Java Practical with Lambdas

Objectives

The objectives of this practical session are to:

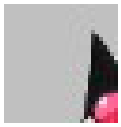
- Experiment with Lambdas and Streams for Data Processing

Overview

This uses Lambdas to process a large number of `Employees` in a variety of ways.

Practical

Recreating the Project Code and Using Lambdas



1. Use your existing project in IntelliJ, from the *Collections* exercises.
2. With your full `List` of `Employee` objects, you can call the `stream()` method with will return a `Stream<Employee>`. This then allows you to do processing using the Stream API, with accept Lambda expressions for doing optimal functional processing.
3. Recall that Streams are immutable, and that any operations you do on them return new Streams, possibly of different type, if you do a mapping for example.

In the following steps, use a single line of code, chaining Stream API code, starting from the original `List`.

4. Print out all the `Employees` (hint: use the `forEach()` method!).
5. Print out the names of all the `Employees` (use the `map()` method).
6. Filter the `Employees` based on age (only those under 35), and print out their names.
7. Print out all the departments, listing each only once.
8. Use the `collect()` function, with a `groupingBy()`. Print out the numbers of Employees in each department:

   ```
   emplist.stream().collect(groupingBy(e ->
                                e.getDepartment()));
   ```

   Note the `groupingBy` function is a static method in `Collectors`. Where have we seen something similar before?