SWENGB - LECTURE 06

AGENDA

- Java8 Lambda Expressions
- ListView example

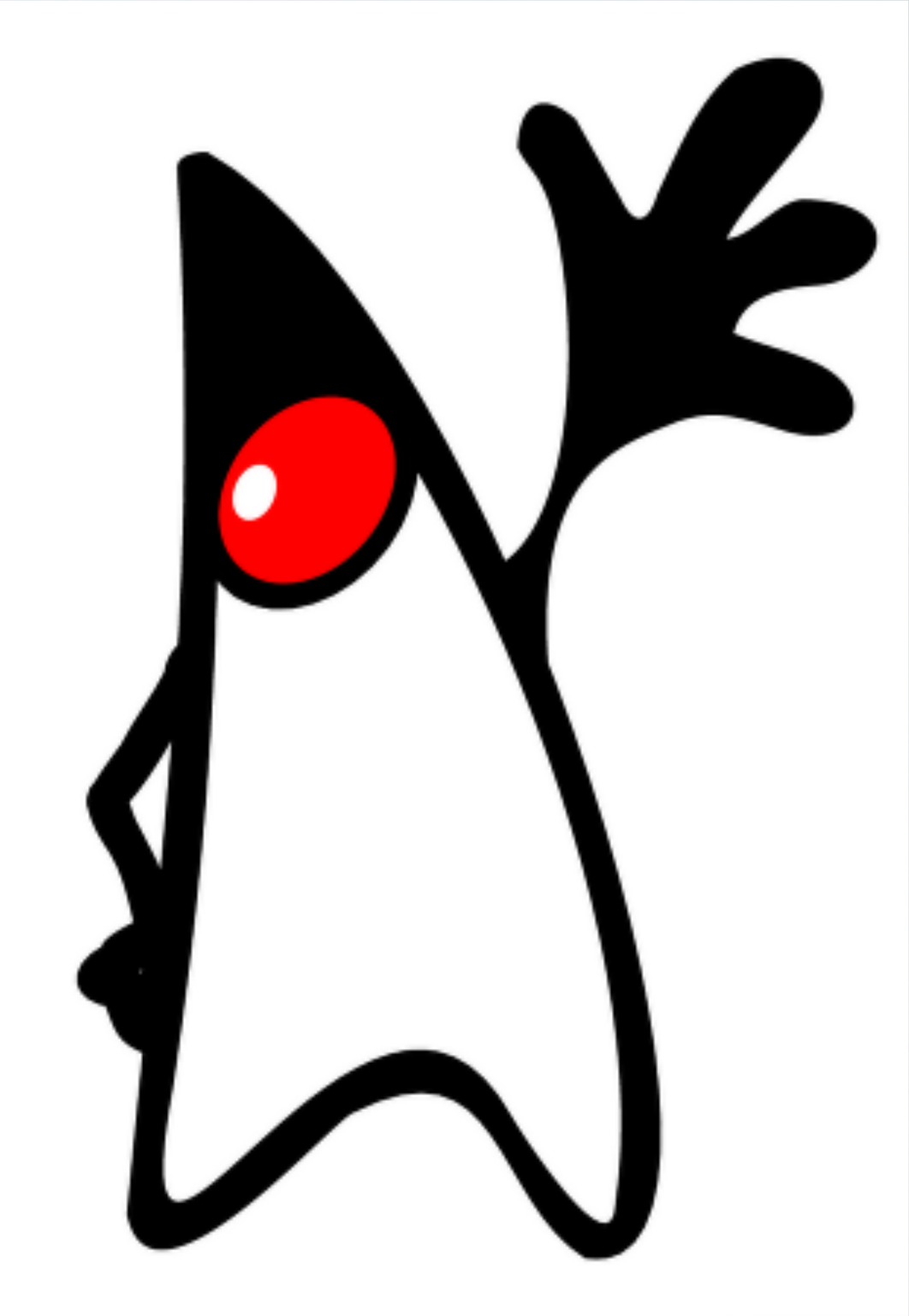
Lambda calculus

From Wikipedia, the free encyclopedia

Lambda calculus (also written as **λ-calculus**) is a formal system in mathematical logic for expressing computation based on function abstraction and application using variable binding and substitution.

JAVA8 LAMBDA EXPRESSIONS

Comparison Java vs Scala
(looking at SAM's)



```java
public class Person {

    public enum Sex {
        MALE, FEMALE
    }

    String name;
    LocalDate birthday;
    Sex gender;
    String emailAddress;

    public int getAge() {
        // ...
    }

    public void printPerson() {
        // ...
    }
}
```

```java
public class Person {

    public enum Sex {
        MALE, FEMALE
    }

    String name;
    LocalDate birthday;
    Sex gender;
    String emailAddress;

    public int getAge() {
        // ...
    }

    public void printPerson() {
        // ...
    }
}
```

```java
interface CheckPerson {
    boolean test(Person p);
}
```

```java
public static void printPersons(
    List<Person> roster, CheckPerson tester) {
    for (Person p : roster) {
        if (tester.test(p)) {
            p.printPerson();
        }
    }
}
```

```java
public class Person {

    public enum Sex {
        MALE, FEMALE
    }

    String name;
    LocalDate birthday;
    Sex gender;
    String emailAddress;

    public int getAge() {
        // ...
    }

    public void printPerson() {
        // ...
    }
}
```

```java
interface CheckPerson {
    boolean test(Person p);
}
```

```java
public static void printPersons(
    List<Person> roster, CheckPerson tester) {
    for (Person p : roster) {
        if (tester.test(p)) {
            p.printPerson();
        }
    }
}
```

```java
class CheckPersonEligibleForSelectiveService implements CheckPerson {
    public boolean test(Person p) {
        return p.gender == Person.Sex.MALE &&
            p.getAge() >= 18 &&
            p.getAge() <= 25;
    }
}
```

```
public class Person {

    public enum Sex {
        MALE, FEMALE
    }

    String name;
    LocalDate birthday;
    Sex gender;
    String emailAddress;

    public int getAge() {
        // ...
    }

    public void printPerson() {
        // ...
    }
}
```

```
interface CheckPerson {
    boolean test(Person p);
}
```

```
public static void printPersons(
    List<Person> roster, CheckPerson tester) {
    for (Person p : roster) {
        if (tester.test(p)) {
            p.printPerson();
        }
    }
}
```

```
class CheckPersonEligibleForSelectiveService implements CheckPerson {
    public boolean test(Person p) {
        return p.gender == Person.Sex.MALE &&
            p.getAge() >= 18 &&
            p.getAge() <= 25;
    }
}
```

```
printPersons(
    roster, new CheckPersonEligibleForSelectiveService());
```

Java can be read by everybody, but it comes with a price.

LAMBDA EXPRESSIONS

```
printPersons(
    roster,
    (Person p) -> p.getGender() == Person.Sex.MALE
        && p.getAge() >= 18
        && p.getAge() <= 25
);
```

http://docs.oracle.com/javase/tutorial/java/javaOO/lambdaexpressions.html

SCALA

```scala
case class Person(age: Int)

def printPersons(f: Person => Boolean): Unit = {...}

printPersons(_.age >= 18)
```

LAMBDA EXPRESSIONS II

```
roster
    .stream()
    .filter(
        p -> p.getGender() == Person.Sex.MALE
            && p.getAge() >= 18
            && p.getAge() <= 25)
    .map(p -> p.getEmailAddress())
    .forEach(email -> System.out.println(email));
```

http://docs.oracle.com/javase/tutorial/java/javaOO/lambdaexpressions.html

LISTVIEW

SelectionModel
CellFactory