Refactoring Dojo

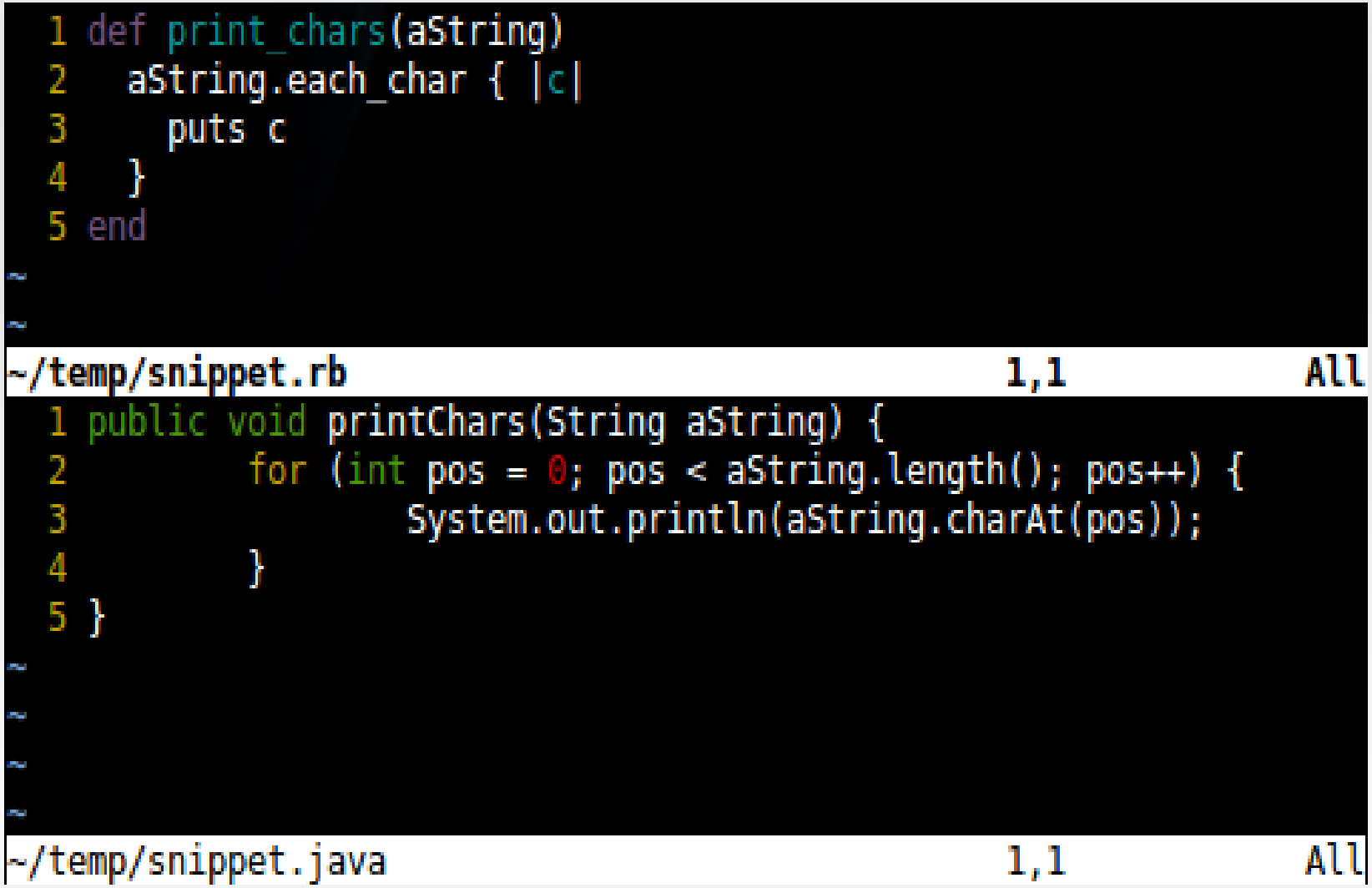
- Qué es refactoring?
- Refactoring en Eclipse
- Kata Euler

¿Qué es refactoring?

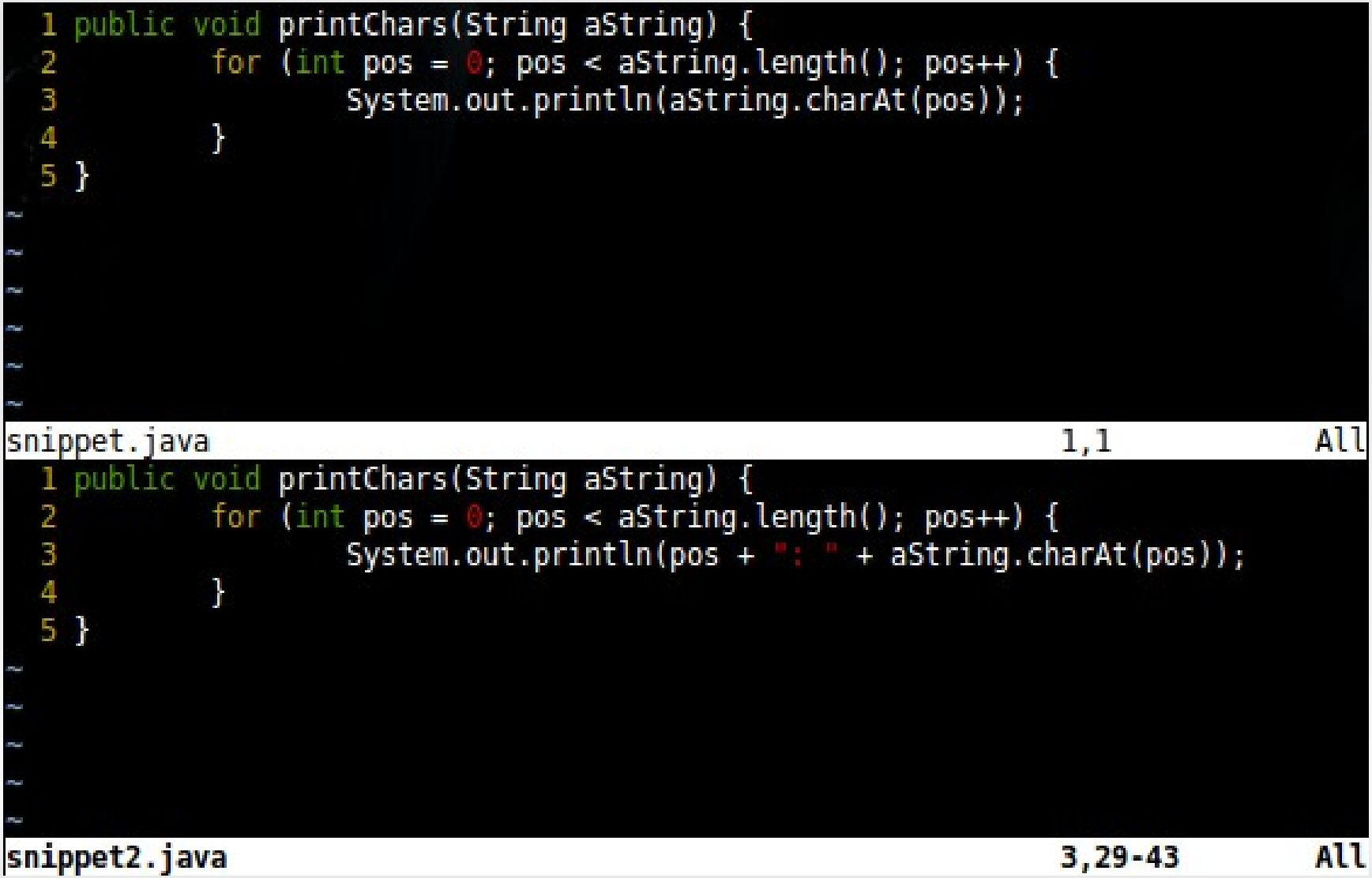
```
public static boolean checkFormat(String pattern, String guess) {
    if (pattern.length() != guess.length())
        return false;
    for (int pos = 0; pos < pattern.length(); pos++) {
        char pattChar = pattern.charAt(pos);
        char guessChar = guess.charAt(pos);
        switch (pattChar) {
        case 'D':
            if (guessChar < '0' || guessChar > '9') {
                return false;
            }
            break;
        default:
            if (guessChar != pattChar) {
                return false;
            }
        }
    }
    return true;
}
```

```java
public static boolean checkFormat(String pattern, String guess) {
    if (notSameLength(pattern, guess))
        return false;
    return compareCharByChar(pattern, guess);
}

private static boolean compareCharByChar(String pattern, String guess) {
    for (int pos = 0; pos < pattern.length(); pos++) {
        if (!charVerifiesPattern(guess.charAt(pos), pattern.charAt(pos)))
            return false;
    }
    return true;
}

private static boolean charVerifiesPattern(char guessChar, char pattChar) {
    if (pattChar == 'D') {
        return (!isDigit(guessChar));
    } else {
        return (guessChar == pattChar);
    }
}

private static boolean isDigit(char guessChar) {
    return !(guessChar < '0' || guessChar > '9');
}

private static boolean notSameLength(String pattern, String guess) {
    return pattern.length() != guess.length();
}
```

¿Qué no es refactoring?

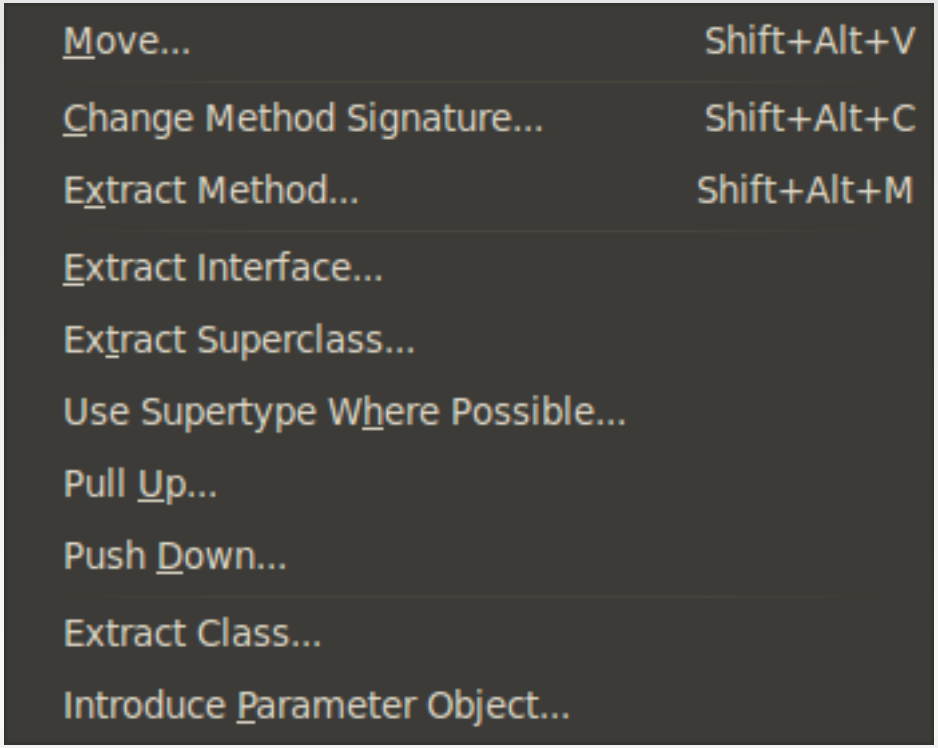


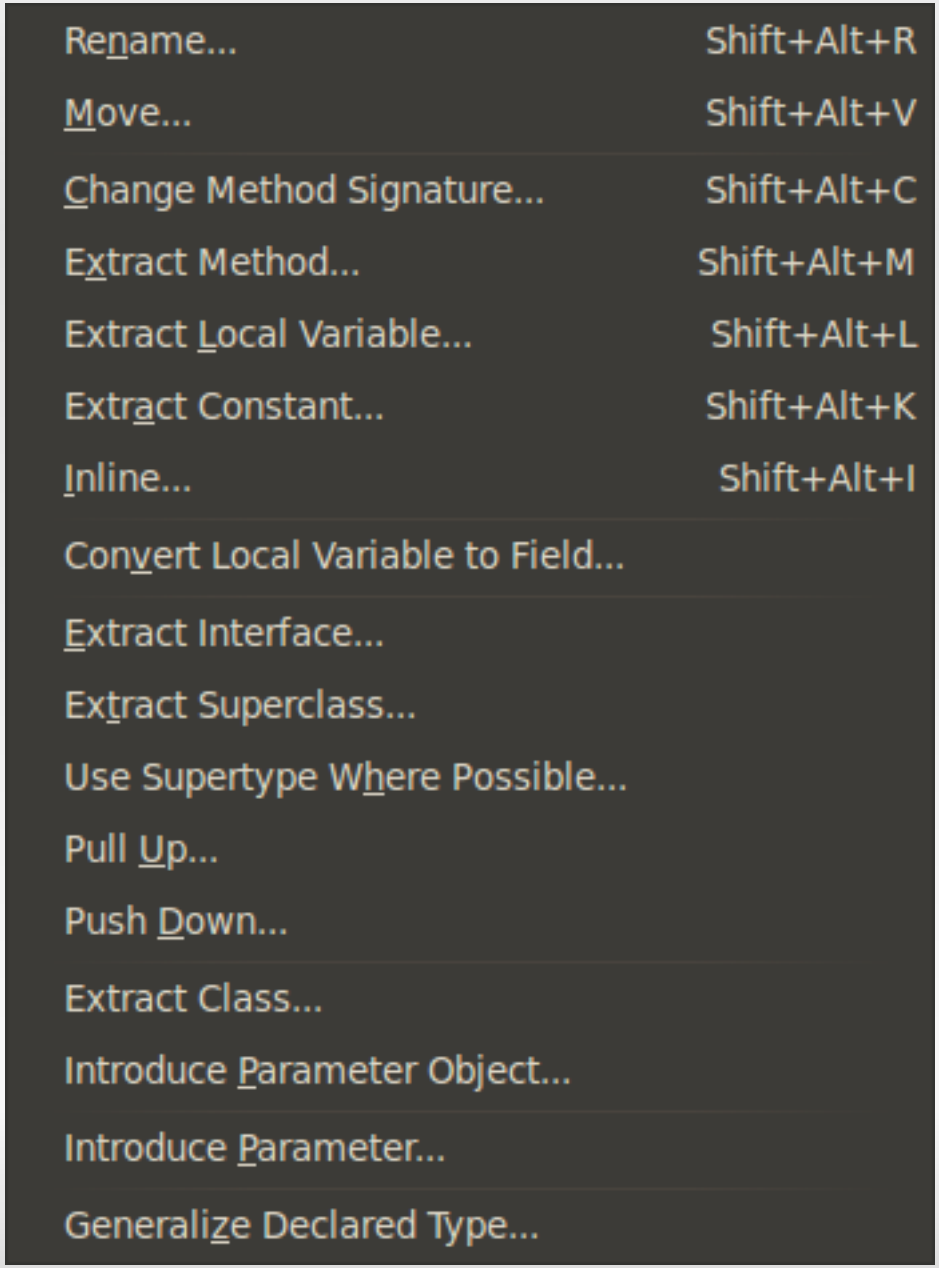
¿Qué no es refactoring?



Refactoring en Eclipse

Alt + Shift + T





Kata Euler

https://github.com/pepellou/KataEuler

Kata Euler

El propósito de esta kata es practicar refactors sobre código de carácter algorítmico. Las katas de refactoring son muy útiles para permitir a programadores de todos los niveles dialogar y compartir formas de entender la programación sobre código existente. La intención de esta kata no es enfocar el debate a ninguna cuestión en concreto, aunque el tipo de problema que se resuelve (problemas matemáticos de cierta complejidad computacional) se ha buscado intencionadamente como escenario típico donde el código pierde legibilidad y/o eficiencia, siendo estos dos temas mi sugerencia para centrar el debate. También se ha perseguido otra cuestión típica de refactoring, como es la reutilización de código, escogiendo como punto de partida un código que resuelve de forma separada varios problemas similares.

Descripción de la Kata

Por cuestiones en las que no vamos a profundizar, hemos recibido un golpe de iluminación vital y hemos decidido que debemos resolver todos los problemas de Project Euler para estar en paz con nosotros mismos. Como sabemos que es un camino largo, empezamos por resolver algunos de los más sencillos, concretamente los problemas 1, 2 y 10 (véase tag start-kata-here).

Es en este punto donde se nos plantea una nueva duda metafísica: ¿vamos por el buen camino? ¡Pues claro que no! Acabamos de resolver apenas 3 problemas y ya nos cuesta entender nuestro propio código. Si queremos resolver cientos de problemas necesitaremos refactorizar nuestro código para:

1. Ser capaces de entenderlo solo con leerlo
2. Reutilizar código para no reinventar la rueda a cada paso
3. Reutilizar cálculos para no tardar siglos en resolver los problemas