David

Yatzy refactoring kata

Esta kata está basado en el juego de dados Yatzy.

Se trata de un kata sobre refactorización, TDD y casos test.

Además, sirve para introducir el concepto de clase "estática"

Prepara el proyecto

1. Lee las reglas del juego en el README.MD del proyecto.
2. Clona este repo de Emily Bache para hacerte con el código con el que trabajaremos. El kata está en diversos lenguajes:
3. https://github.com/emilybache/Yatzy-Refactoring-Kata
4. **Crea un repo PUBLICO en GitHub donde publicar tu proyecto DESDE EL PRIMER MOMENTO. Para este kata es fundamental practicar el flujo de la TDD de manera estricta.**
5. Trabaja con las ramas que consideres adecuadas cuando la refactorización suponga un cambio estructural en la clase o en más de un método.

Este kata exige el trabajo con TDD, motivo por el cual incluye los casos test en **Pytest**.

1. Instala el módulo Pytest siguiendo las instrucciones proporcionadas en: http://doc.pytest.org/en/latest/getting-started.html
2. Si el comando pip no funciona en tu máquina, necesitarás instalar el gestor de módulos de Python pip. Sigue las instrucciones proporcionadas en cualquiera de estos sitios:
   https://pip.pypa.io/en/stable/installing/
   http://recursospython.com/guias-y-manuales/instalacion-y-utilizacion-de-pip-en-windows-linux-y-os-x/
   http://diegorys.es/2016/09/01/como-instalar-el-gestor-de-paquetes-pip-en-windows-7/

Refactorizar

Qué es la refactorización

Martin Fowler la define como:

*"a change made to the internal structure of the software to **make it easier to understand** and **cheaper** to modify without changing its **observable behavior"**.*

En nuestro caso particular:

"A big refactoring is a recipe for disaster." —Kent Beck 📞 @Elmo

En el capítulo 24 Refactoring del libro de Code Complete, échale un ojo a las checklist del **capítulo 24 Refactoring** donde se recogen de modo esquemático los puntos a chequear cuando te encuentras ante un código a refactorizar y utiliza estas listas de chequeo para sanear tu código.

https://docs.google.com/document/d/1_FCBb__1IMo5tzh9kMrsdcp3fuQimGvYPGrdT7PC1-8

CHECKLIST: Summary of Refactorings
- [ ] Data Level Refactorings
- [ ] Replace a magic number with a named constant.
- [ ] Rename a variable with a clearer or more informative name.
- [ ] etc.

Code smells

Toma nota de los code smells que hayas identificado.
Tienes una **checklist** con los code smells más habituales en el libro Code Complete, en el capítulo 24 Refactoring:

https://docs.google.com/document/d/1_FCBb__1IMo5tzh9kMrsdcp3fuQimGvYPGrdT7PC1-8

Cada vez que termines una función, pásale estas checklist a tu código. Asegúrate de que estos code smells no están presentes en tu código refactorizado.

Reflexiona sobre estas cuestiones a medida que avanzas en la refactorización del código:
- ¿Cuánto código duplicado hay en tu solución? ¿Y en tus casos test?
- ¿Escribiste una lista de casos test antes de comenzar?
- ¿Cómo decidiste el orden en el que implementar los casos test?

Solución

En mi repo en GitHub:
https://github.com/dfleta/Python_ejercicios/tree/master/Poo/Yatzy_Refactoring_TDD_Kata