Estimados

esta es una explicación para orientarlos sobre el punto 5 ej 6 del tp4.

El objetivo del ejercicio es que por un lado vean más implementaciones del patrón template method y también
practiquen extendiendo templates existentes o implementando nuevos.
Vamos a cambiar el enunciado levemente para que sea explicable por este medio (y por que el enunciado original involucraba un nivel innecesario de detalle de complejidad(*)).

Nuesrto enunciado era:

*Extienda la implementación para que al terminar cada test case se imprima en el transcript el **resultado** del test. (debe indicar el nombre del método que se testeó y si pasó el test, o fue error o failure).*

Pero vamos a trabajar con este simplificación.

*Extienda la implementación para que al terminar cada test case se imprima en el transcript el **nombre** del test. (debe indicar el nombre del método que se testeó. Ejemplo: MiClaseTest>>testHelloWorld).*

Al final explicaremos por qué este cambio(*).

Una parte clave de todo este ejercicio es el método runCase en TestCase

```
runCase
    self resources do: [:each | each availableFor: self].
    [self setUp.
    self performTest] ensure: [
        self tearDown.
        self cleanUpInstanceVariables]
```

Como verán es un template con ciertos pasos.
Ahora lo que queremos es tener la posibilidad de imprimir en la medida que se ejecutan los tests.

Sabemos que el mecanismo para cambiar comportamiento en un paso de un template es redefiniendo ese paso en una subclase.

Una cosa interesate del codigo de runCase es que el bloque que es parametro de ensure: siempre se ejecuta, no importa
si ocurre o no una excepción en el bloque ejecuta el test (el que hace performTest).

Dentro del bloque del ensure tenemos dos hooks.

Vamos a tomar tearDown que ya es conocido (donde "limpiamos"aquellos recursos que usa nuestro test).
Entonces podemos definir una subclase de TestCase que refine tearDown.

Nuestra subclase de testCase definiría algo como

```
tearDown
  super tearDown.
  self log
```

En log obviamente podemos hacer el log al Transcript.

```
log
  Transcript
     show: self printString;
     cr;
     flush
```

Para lograr que nuestros Tests impriman su nombre en el transcript alcanza entonces con subclasificar nuestra subclase de TestCase.

Les dejamos entonces una última tarea para reflexionar sobre esta solución y la implementación final que hagan:

- Supongan que mañana se les pide loggear los nombres de los tests a un archivo, cómo lo harían?
- Qué pasa si tambien quisieran guardarlos en un string (supongamos una variable global)?
- Como estructurarían su solución?

(*) Respecto al por qué es necesario hacer ese pequeño cambio en el enunciado, el problema radica en que en runCase no disponen del resultado del test. De todas maneras el enunciado original se peude resolver reimplementado otró metodo y con más codigo... si bien es factible agrega complejidad que no hace a lo que nos interesa que aprendan en esta práctica/ejercicio.

Saludos

Arturo