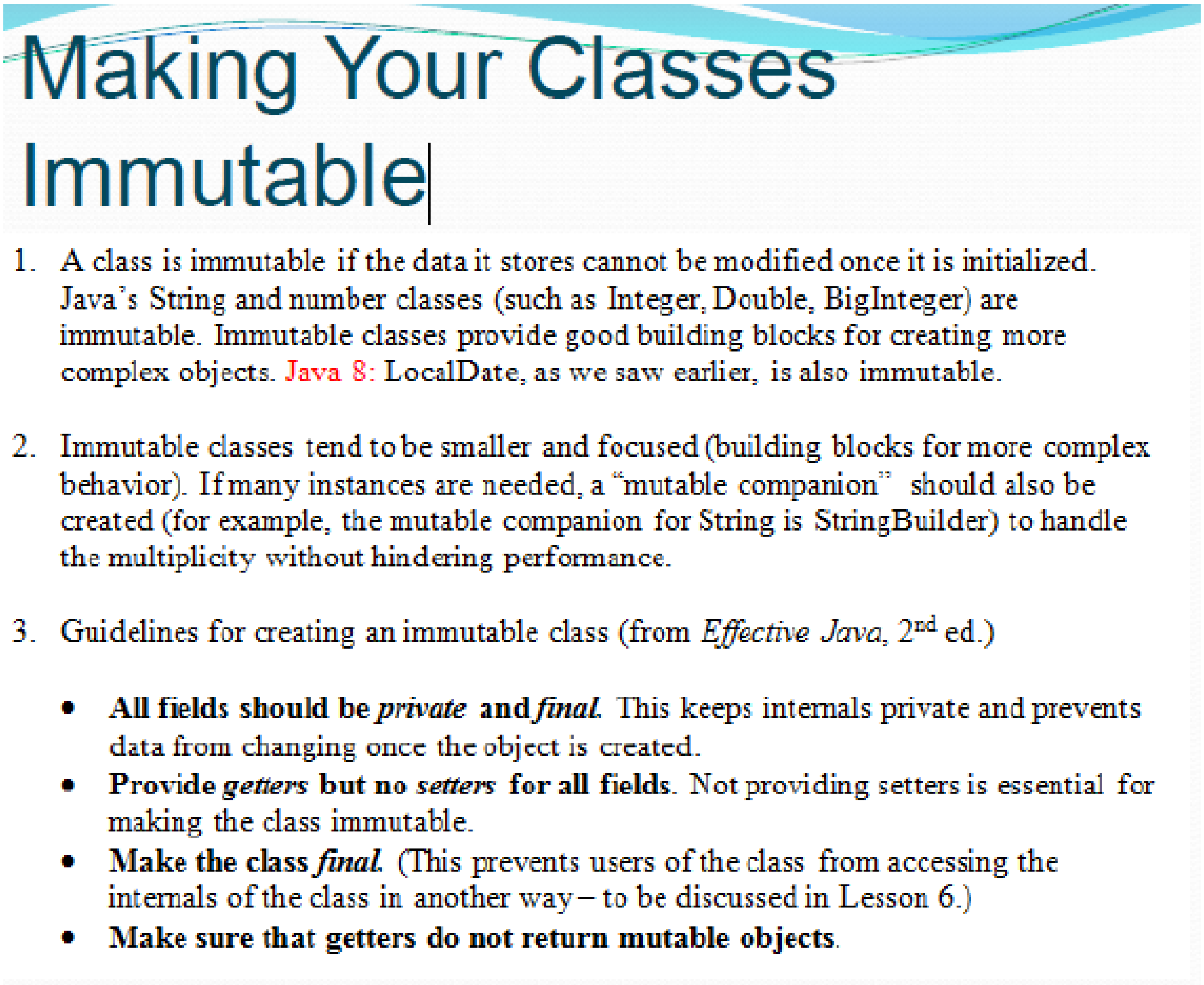

Making Your Classes Immutable

1. A class is immutable if the data it stores cannot be modified once it is initialized. Java's String and number classes (such as Integer, Double, BigInteger) are immutable. Immutable classes provide good building blocks for creating more complex objects. Java 8: LocalDate, as we saw earlier, is also immutable.

2. Immutable classes tend to be smaller and focused (building blocks for more complex behavior). If many instances are needed, a "mutable companion" should also be created (for example, the mutable companion for String is StringBuilder) to handle the multiplicity without hindering performance.

3. Guidelines for creating an immutable class (from *Effective Java*, 2nd ed.)

   - **All fields should be *private* and *final*.** This keeps internals private and prevents data from changing once the object is created.
   - **Provide *getters* but no *setters* for all fields.** Not providing setters is essential for making the class immutable.
   - **Make the class *final*.** (This prevents users of the class from accessing the internals of the class in another way – to be discussed in Lesson 6.)
   - **Make sure that getters do not return mutable objects**.