UCLouvain

icteam

Inner classes & Lambda functions

External class

```
interface MyFunction {
    public int apply(int value);
}

class Square implements MyFunction {
    @Override
    public int apply(int value) {
        return value * value;
    }
}

public class Version1 {
    private MyFunction createSquare() {
        return new Square();
    }

    public void run() {
        MyFunction f = createSquare();
        System.out.println("Square: " + f.apply(4));
    }

    public static void main(String args[]) {
        new Version1().run();
    }
}
```

Inner class (inside a class)

```
interface MyFunction {
    public int apply(int value);
}

public class Version2 {
    private class Square implements MyFunction {
        @Override
        public int apply(int value) {
            // NB: Here, I could also access the members of "Version2"!
            return value * value;
        }
    }

    private MyFunction createSquare() {
        return new Square();
    }

    public void run() {
        MyFunction f = createSquare();
        System.out.println("Square: " + f.apply(4));
    }

    public static void main(String args[]) {
        new Version2().run();
    }
}
```

Inner class (inside a method)

```
interface MyFunction {
    public int apply(int value);
}

public class Version3 {
    private MyFunction createSquare() {
        class Square implements MyFunction {
            @Override
            public int apply(int value) {
                // NB: Here, I could also access the members of
                // "Version3" and the variables of "createSquare()"!
                return value * value;
            }
        }

        return new Square();
    }

    public void run() {
        MyFunction f = createSquare();
        System.out.println("Square: " + f.apply(4));
    }

    public static void main(String args[]) {
        new Version3().run();
    }
}
```

Anonymous inner class

```
interface MyFunction {
    public int apply(int value);
}

public class Version4 {
    private MyFunction createSquare() {
        return new MyFunction() {
            @Override
            public int apply(int value) {
                return value * value;
            }
        };
    }

    public void run() {
        MyFunction f = createSquare();
        System.out.println("Square: " + f.apply(4));
    }

    public static void main(String args[]) {
        new Version4().run();
    }
}
```

Lambda function

```
interface MyFunction {
    public int apply(int value);
}

public class Version5 {
    private MyFunction createSquare() {
        return (value) -> value * value;
    }

    public void run() {
        MyFunction f = createSquare();
        System.out.println("Square: " + f.apply(4));
    }

    public static void main(String args[]) {
        new Version5().run();
    }
}
```


UCLouvain