Computational Expression
Random Class, Math Class, Wrapper Classes

Janyl Jumadinova

18 February, 2019
Random Class

- The Random class is part of the java.util package
- It provides methods that generate pseudorandom numbers
Random Class

- The Random class is part of the java.util package
- It provides methods that generate pseudorandom numbers
- Before you can use its methods, you must create an instance of the Random class
- `Random rand = new Random();`
Random Class

float nextFloat( )
float nextDouble( )
- Returns a random number between 0.0 inclusive and 1.0 exclusive.
Random Class

float nextFloat( )
float nextDouble( )
- Returns a random number between 0.0 inclusive and 1.0 exclusive.
Random rand = new Random( );
float f;
f = rand.nextFloat( );
Random Class

`int nextInt()` - Returns a random number that ranges over all possible int values positive and negative.
Random Class

int nextInt( ) - Returns a random number that ranges over all possible int values positive and negative.
Random rand = new Random( );
int num;
num = rand.nextInt( );
Random Class

```
int nextInt() - Returns a random number that ranges over all possible int values positive and negative.
Random rand = new Random();
int num;
num = rand.nextInt();
```

```
int nextInt(int num) - Returns a random number between 0 (inclusive) and num (exclusive).
```
Random Class

```java
int nextInt( ) - Returns a random number that ranges over all possible int values positive and negative.
Random rand = new Random( );
int num;
num = rand.nextInt( );
```

```java
int nextInt( int num ) - Returns a random number between 0 (inclusive) and num (exclusive).
Random rand = new Random( );
int num;
num = rand.nextInt(5 );
```
Math Class

- Math plays a large role in computer programs.
- Because of this, there is an entire class (Math) that provides easy-to-use interfaces to many common mathematical methods.
- Unlike most other classes, the Math class is part of the java.lang package, which is imported automatically by the compiler when you compile a program. Therefore, you don’t need to do anything special in your program to have access to these methods.
Math Class

Math class consists of:

- **static methods**, which are methods that don’t depend on the contents of an object.

- **static fields**, which are values that are usually defined to be public, final and static, meaning that anyone can access them outside the package. Since their values are final, that means that they are constant and can’t be changed.
## Math Class Examples

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Math.abs(x)</code></td>
<td>Absolute value of <code>x</code></td>
<td><code>Math.abs(23.7)</code> is 23.7</td>
</tr>
<tr>
<td><code>Math.ceil(x)</code></td>
<td>Rounds <code>x</code> to smallest integer not less than <code>x</code></td>
<td><code>Math.ceil(9.2)</code> is 10.0</td>
</tr>
<tr>
<td><code>Math.exp(x)</code></td>
<td>Exponential method $e^x$</td>
<td><code>Math.exp(1.0)</code> is 2.71828</td>
</tr>
<tr>
<td><code>Math.floor(x)</code></td>
<td>Rounds <code>x</code> to the largest integer not greater than <code>x</code></td>
<td><code>Math.floor(9.2)</code> is 9.0</td>
</tr>
<tr>
<td><code>Math.max(x, y)</code></td>
<td>Larger value of <code>x</code> and <code>y</code></td>
<td><code>Math.max(2.3, 12.7)</code> is 12.7</td>
</tr>
<tr>
<td><code>Math.min(x, y)</code></td>
<td>Smaller value of <code>x</code> and <code>y</code></td>
<td><code>Math.min(2.3, 12.7)</code> is 2.3</td>
</tr>
<tr>
<td><code>Math.pow(x, y)</code></td>
<td><code>x</code> raised to the power <code>y</code>, $x^y$</td>
<td><code>Math.pow(2.0, 7.0)</code> is 128.0</td>
</tr>
<tr>
<td><code>Math.sqrt(x)</code></td>
<td>Square root of <code>x</code></td>
<td><code>Math.sqrt(900.0)</code> is 30.0</td>
</tr>
<tr>
<td><code>Math.PI</code></td>
<td>The mathematical value Pi</td>
<td>3.14159265358979323846</td>
</tr>
<tr>
<td><code>Math.E</code></td>
<td>Base value for natural logs</td>
<td>2.7182818284590452354</td>
</tr>
</tbody>
</table>
May want to have an object hold a simple primitive value.
Wrapper Classes

- May want to have an object hold a simple primitive value.
- A wrapper class represents a particular primitive type.
May want to have an object hold a simple primitive value.

A wrapper class represents a particular primitive type.

For example, `Integer` represents a simple integer value.

An object created from the `Integer` class stores a single `int` value.