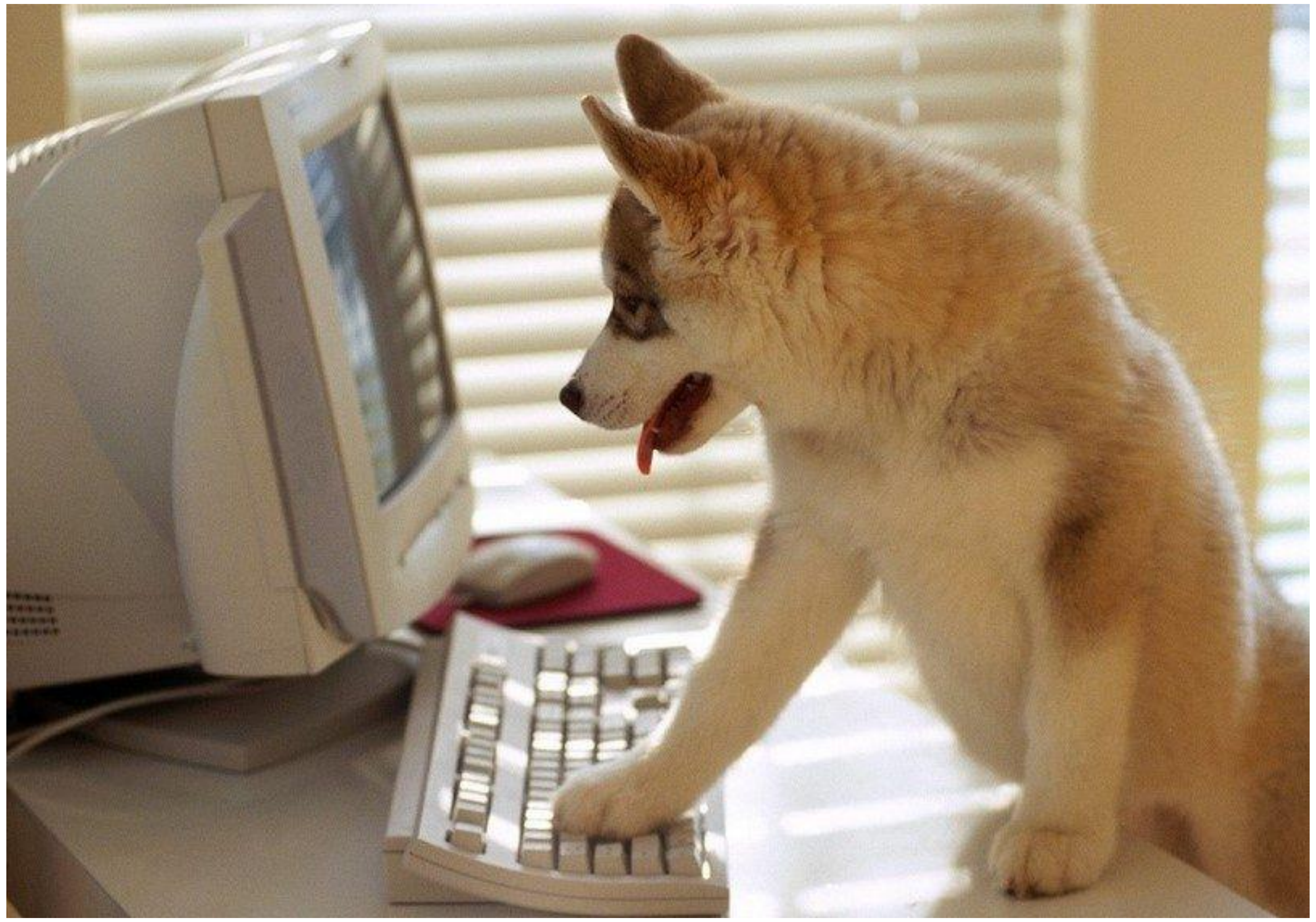
Programming Tutorial [Advanced]



Lambda Expressions

Functional Interface = An Interface consisting just of one method to be implemented

Lambda Expressions

```
JButton btn = new JButton("Hello");
btn.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent e) {
         System.out.println("Hello World!");
      }
});
```

```
JButton btn = new JButton("Hello");
btn.addActionListener((e) ->
         {System.out.println("Hello World!");}
);
```

forEach()

```
List<String> list = Arrays.asList(
                    “value1”,
                    “value2”,
                    “value3”);
for(String value : myList){
  System.out.println(value);
}
```

```
List<String> list = Arrays.asList(
                    “value1”,
                    “value2”,
                    “value3”);
list.forEach(new Consumer<String>(){
  public void accept(String value){
    System.out.println(value);
  }
});
```

forEach()

```
List<String> list = Arrays.asList(
                    "value1",
                    "value2",
                    "value3");
list.forEach(new Consumer<String>(){
  public void accept(String value){
     System.out.println(value);
  }
});
```

```
List<String> list = Arrays.asList(
                    "value1",
                    "value2",
                    "value3");
list.forEach(value -> System.out.println(value));
```

Streams

```
Collection<String> list =
          Arrays.asList(“Hello”,
                        “World”,
                        “And”
                        “Hello”,
                        “Java”);
long counter = list.stream().filter(
             new Predicate<String>(){
                @Override
                public boolean test(
                        String element){
                   return element.length>4
                }
             }
).count()
```

```
Collection<String> list =
          Arrays.asList(“Hello”,
                        “World”,
                        “And”
                        “Hello”,
                        “Java”);
long counter = list.stream()
                   .filter(
                      element->element.length>4)
                    .count()
```

Intermediate Operations

```
map()

sorted()

unsorted()

distinct()

limit()

peek()
```

Terminal Operations

```
sum()

min()

max()

reduce()

findFirst()
```

Example File IO

```
try(Stream<Stream> stream = Files.lines(Paths.get(FILENAME))){
   List<String> lines = stream.collect(Collectors.toList());
} catch(IOException e){
   e.printStackTrace();
}
```

Lambda and Streams

In this course we will use Github Classroom

1. Get a Github Account if you don’t have one
2. Go to: https://classroom.github.com/a/Efy3G1nD (or scan the QR Code with your phone)
3. Authorize Github and accept the assignment
4. Click on the repository