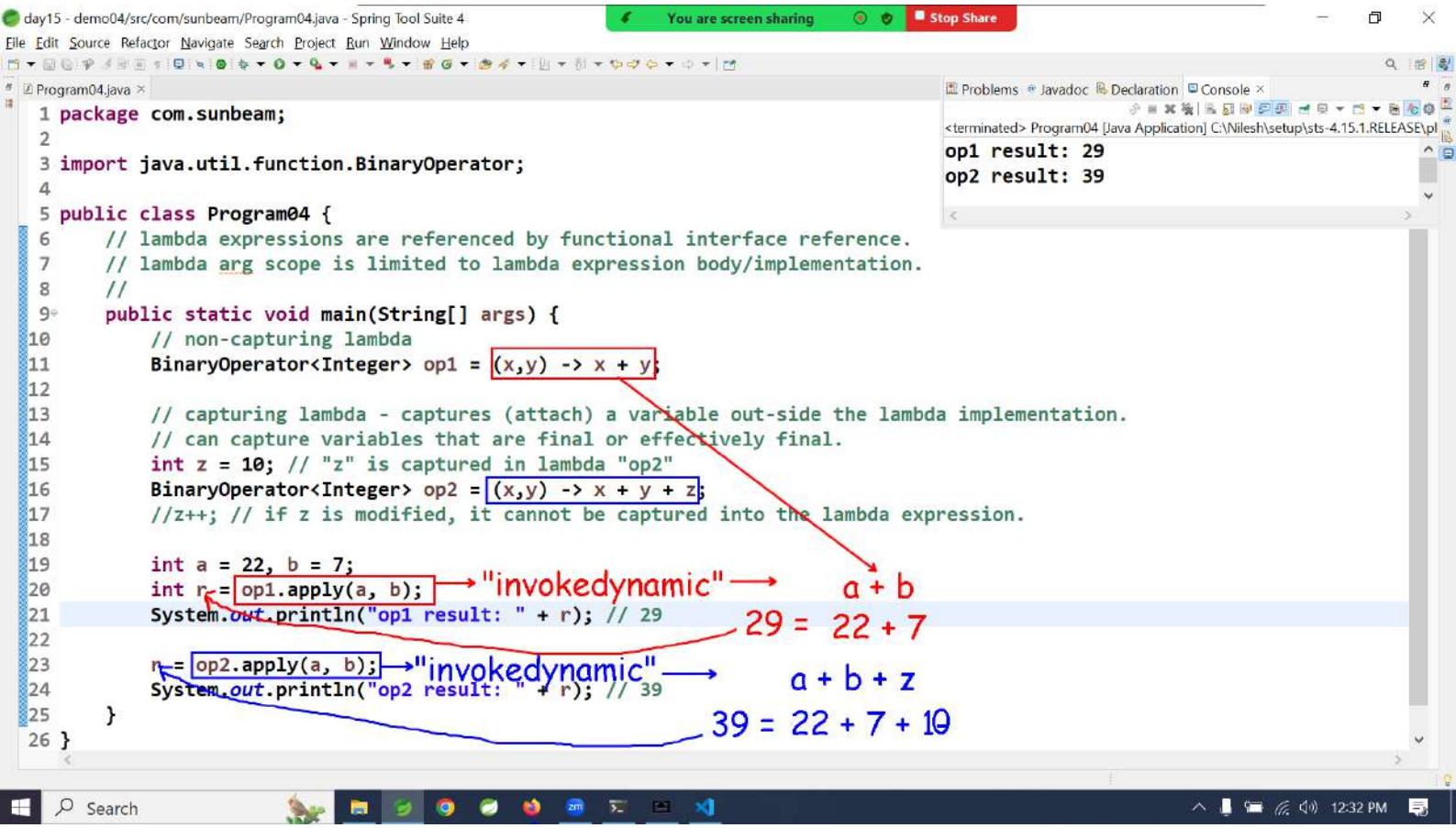
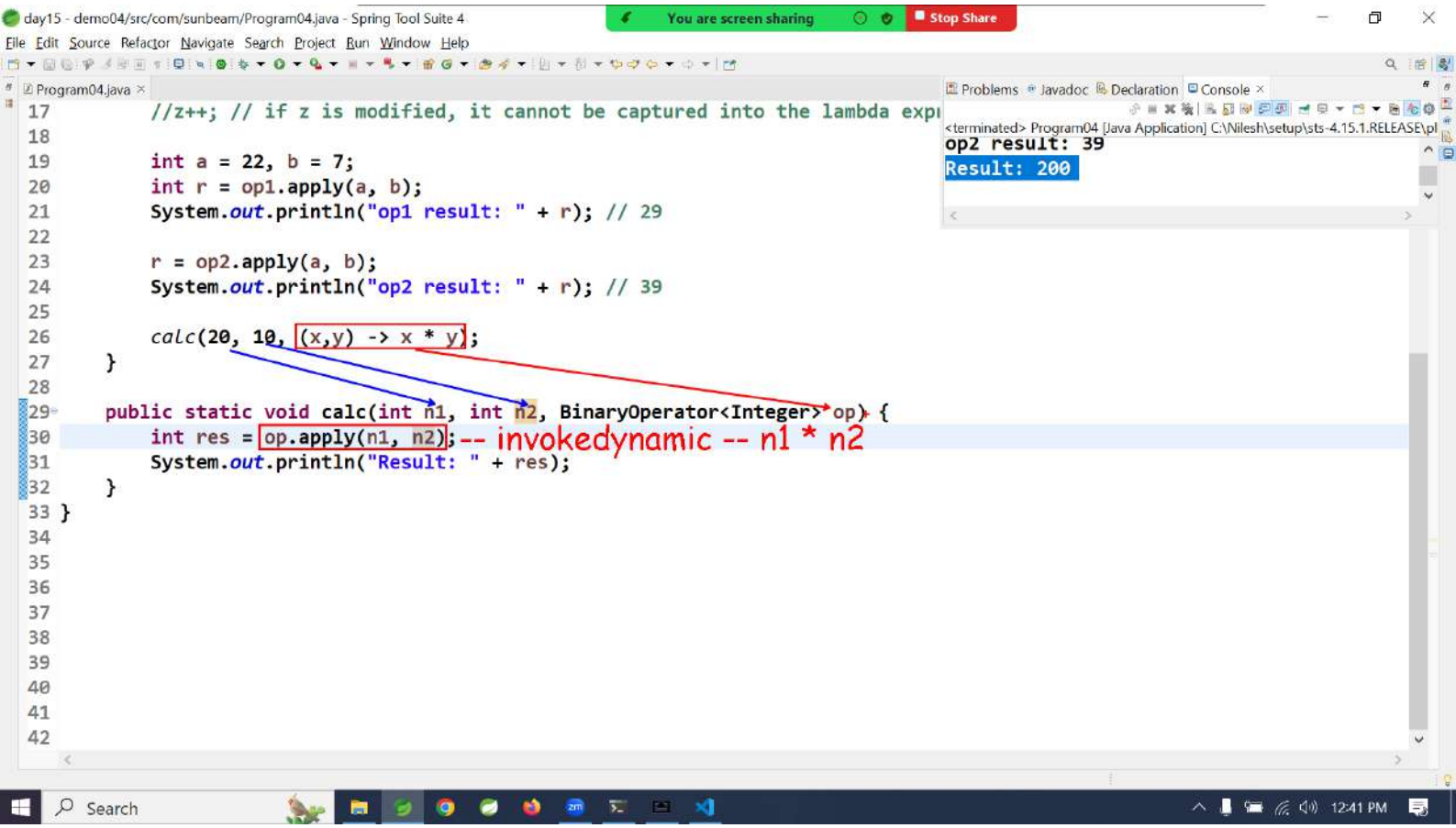
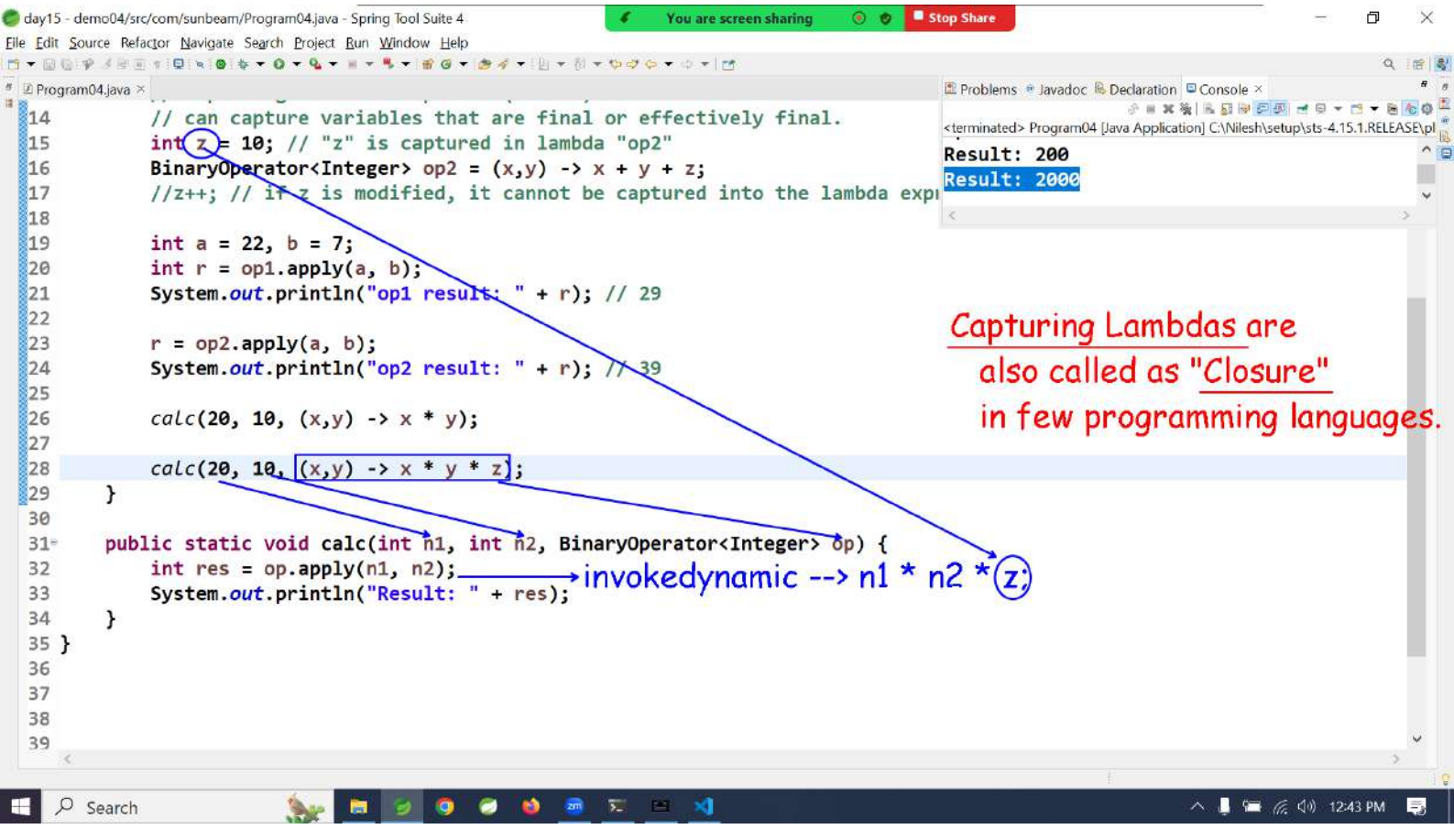
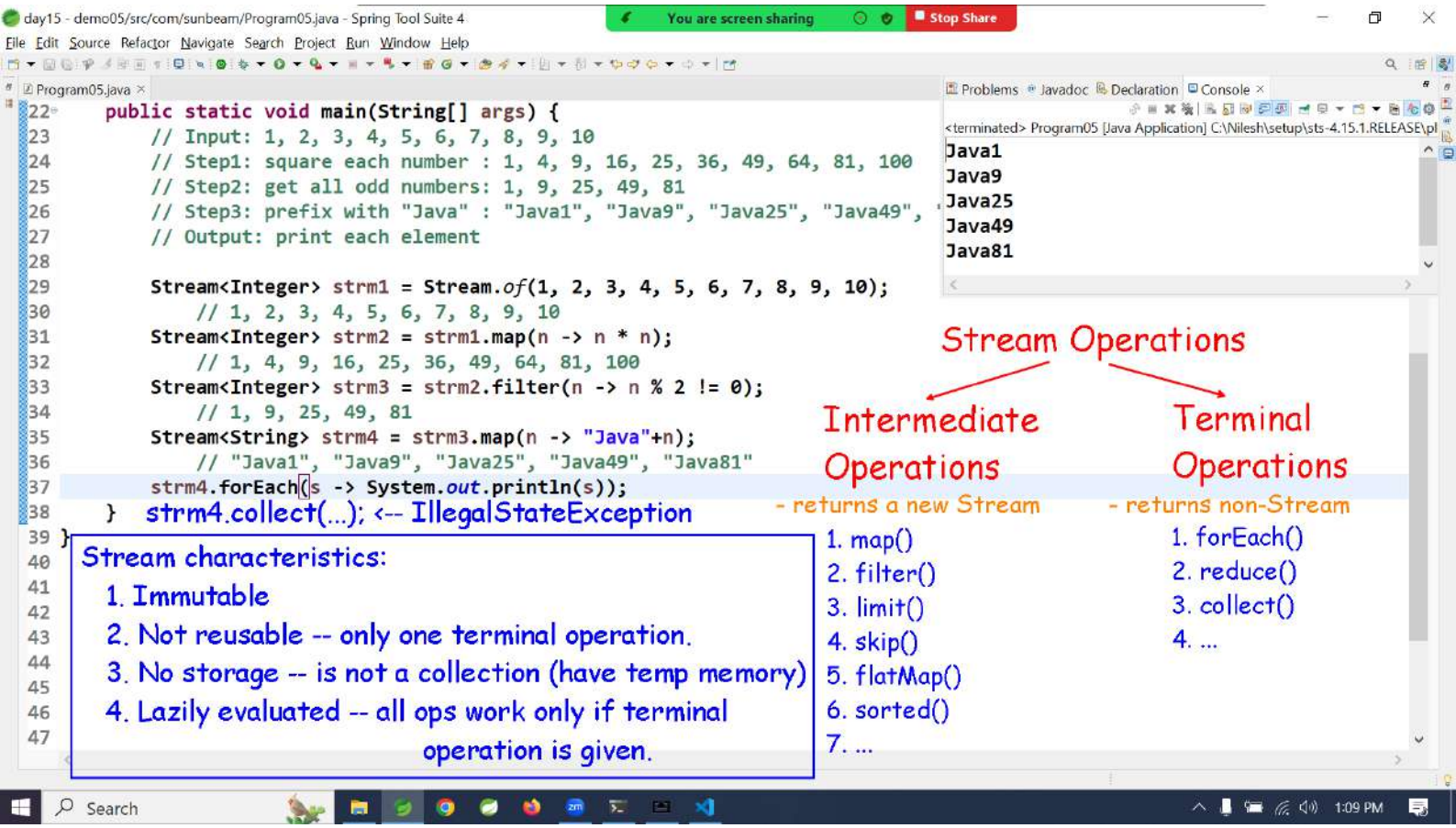

day15 - demo04/src/com/sunbeam/Program04.java - Spring Tool Suite 4
You are screen sharing
Stop Share
File Edit Source Refactor Navigate Search Project Run Window Help
Program04.java
Problems Javadoc Declaration Console
<terminated> Program04 [Java Application] C:\Nilesh\setup\sts-4.15.1.RELEASE\pl
op1 result: 29
op2 result: 39
1 package com.sunbeam;
2
3 import java.util.function.BinaryOperator;
4
5 public class Program04 {
6 // lambda expressions are referenced by functional interface reference.
7 // lambda arg scope is limited to lambda expression body/implementation.
8 //
9 public static void main(String[] args) {
10 // non-capturing lambda
11 BinaryOperator<Integer> op1 = (x,y) -> x + y;
12
13 // capturing lambda - captures (attach) a variable out-side the lambda implementation.
14 // can capture variables that are final or effectively final.
15 int z = 10; // "z" is captured in lambda "op2"
16 BinaryOperator<Integer> op2 = (x,y) -> x + y + z;
17 //z++; // if z is modified, it cannot be captured into the lambda expression.
18
19 int a = 22, b = 7;
20 int r = op1.apply(a, b);
21 System.out.println("op1 result: " + r); // 29
22
23 r = op2.apply(a, b);
24 System.out.println("op2 result: " + r); // 39
25 }
26 }
"invokedynamic"
a + b
29 = 22 + 7
"invokedynamic"
a + b + z
39 = 22 + 7 + 10
Search
12:32 PM


day15 - demo04/src/com/sunbeam/Program04.java - Spring Tool Suite 4
You are screen sharing
Stop Share
File Edit Source Refactor Navigate Search Project Run Window Help
Program04.java
//z++; // if z is modified, it cannot be captured into the lambda expr
int a = 22, b = 7;
int r = op1.apply(a, b);
System.out.println("op1 result: " + r); // 29
r = op2.apply(a, b);
System.out.println("op2 result: " + r); // 39
calc(20, 10, (x,y) -> x * y);
}
public static void calc(int n1, int n2, BinaryOperator<Integer> op) {
int res = op.apply(n1, n2);
-- invokedynamic -- n1 * n2
System.out.println("Result: " + res);
}
}
Problems Javadoc Declaration Console
<terminated> Program04 [Java Application] C:\Nilesh\setup\sts-4.15.1.RELEASE\pl
op2 result: 39
Result: 200
Search
12:41 PM


day15 - demo04/src/com/sunbeam/Program04.java - Spring Tool Suite 4
You are screen sharing
Stop Share
File Edit Source Refactor Navigate Search Project Run Window Help
Program04.java
// can capture variables that are final or effectively final.
int z = 10; // "z" is captured in lambda "op2"
BinaryOperator<Integer> op2 = (x,y) -> x + y + z;
//z++; // if z is modified, it cannot be captured into the lambda exp
int a = 22, b = 7;
int r = op1.apply(a, b);
System.out.println("op1 result: " + r); // 29
r = op2.apply(a, b);
System.out.println("op2 result: " + r); // 39
calc(20, 10, (x,y) -> x * y);
calc(20, 10, (x,y) -> x * y * z);
}
public static void calc(int n1, int n2, BinaryOperator<Integer> op) {
int res = op.apply(n1, n2);
System.out.println("Result: " + res);
}
}
Problems Javadoc Declaration Console
<terminated> Program04 [Java Application] C:\Nilesh\setup\sts-4.15.1.RELEASE\pl
Result: 200
Result: 2000
Capturing Lambdas are also called as "Closure" in few programming languages.
invokedynamic --> n1 * n2 * z;
Search
12:43 PM


day15 - demo05/src/com/sunbeam/Program05.java - Spring Tool Suite 4
You are screen sharing
Stop Share
File Edit Source Refactor Navigate Search Project Run Window Help
Program05.java
Problems Javadoc Declaration Console
<terminated> Program05 [Java Application] C:\Nilesh\setup\sts-4.15.1.RELEASE\pl
Java1
Java9
Java25
Java49
Java81
22 public static void main(String[] args) {
23 // Input: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
24 // Step1: square each number : 1, 4, 9, 16, 25, 36, 49, 64, 81, 100
25 // Step2: get all odd numbers: 1, 9, 25, 49, 81
26 // Step3: prefix with "Java" : "Java1", "Java9", "Java25", "Java49",
27 // Output: print each element
28
29 Stream<Integer> strm1 = Stream.of(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
30 // 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
31 Stream<Integer> strm2 = strm1.map(n -> n * n);
32 // 1, 4, 9, 16, 25, 36, 49, 64, 81, 100
33 Stream<Integer> strm3 = strm2.filter(n -> n % 2 != 0);
34 // 1, 9, 25, 49, 81
35 Stream<String> strm4 = strm3.map(n -> "Java"+n);
36 // "Java1", "Java9", "Java25", "Java49", "Java81"
37 strm4.forEach(s -> System.out.println(s));
38 } strm4.collect(...); <-- IllegalStateException
39 }
40
41
42
43
44
45
46
47
Stream Operations
Intermediate Operations
- returns a new Stream
1. map()
2. filter()
3. limit()
4. skip()
5. flatMap()
6. sorted()
7. ...
Terminal Operations
- returns non-Stream
1. forEach()
2. reduce()
3. collect()
4. ...
Stream characteristics:
1. Immutable
2. Not reusable -- only one terminal operation.
3. No storage -- is not a collection (have temp memory)
4. Lazily evaluated -- all ops work only if terminal operation is given.
Search
1:09 PM

day15 - demo05/src/com/sunbeam/Program05.java - Spring Tool Suite 4

You are screen sharing | Stop Share

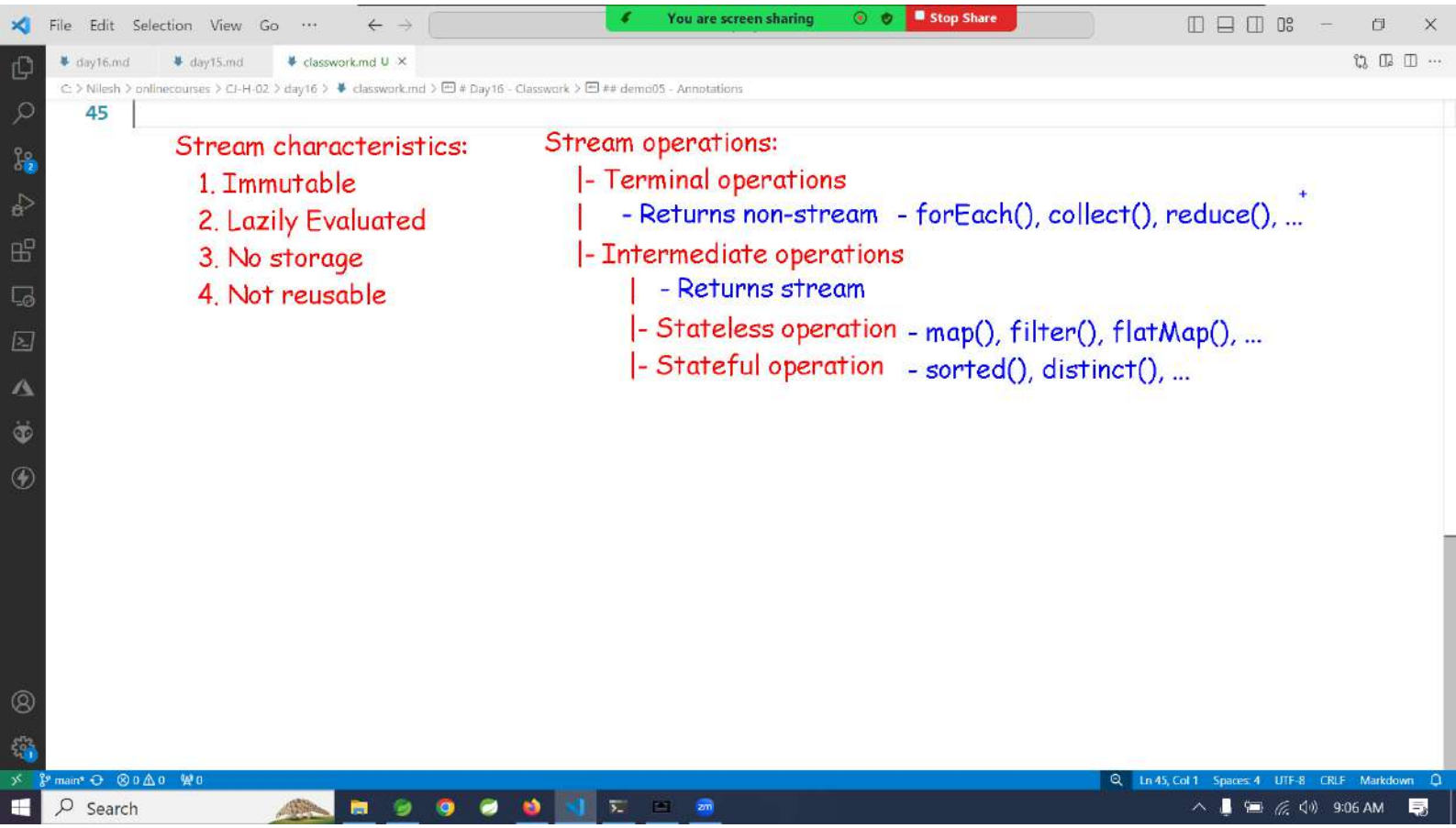
File Edit Source Refactor Navigate Search Project Run Window Help

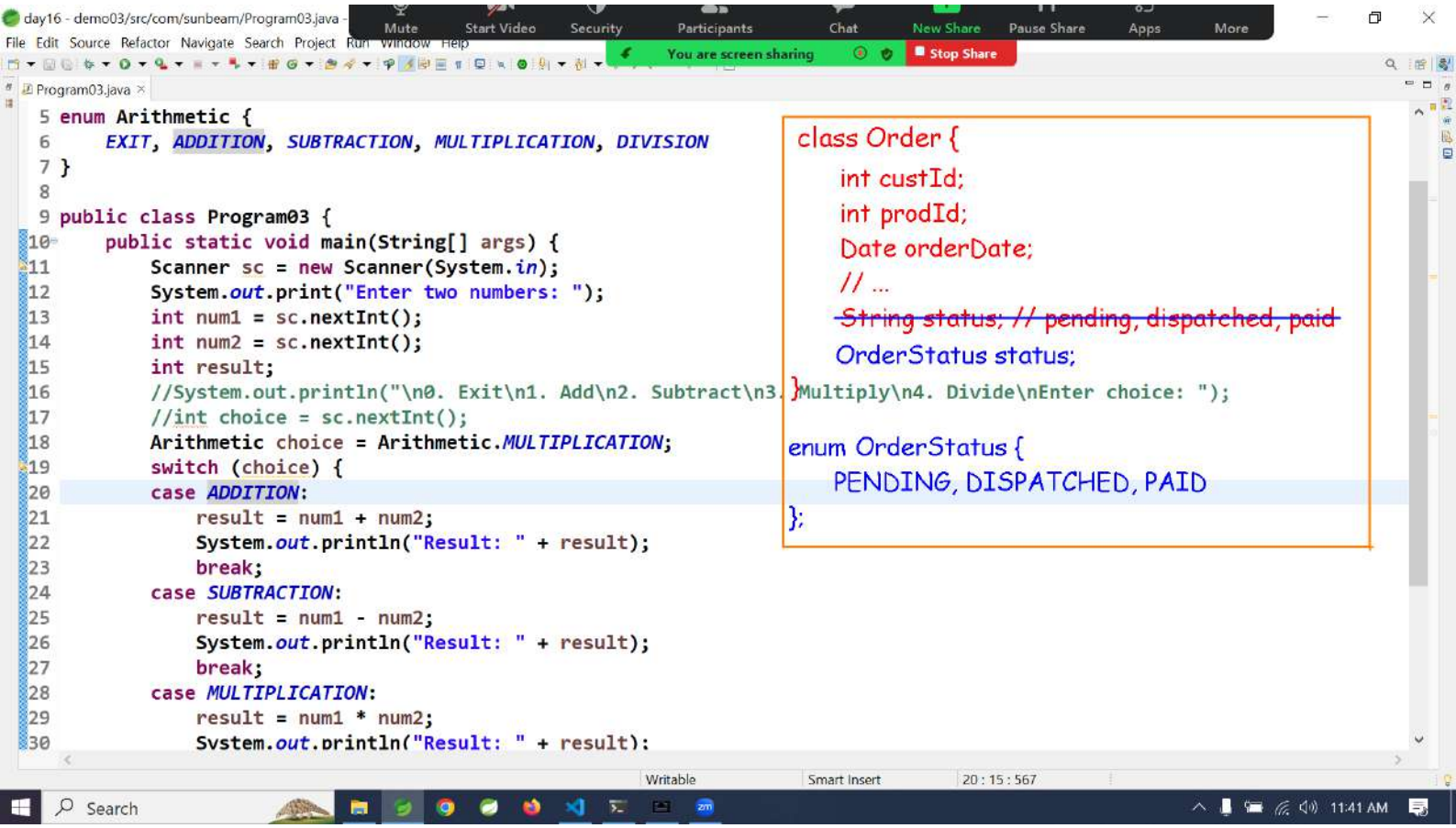
Program05.java

```
59      }
60      */
61
62      public static void main(String[] args) {
63          Stream.of(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
64              .map(n -> {
65                  System.out.println("map() -- square -- " + n);
66                  return n * n;
67              })
68              .filter(n -> {
69                  System.out.println("filter() -- odd nums -- " + n);
70                  return n % 2 != 0;
71              })
72              .sorted((x,y) -> {
73                  System.out.println("sorted() -- " + x + " <> " + y);
74                  return y - x; // desc sort
75              })
76              .map(n -> {
77                  System.out.println("map() -- prefix Java -- " + n);;
78                  return "Java"+n;
79              })
80              .forEach(s -> System.out.println("forEach() -- " + s));
81          System.out.println("Bye!");
82      }
83  }
84
```

Writable | Smart Insert | 84 : 1 : 2472

Search | 1:32 PM


File Edit Selection View Go ···
You are screen sharing
Stop Share
day16.md
day15.md
classwork.md U
C: > Nilesh > onlinecourses > CJ-H-02 > day16 > classwork.md > # Day16 - Classwork > ## demo05 - Annotations
45
Stream characteristics:
1. Immutable
2. Lazily Evaluated
3. No storage
4. Not reusable
Stream operations:
|- Terminal operations
| - Returns non-stream - forEach(), collect(), reduce(), ...
|- Intermediate operations
| - Returns stream
|- Stateless operation - map(), filter(), flatMap(), ...
|- Stateful operation - sorted(), distinct(), ...
main*
Ln 45, Col 1
Spaces: 4
UTF-8
CRLF
Markdown
Search
9:06 AM


day16 - demo03/src/com/sunbeam/Program03.java - Eclipse IDE
Mute
Start Video
Security
Participants
Chat
New Share
Pause Share
Apps
More
File Edit Source Refactor Navigate Search Project Run Window Help
You are screen sharing
Stop Share
Program03.java
enum Arithmetic {
EXIT, ADDITION, SUBTRACTION, MULTIPLICATION, DIVISION
}
public class Program03 {
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
System.out.print("Enter two numbers: ");
int num1 = sc.nextInt();
int num2 = sc.nextInt();
int result;
//System.out.println("\n0. Exit\n1. Add\n2. Subtract\n3. Multiply\n4. Divide\nEnter choice: ");
//int choice = sc.nextInt();
Arithmetic choice = Arithmetic.MULTIPLICATION;
switch (choice) {
case ADDITION:
result = num1 + num2;
System.out.println("Result: " + result);
break;
case SUBTRACTION:
result = num1 - num2;
System.out.println("Result: " + result);
break;
case MULTIPLICATION:
result = num1 * num2;
System.out.println("Result: " + result);
class Order {
int custId;
int prodId;
Date orderDate;
// ...
String status; // pending, dispatched, paid
OrderStatus status;
}
enum OrderStatus {
PENDING, DISPATCHED, PAID
};
Writable
Smart Insert
20 : 15 : 567
Search
11:41 AM

You are screen sharing Stop Share

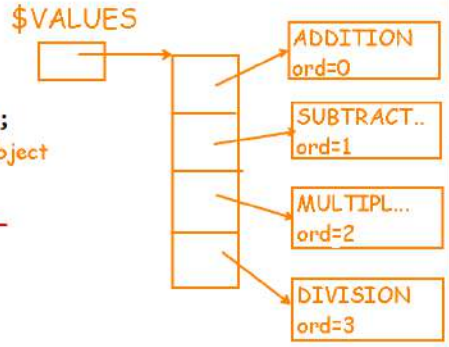
day16.md day15.md classwork.md U

day16.md > # Core Java > ## enum

```
102     // generated enum code
103     final class ArithmeticOperations extends Enum {
104         public static ArithmeticOperations[] values() {
105             return (ArithmeticOperations[])$VALUES.clone();
106         }
107         public static ArithmeticOperations valueOf(String s) {
108             return (ArithmeticOperations)Enum.valueOf(ArithmeticOperations, s);
109         }
110         private ArithmeticOperations(String name, int ordinal) {
111             super(name, ordinal); // invoke sole constructor Enum(String,int);
112         }
113         public static final ArithmeticOperations ADDITION;
114         public static final ArithmeticOperations SUBTRACTION;
115         public static final ArithmeticOperations MULIPLICATION;
116         public static final ArithmeticOperations DIVISION;
117         private static final ArithmeticOperations $VALUES[];
118         static {
119             ADDITION = new ArithmeticOperations("ADDITION", 0);
120             SUBTRACTION = new ArithmeticOperations("SUBTRACTION", 1);
121             MULIPLICATION = new ArithmeticOperations("MULIPLICATION", 2);
122             DIVISION = new ArithmeticOperations("DIVISION", 3);
123             $VALUES = (new ArithmeticOperations[] {
124                 ADDITION, SUBTRACTION, MULIPLICATION, DIVISION
125             });
126         }
```

"ADDITION" --> ADDITION object

enum ArithmeticOperations {
  ADDITION, SUBTRACTION,..
}



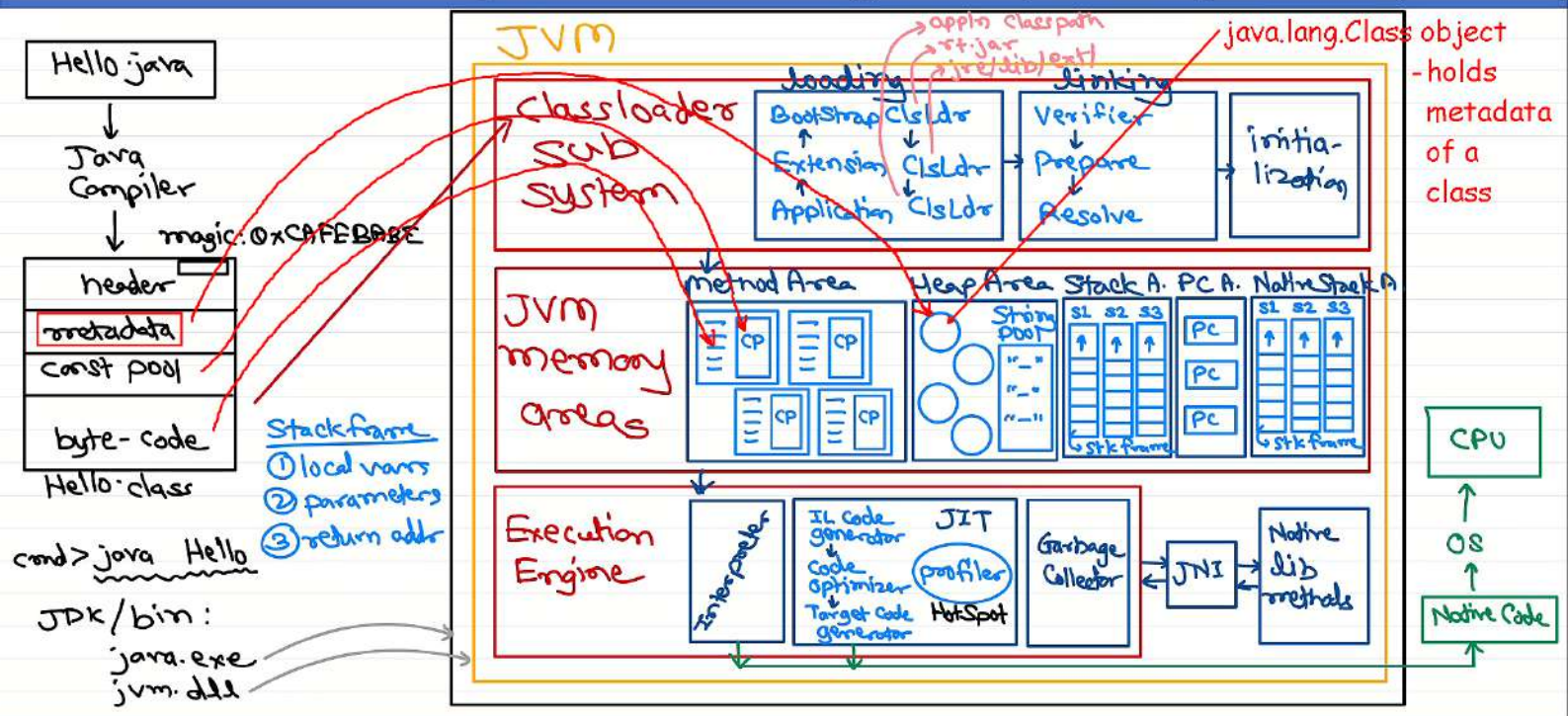
java.lang.Enum class

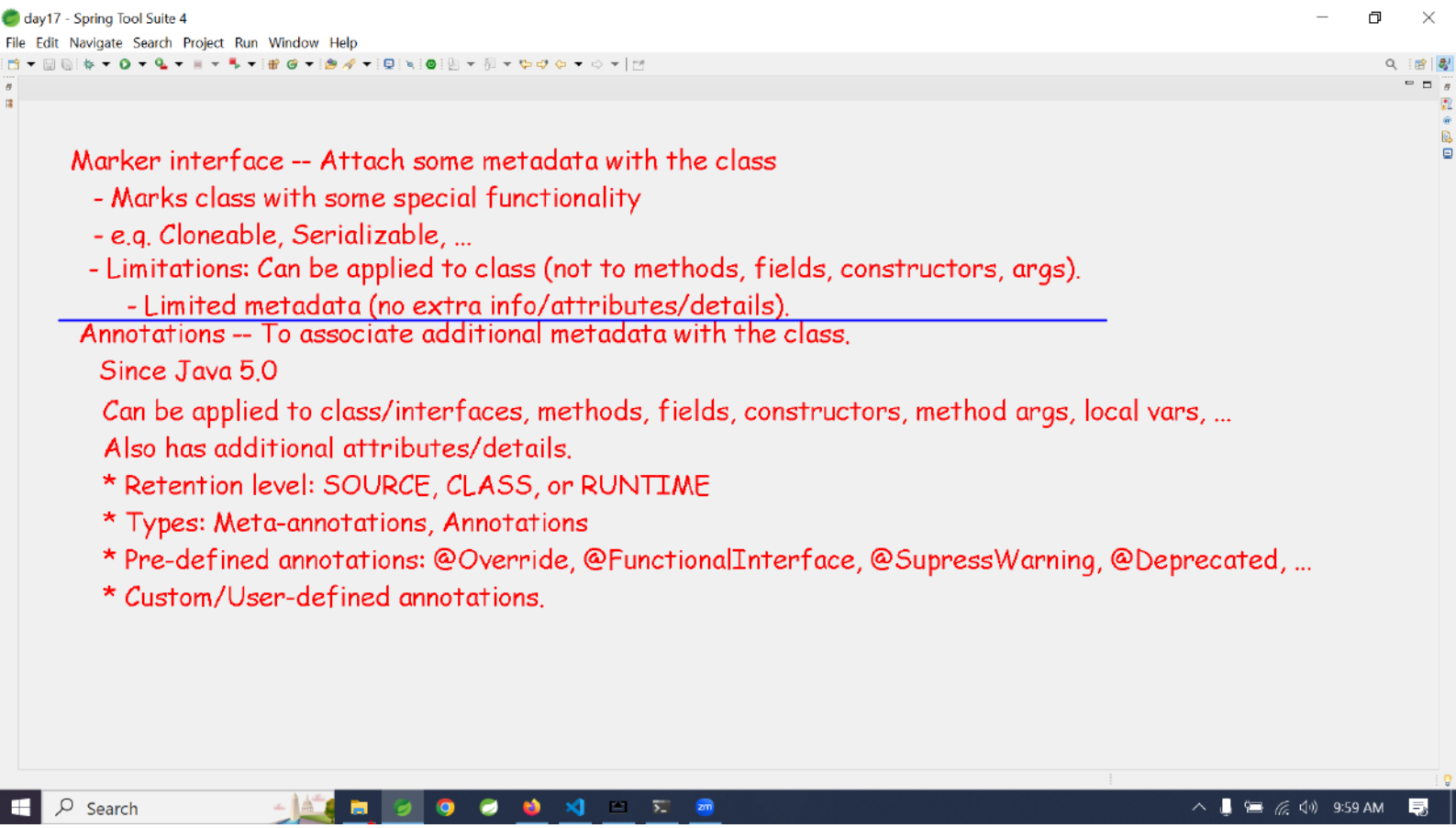
1. fields: name String, ordinal int.
2. methods:
   a. label() -- returns name string
   b. ordinal() -- returns pos/ord
   c. Enum(label,ordinal) -- ctor
   d. toString() -- returns name.
   e. valueOf() -- String to enum.

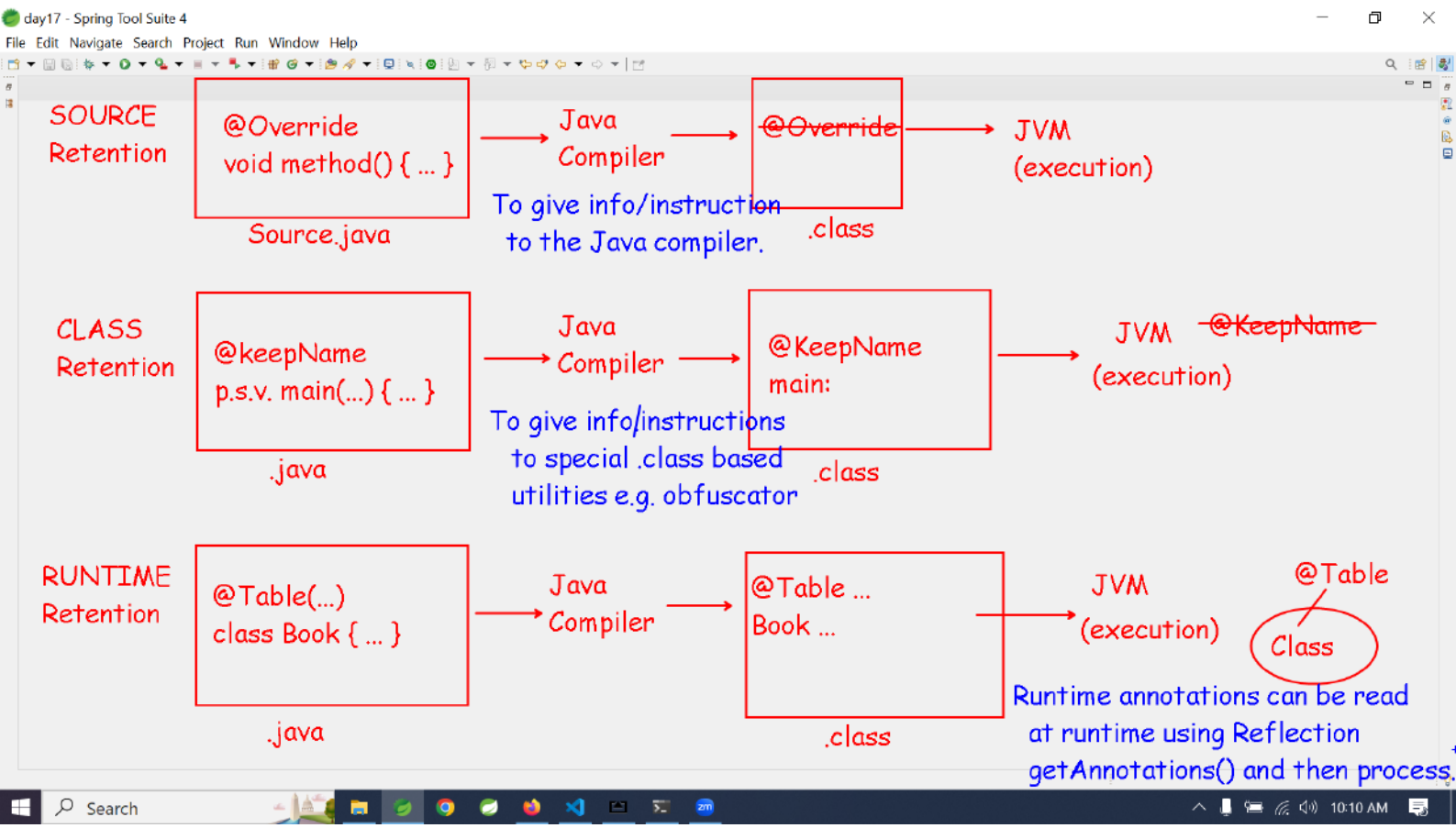
main* 0 0 0 Ln 98, Col 56 Spaces: 4 UTF-8 CRLF Markdown

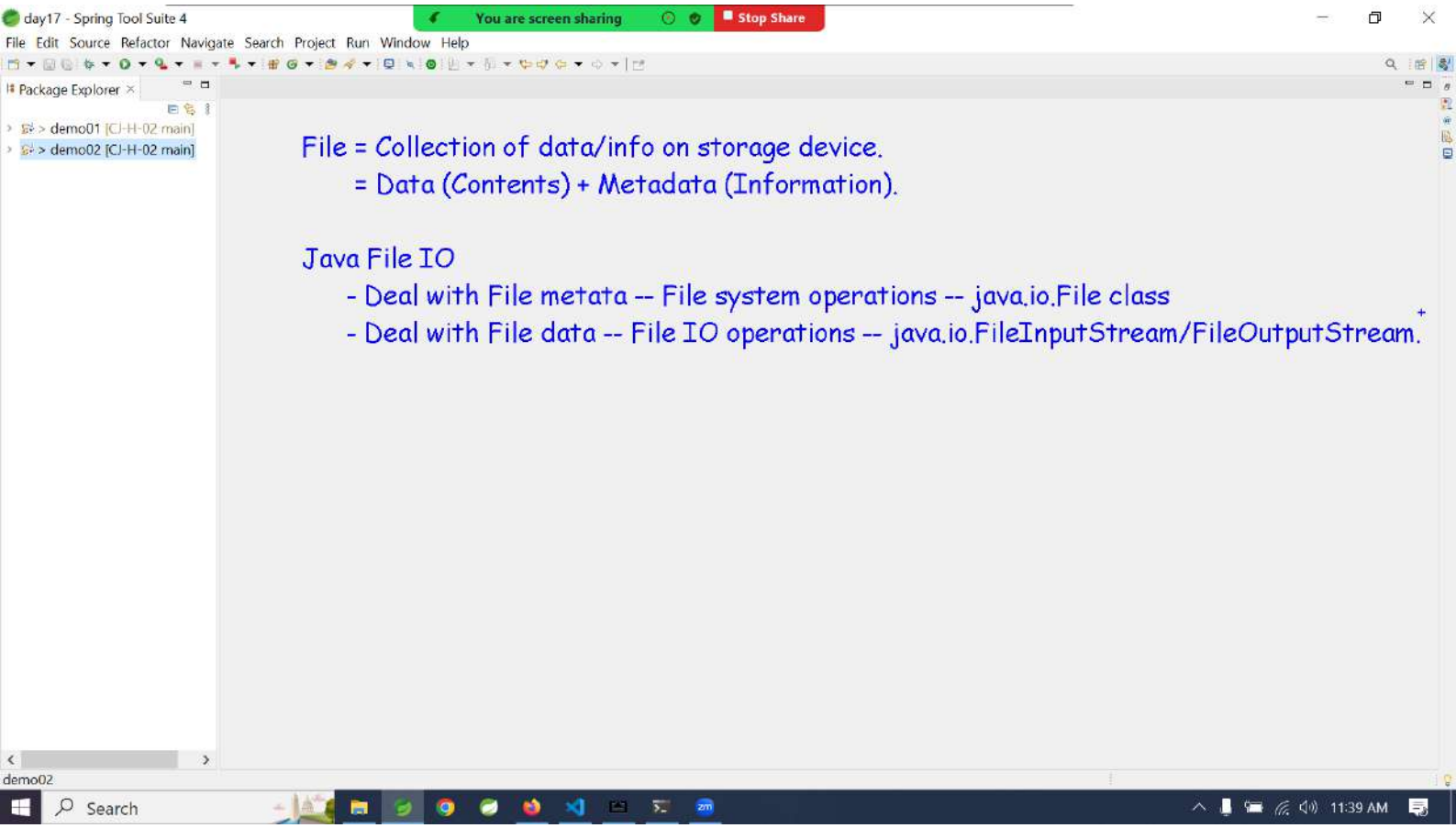
JVM architecture

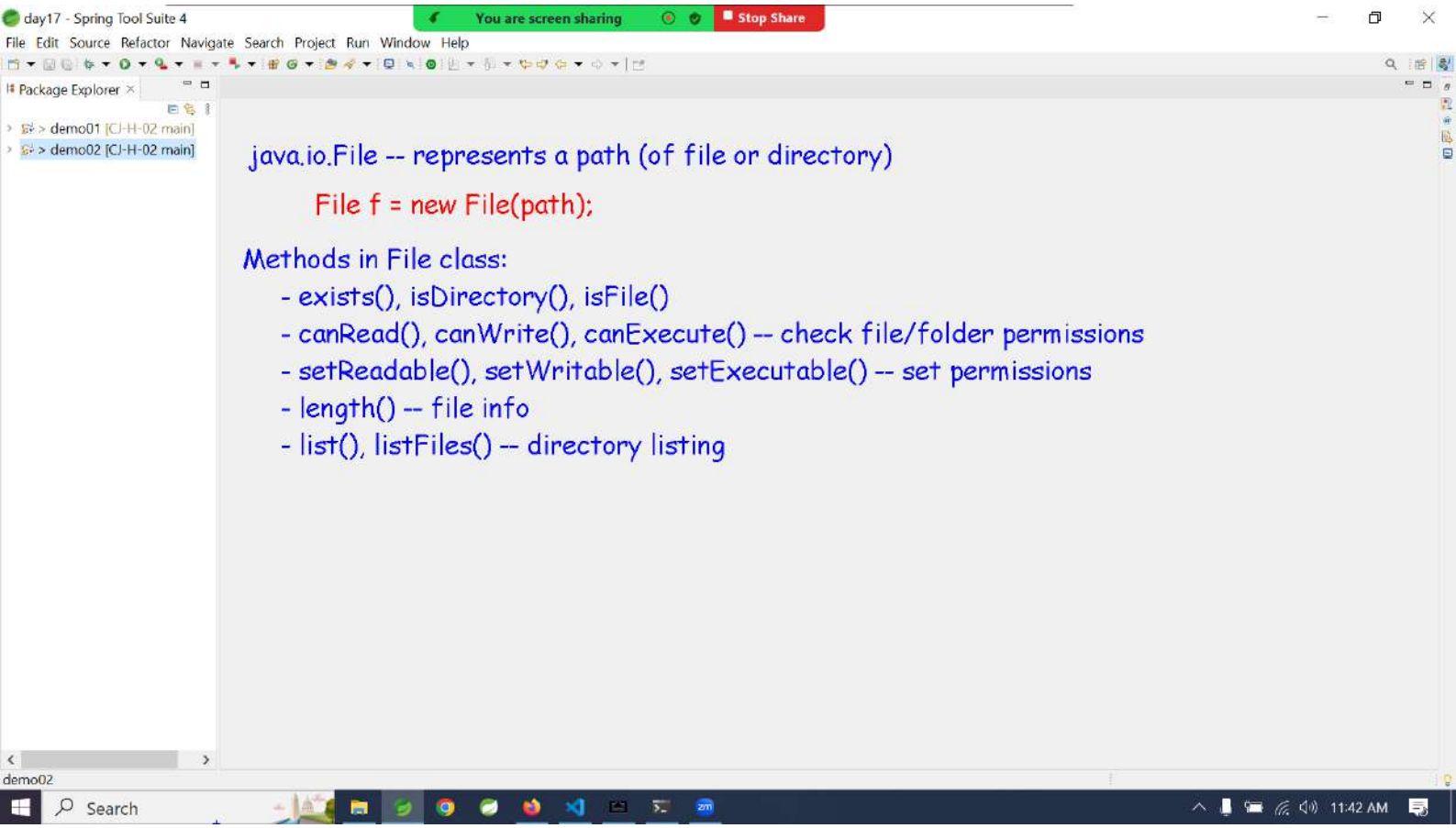
OS process for "java" (java app launcher)




day17 - Spring Tool Suite 4
File Edit Navigate Search Project Run Window Help
Marker interface -- Attach some metadata with the class
- Marks class with some special functionality
- e.g. Cloneable, Serializable, ...
- Limitations: Can be applied to class (not to methods, fields, constructors, args).
- Limited metadata (no extra info/attributes/details).
Annotations -- To associate additional metadata with the class.
Since Java 5.0
Can be applied to class/interfaces, methods, fields, constructors, method args, local vars, ...
Also has additional attributes/details.
* Retention level: SOURCE, CLASS, or RUNTIME
* Types: Meta-annotations, Annotations
* Pre-defined annotations: @Override, @FunctionalInterface, @SupressWarning, @Deprecated, ...
* Custom/User-defined annotations.
Search
9:59 AM


day17 - Spring Tool Suite 4
File Edit Navigate Search Project Run Window Help
SOURCE Retention
@Override
void method() { ... }
Source.java
Java Compiler
@Override
.class
JVM (execution)
To give info/instruction to the Java compiler.
CLASS Retention
@keepName
p.s.v. main(...) { ... }
.java
Java Compiler
@KeepName
main:
.class
JVM
@KeepName
(execution)
To give info/instructions to special .class based utilities e.g. obfuscator
RUNTIME Retention
@Table(...)
class Book { ... }
.java
Java Compiler
@Table ...
Book ...
.class
JVM
(execution)
@Table
Class
Runtime annotations can be read at runtime using Reflection getAnnotations() and then process.
Search
10:10 AM


day17 - Spring Tool Suite 4
You are screen sharing
Stop Share
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
> demo01 [CJ-H-02 main]
> demo02 [CJ-H-02 main]
File = Collection of data/info on storage device.
= Data (Contents) + Metadata (Information).
Java File IO
- Deal with File metata -- File system operations -- java.io.File class
- Deal with File data -- File IO operations -- java.io.FileInputStream/FileOutputStream.
demo02
Search
11:39 AM


day17 - Spring Tool Suite 4
You are screen sharing
Stop Share
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
> demo01 [CJ-H-02 main]
> demo02 [CJ-H-02 main]
java.io.File -- represents a path (of file or directory)
File f = new File(path);
Methods in File class:
- exists(), isDirectory(), isFile()
- canRead(), canWrite(), canExecute() -- check file/folder permissions
- setReadable(), setWritable(), setExecutable() -- set permissions
- length() -- file info
- list(), listFiles() -- directory listing
demo02
Search
11:42 AM