - **CountDownLatch**
    - Allows one or more threads to wait on the completion of operations in other threads
    - Supports entry & exit barriers, but not cyclic barriers

```java
<<Java Class>>

CountDownLatch

- CountDownLatch(int)
- await():void
- await(long, TimeUnit):boolean
- countDown():void
- getCount():long
- toString()
```
Types of Java Barrier Synchronizers

- Java supports 3 types of barrier synchronizers
  - **CountDownLatch**
    - Allows one or more threads to wait on the completion of operations in other threads
    - Supports entry & exit barriers, but not cyclic barriers
  - The CountDownLatch API is very simple

```
<<Java Class>>
CountDownLatch

- CountDownLatch(int)
- await():void
- await(long, TimeUnit):boolean
- countDown():void
- getCount():long
- toString()
```
Types of Java Barrier Synchronizers

- Java supports 3 types of barrier synchronizers
  - `CountDownLatch`
  - `CyclicBarrier`

- **CyclicBarrier**
  - Allows a set of threads to all wait for each other to reach a common barrier point

See [docs.oracle.com/javase/8/docs/api/java/util/concurrent/CyclicBarrier.html](docs.oracle.com/javase/8/docs/api/java/util/concurrent/CyclicBarrier.html)

e.g., a team begins their work when the next car arrives on the assembly line
Types of Java Barrier Synchronizers

- Java supports 3 types of barrier synchronizers
  - CountDownLatch
  - CyclicBarrier
    - Allows a set of threads to all wait for each other to reach a common barrier point
    - Supports entry, exit, & cyclic barriers for a fixed # of threads

![Java Class]

- CyclicBarrier
  - Constructor: CyclicBarrier(int, Runnable)
  - Constructor: CyclicBarrier(int)
  - Method: getParties(): int
  - Method: await(): int
  - Method: await(long, TimeUnit): int
  - Method: isBroken(): boolean
  - Method: reset(): void
  - Method: getNumberOfWaiting(): int
Types of Java Barrier Synchronizers

- Java supports 3 types of barrier synchronizers
  - **CountDownLatch**
  - **CyclicBarrier**
    - Allows a set of threads to all wait for each other to reach a common barrier point
    - Supports entry, exit, & cyclic barriers for a fixed # of threads
  - The CyclicBarrier API is also very simple
Types of Java Barrier Synchronizers

- Java supports 3 types of barrier synchronizers
  - CountDownLatch
  - CyclicBarrier
  - Phaser
- A more flexible, reusable, & dynamic barrier synchronizer that subsumes CyclicBarrier & CountDownLatch

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/Phaser.html
e.g., crews begin their work when all the team members arrive
Types of Java Barrier Synchronizers

- Java supports 3 types of barrier synchronizers
  - CountDownLatch
  - CyclicBarrier
  - Phaser
    - A more flexible, reusable, & dynamic barrier synchronizer that subsumes CyclicBarrier & CountDownLatch
    - Supports entry, exit, & cyclic barriers for a variable # of threads
Types of Java Barrier Synchronizers

- Java supports 3 types of barrier synchronizers
  - **CountDownLatch**
  - **CyclicBarrier**
  - **Phaser**
    - A more flexible, reusable, & dynamic barrier synchronizer that subsumes CyclicBarrier & CountDownLatch
    - Supports entry, exit, & cyclic barriers for a variable # of threads
    - The Phaser API is more complex..
Categorizing Java Barrier Synchronizers
Java’s barrier synchronizers can be categorized in several ways.

<table>
<thead>
<tr>
<th># of Iterations</th>
<th>Fixed # of Parties</th>
<th>Variable # of Parties</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-Shot</td>
<td>CountDown Latch</td>
<td>Phaser</td>
</tr>
<tr>
<td>Cyclic</td>
<td>CyclicBarrier</td>
<td>Phaser</td>
</tr>
</tbody>
</table>
Java’s barrier synchronizers can be categorized in several ways.

<table>
<thead>
<tr>
<th># of Iterations</th>
<th>Fixed # of Parties</th>
<th>Variable # of Parties</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-Shot</td>
<td>CountDown Latch</td>
<td>Phaser</td>
</tr>
<tr>
<td>Cyclic</td>
<td>CyclicBarrier</td>
<td>Phaser</td>
</tr>
</tbody>
</table>
Java’s barrier synchronizers can be categorized in several ways.

<table>
<thead>
<tr>
<th># of Iterations</th>
<th>Fixed # of Parties</th>
<th>Variable # of Parties</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-Shot</td>
<td>CountDownLatch</td>
<td>Phaser</td>
</tr>
<tr>
<td>Cyclic</td>
<td>CyclicBarrier</td>
<td>Phaser</td>
</tr>
</tbody>
</table>
Java’s barrier synchronizers can be categorized in several ways:

<table>
<thead>
<tr>
<th># of Iterations</th>
<th>Fixed # of Parties</th>
<th>Variable # of Parties</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-Shot</td>
<td>CountDownLatch</td>
<td>Phaser</td>
</tr>
<tr>
<td>Cyclic</td>
<td>CyclicBarrier</td>
<td>Phaser</td>
</tr>
</tbody>
</table>

A CountDownLatch can be used with a variable number of parties, but it’s uncommon.
Java’s barrier synchronizers can be categorized in several ways:

<table>
<thead>
<tr>
<th># of Iterations</th>
<th>Fixed # of Parties</th>
<th>Variable # of Parties</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-Shot</td>
<td>CountDown Latch</td>
<td>Phaser</td>
</tr>
<tr>
<td>Cyclic</td>
<td>CyclicBarrier</td>
<td>Phaser</td>
</tr>
</tbody>
</table>

These categories are not mutually exclusive, i.e., Phaser appears multiple times.
End of Overview of Java Barrier Synchronizers