java14 新特性，代码简化神器

作者：Eli
原文链接：https://ld246.com/article/1596957739057
来源网站：链滴
许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

  Java14在今年3月份被发布，其中包含了许多的新特性，关于具体有哪些特，可以查阅官方文档。<div text-align:center>

JDK 14

JDK 14 is the open-source reference implementation of version 14 of the Java SE Platform as specified by by JSR 389 in the Java Community Process.

JDK 14 reached General Availability on 17 March 2020. Production-ready binaries under the GPL are available from Oracle; binaries from other vendors will follow shortly.

The features and schedule of this release were proposed and tracked via the JEP Process, as amended by the JEP 2.0 proposal. The release was produced using the JDK Release Process (JEP 3).

Features

305: Pattern Matching for instanceof (Preview)
343: Packaging Tool (Incubator)
345: NUMA-Aware Memory Allocation for G1
349: JFR Event Streaming
352: Non-Volatile Mapped Byte Buffers
358: Helpful NullPointerExceptions
359: Records (Preview)
361: Switch Expressions (Standard)
362: Deprecate the Solaris and SPARC Ports
363: Remove the Concurrent Mark Sweep (CMS) Garbage Collector
364: ZGC on macOS
365: ZGC on Windows
366: Deprecate the ParallelScavenge + SerialOld GC Combination
367: Remove the Pack200 Tools and API
368: Text Blocks (Second Preview)
370: Foreign-Memory Access API (Incubator)

Schedule

2019/12/12	Rampdown Phase One (fork from main line)
2020/01/16	Rampdown Phase Two
2020/02/06	Initial Release Candidate
2020/02/20	Final Release Candidate
2020/03/17	General Availability

今天跟大家分享一个一个新的特性：

JEP 359: Records (Preview)

Java14祭出的大器！

背景介绍

  我们经常听到不少开发者对Java抱怨，“Java代码太冗长啰嗦了”，确实Jav
经常要写很多低级的比较Low的代码；比如：constructors, getters, equals(), hashCode(), toString(
方法等，是不是感同身受?

  如果开发人员为了省事而偷懒省略了其中的一些方法，甚至会发生一些问题。

基本介绍

  Records 是一种新的类申明形式，和枚举类型一样，它是一种受限制的类（class）。简单的说Records就是一种新语法糖，最终的目的还是为了简化代码，相当于Lombok的 @Data注解，但是Java14中的Records并不能完全将其替代掉。不过在未来可能会发生变更，因为现在的Records毕竟还是Preview（预览）版本。

关于Lombock的介绍和使用方式因为我本人没有写过相关的文章，大家可以直接查阅相关资料跟项官网以及该项目的GitHub。

Records 的语法是：

```
record Point(int x, int y) { }
```

用 record 修饰一个类，括号里面直接带参数，{ } 里面是可选的。说了这么多到底有啥用呢？

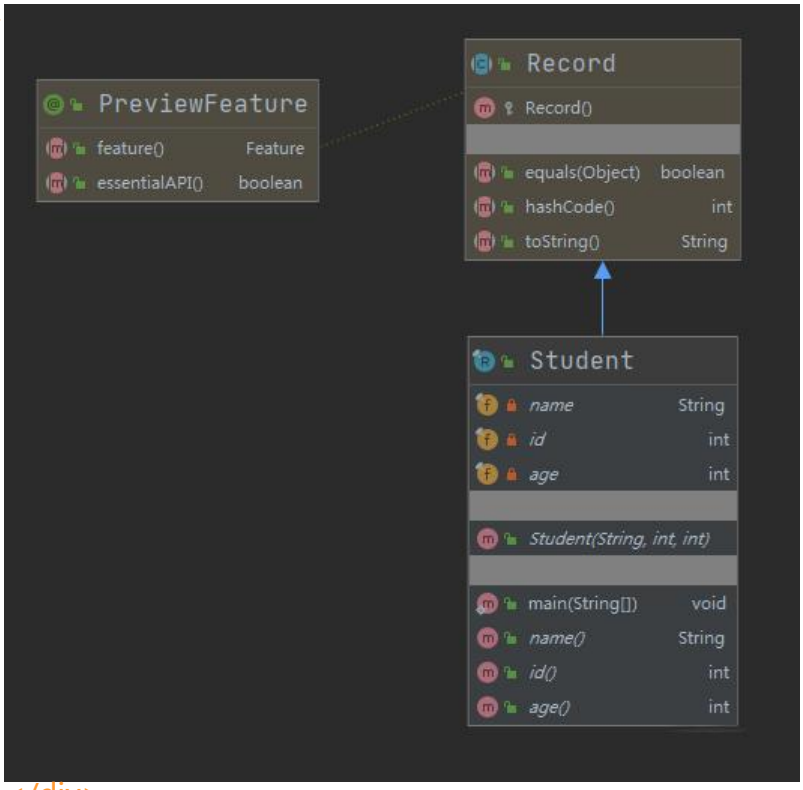
最好的方式就是进行实践，下面通过示例代码一起了解一下。

示例分析

  关于环境变量的配置我就不在这里说了，能来看示例代码说明环境变量的配应该难不倒你。下面是一个示例：

```
public record Student(String name, int id, int age) {}
```

没错，一行搞定（public 都可以省略），就是这么简单粗暴！！！再来看一下它类的结构继承图：<div text-align:center>



我们再通过 IDEA 反编译 class 类的方式来看下它到底做了什么：<div text-align:center>

```
public final class Student extends java.lang.Record {
    private final java.lang.String name;
    private final int id;
    private final int age;

    public Student(java.lang.String name, int id, int age) { /* compiled code */ }

    public java.lang.String toString() { /* compiled code */ }

    public final int hashCode() { /* compiled code */ }

    public final boolean equals(java.lang.Object o) { /* compiled code */ }

    public java.lang.String name() { /* compiled code */ }

    public int id() { /* compiled code */ }

    public int age() { /* compiled code */ }
}
```

看完之后是不是感觉有点像Lombok?

1. 生成的类是final类型的，并且继承了： java.lang.Record；
2. 生成的类成员变量全是 private final 类型的；
3. 自动生成了类构造器、toString()、hashCode()、equals()，以及类似 getter 的变量访问方法；

由于工具编译器的问题，上边看到的部分源代码是 /* compiled code */，我们再在 Student 类里面入 main 方法测试下：

```
public record Student(String name, int id, int age) {

    public static void main(String[] args) {
        Student student1 = new Student("张三", 1001, 18);
        System.out.println(student1.name());
        System.out.println(student1.id());
        System.out.println(student1.age());

        System.out.println(student1);

        Student student2 = new Student("张三", 1001, 18);
        Student student3 = new Student("张三", 1003, 18);
        System.out.println(student1.equals(student2));
        System.out.println(student1.equals(student3));
    }

}
```

输出结果：

```
张三
```

```
1001
18
Student[name=张三, id=1001, age=18]
true
false
```

从结果可以得知 toString/ equals 等生成的方法都按照特定的规则重写了，而不是使用内存地址。

是否可以添加成员变量?

  答案是不能，Records类里面并不能手动添加成员变量。比如说如下代码添了一个地址成员变量就编译报错了。<div text-align:center>

能否替代现有的Lombok?

答案是否定的，并不能完全替代!

从上面的结论我们可以得知 Records 类有以下限制：

1. record 类是 final 修饰的，所以不能被其他子类继承；

2. 因为 Java 类是单继承，而自身又已经继承了 Record 类，所以不能再继承其他类，但是可以实现口；

3. 成员变量也是 final 类型的，所以其值或者引用不能被更改，如果是引用类型，可以修改对象里面值。

  由于它以上这些限制，想完全代替 Lombok 是不可能的，当然，不用纠结些限制的话，某些场合是可以代替 Lombok 使用的。并且目前还处于预览版，很有可能进行改动。

使用注意事项

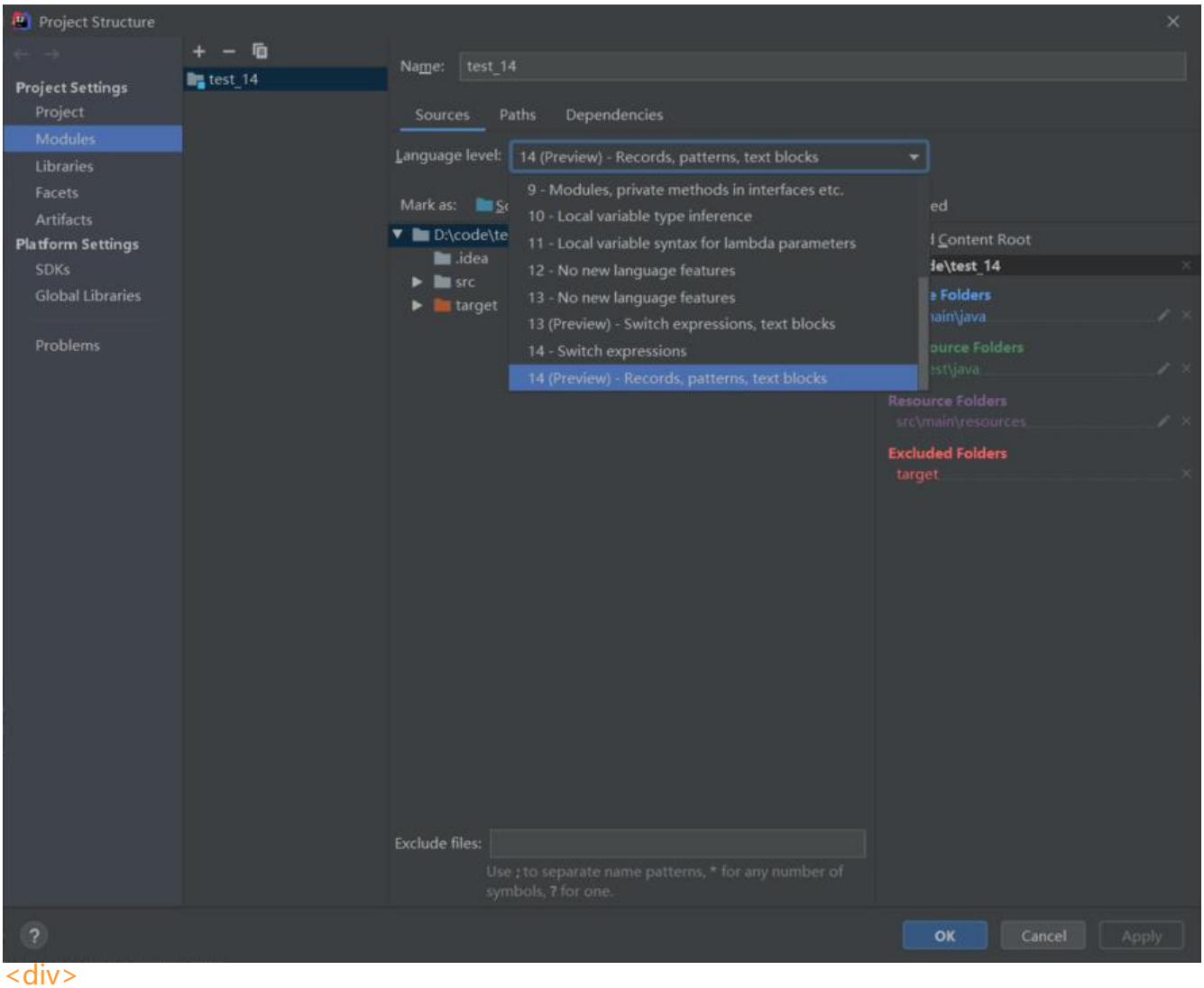
  如果想要安装Java14用于学习还是可以的，实际的生产环境可能不会太尽人，毕竟现在大多数的人还在守着Java8的环境，所以使用Java14出现问题能参考的文档较少，或者是现诡异的BUG。目前Java14的Record还是属于预览功能因此要使用Java14还需要通过以下方式来启它。

1. 我们需要使用 --enable-preview选项来手动启用它。

```
javac --enable-preview --release 14 XXX.java

java --enable-preview
```

2. 对于IntelliJ IDE，请更新到最新版本2020.1.1；否则，请重新安装。 它应该支持Java 14的新预览能。<div text-align:center>



总结

  Java 14 Records 是一个新的语法糖，是一种 "数据载体"，可以告别传统的
效的生成代码模板，现在还是预览特性,后续如果出现新的改动再跟大家一起探讨。

  虽然现在存在着诸多的限制，至少 Java 正在大步往前走，变得越来越智能
越来越简化了。可能有人会说，没卵用Lombok足以！现在来看确实是这样的，但是有朝一日Java总
干掉Lombok，因为Records有十足的优势，它是Java自带的语法，不需要装插件，不需要jar包；Lo
bok 是团队工具，不一定都会用，你要知道，有些公司是禁止使用 Lombok 插件的。

  最后本文只作为了解，学习即可，不能用于正式的生产力，出去吹牛逼还是
以的。

原文链接：java14 新特性，代码简化神器