BrainTraining

Por: Sebastian Rojas Cortez
Para: Programación Orientada a Objetos
Prof: Eric Jeltsch, Universidad de La Serena
Fecha: Martes 29 de Noviembre, 2016

Índice

Introducción.

A continuación se abordará el proceso de creación del programa informático “BrainTraining”, para la asignatura de Programación Orientada Objetos (POO) de la carrera Ingeniería en computación de la Universidad de La Serena. Se revisará la pauta entregada por el profesor y se puntualizará en el cumplimiento de estos..

Justificación.

La motivación de este informe es la petición realizada por el profesor, ya que forma parte de la pauta evaluativa entregada por éste. También será útil para poder revisar y dejar constancia escrita del desarrollo del software, el cumplimiento de la pauta al momento de la evaluación final y valorar el contenido aprendido y cómo fue puesto en práctica en la creación de éste software.

Objetivo.

Escribir un programa que pueda extender las clases dadas con el fin de cumplir con todos los conceptos de la POO y las librerías Swing. Para tal efecto, se entrega una versión preliminar de círculos en movimiento, los que se cruzan en el frame.

En principio se tienen 4 clases:

1. Proy_base.java: el main frame para el programa
2. Panel_animacion.java: el panel que despliega los movimientos del círculo
3. Mover_figura.java: una clase abstract para mover figuras
4. Mover_circulo.java: una clase concreta para mover círculos

Ud. podrá cambiar o transformarlas, pero sin perder de vista el objetivo y el contexto. En todo caso, la misma propuesta debería socializarse entre sus compañeros para saber quién es el más rápido de reacción, dejando algunos resultados en la base de datos. Además, cualquier otra propuesta de figuras, deberán estar implementadas en una clase separada. Se deberá también considerar los comandos que el usuario ha pulsado, para ver o estudiar la usabilidad del mismo. En lo que respecta a la implementación, la propuesta deberá tener al menos TODAS estas etapas:

1. Otro tipo de figura (triángulo)
2. Destruir la figura (hacerla desaparecer)
3. Escoger las figuras
4. Destrucción selectiva
5. Marcador o scores para medir
6. Botones
7. Campo de texto,
8. Fuentes y colores.

El juego debe terminar, entregando un ganador, para eso defina las reglas y los objetivos que hacen a un ganador y por ende a un perdedor.

Además se deberá chequear Errores: Someter a pruebas el programa.

Los Criterios de Diseño a utilizar: Se realiza usando entornos de desarrollo, ya sea Netbeans, http://www.netbeans.org/, Eclipse http://www.eclipse.org/ , BlueJ http://www.bluej.org/, u otros, tomando las precauciones del caso, advertidas en clase.

Uso de bases de datos, por ejemplo, MySQL u otro motor.

Desarrollo.

Se comenzó con la visita a la página guía brainmetrix donde se revisó el catálogo de juegos y se tomó Reflex Text como inspiración para uno de los juegos implementados en este software (más precisamente el mini juego “BrainClick”). El Juego se consideró relativamente fácil de implementar, por lo que se decidió agregar más de uno al proyecto general con la intención de “innovar” o realizar algo más de lo solicitado por el profesor.

Teniendo ya una idea del software a implementar se fue directamente al código, partiendo por crear los paquetes para delimitar el patrón MVC, se siguió con el lado de la Vista (frame, paneles, botones, etc), creando así un frameBase que podría ser útil para más de un juego (que era la idea principal). Una vez teniendo listo el frameBase, se pasó a la introducción del programa, los mensajes de introducción, la identificación del usuario, la selección del juego y el traspaso de los datos del usuario al frame (panel superior). Al lograr esto, se prosiguió con implementar el tablero del juego BrainClick en el panel medio (jp2) utilizando Graphics e integrando los listener necesarios. Cabe destacar que este parte de la implementación fue el primer pequeño obstáculo en el desarrollo del software, ya que fue la primera vez que se trabajaba con Listener y no se tenía claro cómo realizar las acciones de click, repintado del tablero, entre otros. Una vez logrado, utilizando clases Timer, Listener, método repaint(), se logró dar por logrado el juego.

Acabado el juego BrainClick se consideró proseguir con un nuevo juego, queriendo acogerse al programa ejemplo entregado en la pauta del profesor (Coaliciones de figuras), pero se pensó que sería mejor implementar en primera instancia una barra de menú, al implementarse se volvió a la codificación del nuevo juego. Agregado el Bar y sus Listener, se remodeló el frameBase y otras clases para poder encajarlos en este nuevo juego, sin mayor dificultad. Hecho esto, se implementó un nuevo tablero para este juego, también utilizando Graphics, se creó una clase especial para sus figuras y se agregaron los listener necesarios. Esta parte fue probablemente la más problemática en el su desarrollo, sobretodo por sus figuras hexagonales o la figura de estrella en particular, ya que al momento de realizar su movimiento se debía mover cada uno de sus puntos en el tablero, lo que por un momento pareció imposible, pero se logró realizar mediante la implementación un par de métodos que rehacía la figura utilizando sus valores iniciales y asignando los del movimiento y otra escalaba la figura. También hubo otros problemas en el Listener, cómo que algunas figuras pasaban los márgenes del frame, pero también se pudo solucionar.

Logrado ya el segundo juego, se acabó por integrar la base de datos y su modelo, se partió con la idea de realizarlo mediante la herramienta WorkBench, con la cual no resultó la conección y proseguí con WAMP que era la más utilizada en los ejemplos

de internet. Teniendo ya el programa casi acabado (faltaban algunos detalles de interacción o visual) se prosiguió con la documentación, comenzando por el JavaDoc (que se realizó sin mayor dificultad) y los diagramas de UML. El Diagrama de clases se implementó mediante ObjectAid y se buscó una herramienta apropiada para realizar los siguientes diagramas, se intentó utilizar StratUML y UML2 (ahora MDT/Papyrus), pero pareció aparatoso y se encontró la herramienta PlantUML que genera diagramas a partir de un lenguaje propio. Con esta última se realizó el diagrama de secuencia y se intentó utilizar para otros tipos de diagramas pero aparecieron errores que no se pudo solucionar y se decidió realizar el diagrama de casos de uso en StartUML.

Acabado los diagramas, se arreglaron algunos detalles del programa y se logró transformarlo a exe mediante la herramienta Launch4j.

Sobre el Software.

El software fue pensado primeramente como un juego de desarrollo mental, esto por falta de información e ignorancia en el área, pero ya una vez investigado el tema, se precisó que el programa podría ser considerado de reacción o de mero entretenimiento, ya que el desarrollo mental mediante juegos interactivos no está comprobado y los estudios no lo avalan (noticia, estudio). Fue desarrollado en el IDE Eclipse neon

El software corre también sobre cualquier computadora que tenga instalado java 1.4.0 o superior, ya sea Windows, Linux u otro sistema operativo. (el exe solo corre en windows, para otros utilizar el código base), también no exige mayor memoria ya que solo pesa 7mb.

El lenguaje utilizado ha sido Java ya que es orientado a objetos y el cual se enseña para la asignatura.

Documentación.

JavaDoc. (Ubicado en la carpeta Doc dentro de la carpeta POO)

Diagramas (Ubicado en la carpeta img dentro de la carpeta POO)

Ud puede encontrar todo el código y cómo fui avanzando en el proyecto en:
Github

Pruebas realizadas.

Sonia Cortez 37 años, nivel informático básico.
No tuvo problemas para llevar a cabo las tareas requeridas por el software, pero puntualizó en que el juego era bastante básico y no tenía mayor complejidad, lo que le hacía algo aburrido.

Jorge Sanhueza 41 años, nivel medio bajo
Sin problemas para realizar las actividades del software, su comentario fue que el juego no parecía del todo entretenido y que él no lo usaría con regularidad.

Conclusión.

Como se logra notar en este informe, los objetivos requeridos por el profesor fueron cumplidos en su mayoría, si bien no fue estrictamente apegado, se daba también la flexibilidad en cuanto al modelo, herramientas y codificación al momento del desarrollo. Si bien el software en sí no representa una mayor innovación en el área o no llegaría a ser un boom popular, podemos notar que se logró aprender y emplear los conocimientos enseñados en la asignatura.

Netgrafía.

stackoverflow
PlantUML
MDT/Papyrus
ObjectAid
chuidiang
JavaApi
java2s
forosdelweb
chuidiang
java-white-box
jarroba
geekytheory
GWTLecturer
visual-paradigm

Bibliografía.

Deitel y Deitel - Cómo programar en Java