

Contents

Preface	2
Var variable type of java.	3
Usage	3
Var is not backward compatible	3
Var does not impact performance	4
Unmodifiable Collections.....	4
List.copyList(Collection)	4
toUnmodifiable()	4
Optional new method orElseThrow()	5
Time-Based Release of Versions java	5
Reference	6

Preface

Java 10 was introduced 20/03/2018

Var variable is added

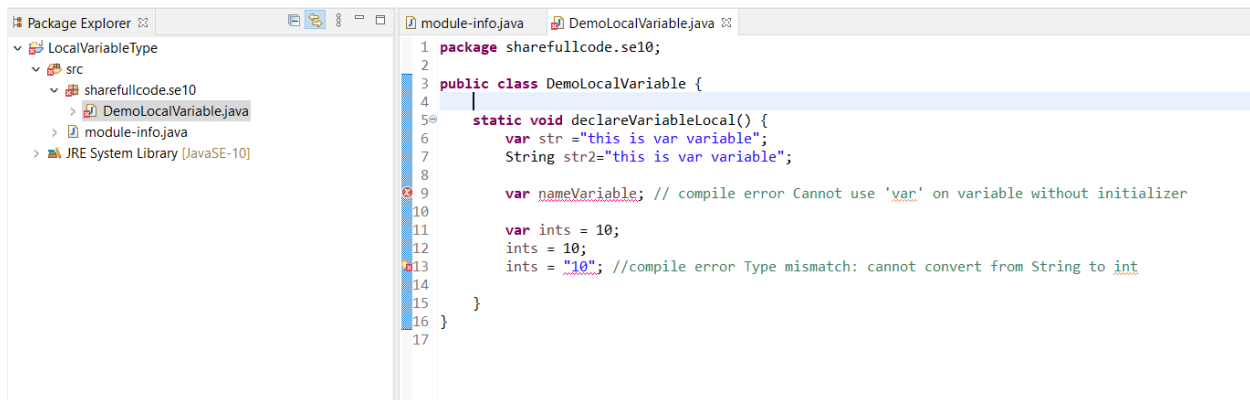
Unmodifiable Collections

Optional new method orElseThrow()

Time-Based Release of Versions java

Var variable type of java.

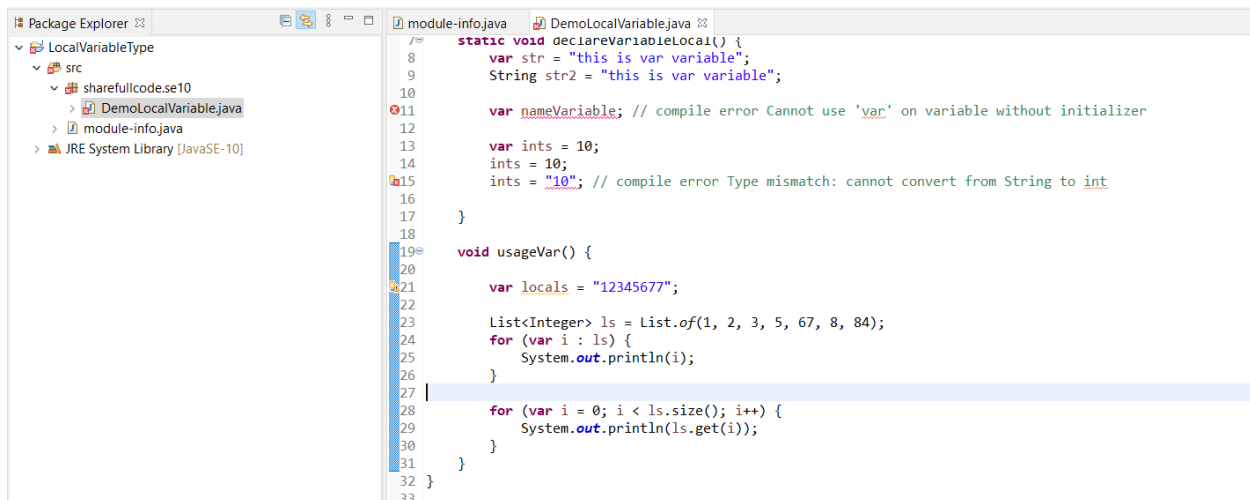
Java has been progressively working on reducing the verbosity from syntax. It was Diamond operator (<> Generic), and now it is var to declare variables in java. When you are using Var to declare basically, instead of declaring a variable type, it assumes its type from what it is being set.



```
1 package sharefullcode.se10;
2
3 public class DemoLocalVariable {
4
5     static void declareVariableLocal() {
6         var str = "this is var variable";
7         String str2 = "this is var variable";
8
9         var nameVariable; // compile error Cannot use 'var' on variable without initializer
10
11         var ints = 10;
12         ints = 10;
13         ints = "10"; //compile error Type mismatch: cannot convert from String to int
14
15     }
16 }
17
```

Usage

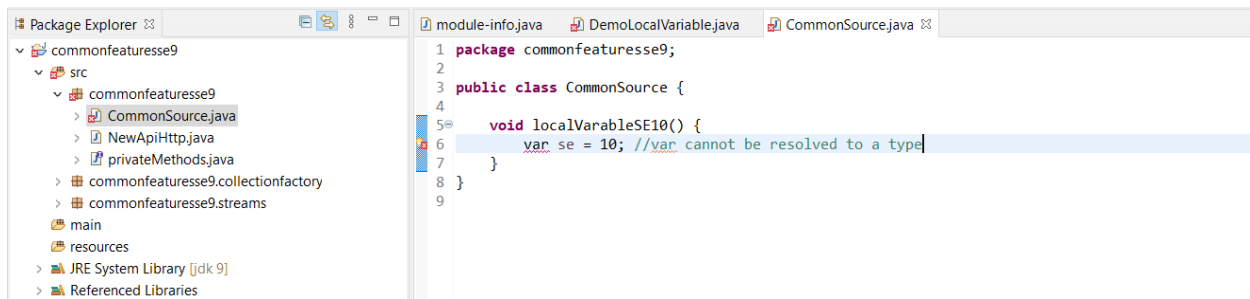
Using var for to local variables with initializers, indexes in the enhanced for loop. It would not be available for method formals, constructor formals, method return types, fields, catch formals, or any other kind of variable declaration.



```
1 //
2 static void declareVariableLocal() {
3     var str = "this is var variable";
4     String str2 = "this is var variable";
5
6     var nameVariable; // compile error Cannot use 'var' on variable without initializer
7
8     var ints = 10;
9     ints = 10;
10    ints = "10"; // compile error Type mismatch: cannot convert from String to int
11
12 }
13
14 void usageVar() {
15
16     var locals = "12345677";
17
18     List<Integer> ls = List.of(1, 2, 3, 5, 67, 8, 84);
19     for (var i : ls) {
20         System.out.println(i);
21     }
22
23     for (var i = 0; i < ls.size(); i++) {
24         System.out.println(ls.get(i));
25     }
26 }
27
28 }
29
30 }
31
32 }
33
```

Var is not backward compatible

As this is new language feature, code written using var will not be compiled in lower JDK versions 10. So use this feature only when you are sure about it.



Var does not impact performance

Remember, in Java the types are not inferred at Runtime, but at compile time. That means the resulting bytecode is the same as with explicit type declaration it does include the information about the type.

Unmodifiable Collections

List.copyOfList(Collection)

List, Map, Set each got a new static method copyOf(Collection). It returns the unmodifiable copy of the given Collection.

```
List<String> copyList = List.copyOf(CollectionList);
```

```
Copylist.addAll(list_other);
```

Throw exception UnsupportedOperationException.

toUnmodifiable()

it return a Collection not change

```
List<Integer> listUnmodifiable = Stream.of(2,3,5,76,2,2,45,3,2).collect ( Collectors.toUnmodifiable() );
```

```
listUnmodifiable.addAll(list_other);
```

Throw exception UnsupportedOperationException.

Optional new method orElseThrow()

Optional, OptionalDouble, OptionalInt, OptionalLong each got a new method orElseThrow which doesn't take any argument and throws NoSuchElementException

```
List.Stream().Filter (true).findFirst().orElseThrow()
```

It is the preferred alternative to the use get() method.

Time-Based Release of Versions java

Start with java 10, Oracle has moved to the time-based release of java. This has following implications:

A new java release every six months.

Long-term support release will be marked as LTS, support for three years

Java 11 will be an LTS release.

Reference

This section is non commercial mainly sharing and advance knowlage for java.This tutorials has referenced document from the list below if you has complain for license, i will remove all from internet. Thank you all everything

<https://howtodoinjava.com/java10/var-local-variable-type-inference/>

<https://www.baeldung.com/java-10-overview>