**Java 8**

The main changes of the Java 8 release were these:

* [Lambda Expression and Stream API](https://reflectoring.io/java-release-notes/#lambda-expressions-and-stream-api)
* [Method Reference](https://reflectoring.io/java-release-notes/#method-reference)
* [Default Methods](https://reflectoring.io/java-release-notes/#default-methods)
* [Type Annotations](https://reflectoring.io/java-release-notes/#type-annotations)
* [Repeating Annotations](https://reflectoring.io/java-release-notes/#repeating-annotations)
* [Method Parameter Reflection](https://reflectoring.io/java-release-notes/#method-parameter-reflection)

**Java 9**

Java 9 introduced these main features:

* [Java Module System](https://reflectoring.io/java-release-notes/#java-module-system)
* [Try-with-resources](https://reflectoring.io/java-release-notes/#try-with-resources)
* [Diamond Syntax with Inner Anonymous Classes](https://reflectoring.io/java-release-notes/#diamond-syntax-with-inner-anonymous-classes)
* [Private Interface Methods](https://reflectoring.io/java-release-notes/#private-interface-methods)

## Java 10

### Local Variable Type Inference

#### Implicit Typing with var

## Java 11

### Local Variable Type in Lambda Expressions



## Java 14

### Switch Expressions

Switch expressions allowed us to omit break calls inside every case block.

Using Switch Expressions

days = switch (month) {

 case JANUARY, MARCH, MAY, JULY, AUGUST, OCTOBER, DECEMBER -> 31;

 case FEBRUARY -> 28;

 case APRIL, JUNE, SEPTEMBER, NOVEMBER -> 30;

 default -> throw new IllegalStateException();

 };

The yield Keyword

days = switch (month) {

 case JANUARY, MARCH, MAY, JULY, AUGUST, OCTOBER, DECEMBER -> {

 System.out.println(month);

 yield 31;

 }

## Java 15

### Text Blocks “””



## Java 16

### Pattern Matching of instanceof





defining variable Car c and Bicycle b

public class PatternMatching {

 public static double price(Vehicle v) {

 if (v instanceof Car c) {

 return 10000 - c.kilomenters \* 0.01 -

 (Calendar.getInstance().get(Calendar.YEAR) -

 c.year) \* 100;

 } else if (v instanceof Bicycle b) {

 return 1000 + b.wheelSize \* 10;

 } else throw new IllegalArgumentException();

 }

}

### Records

public record VehicleRecord(String code, String engineType) {}

we cannot extend a record class

## Java 17

### Sealed Classes

public sealed class Vehicle permits Bicycle, Car {...}

The final modifier on a class doesn’t allow anyone to extend it

What about when we want to extend a class but only allow it for some classes?

public sealed class Vehicle permits Bicycle, Car {...}