

```
global _scale_s  
name="y  
min=0.0  
default  
)  
  
def execute(sel  
  
# get the f  
folder_path  
  
# get obj  
viewport_  
  
# get exp  
obj_expor  
if self.u  
obj_e  
  
# deselect  
bpy.ops.obje  
  
for item in  
item.sel  
if item.  
file,  
bpy-4
```

UNIVERSIDAD NACIONAL DE LA PLATA
FACULTAD DE INFORMÁTICA

Informe final

María Emilia Romero
Francisco Venegas Naboulet

Seminario de Lenguajes - Python

09 DE AGOSTO DE 2022

Índice

Introducción	1
Temas estudiados	1
Problemas y soluciones surgidas durante el desarrollo	2
Consideraciones éticas sobre el desarrollo	3
Conclusiones y trabajos futuros	4
Referencias	6
Anexo 1: guía de usuario	7
Anexo 2: guía para el desarrollador	13

Introducción

El presente informe describe cómo se ha efectuado la implementación del juego *FiguRace*, desarrollado en la materia *Seminario de Lenguajes - Python* de la Facultad de Informática (UNLP).

FiguRace es un juego educativo basado en tarjetas, el cual muestra en pantalla una tarjeta con una serie de datos al usuario, pertenecientes a distintas categorías. Pudiéndose seleccionar para jugar de entre las siguientes: Spotify, Netflix y Lagos de Argentina. Estos datos representan características que permiten que el jugador identifique el objeto o persona a la que hace referencia la tarjeta dentro de un tiempo configurable.

A medida que el usuario avanza en el juego, suma o resta puntos de acuerdo a su desempeño. El juego finaliza luego de una cantidad determinada de tarjetas o cuando el jugador lo requiera, utilizando el botón salir.

A continuación, se detallarán los módulos y herramientas utilizadas para la implementación del mismo, así como la forma en la que se desarrolló el proyecto.

Temas estudiados

A lo largo del desarrollo del Trabajo Integrador, el equipo, y también con impulso de la cátedra, vio la necesidad de utilizar distintas librerías, módulos y métodos para llevarlo a cabo. En particular, se utilizaron las librerías PySimpleGUI, Pandas y Matplotlib para la implementación de la interfaz gráfica, análisis y procesamiento de datos, y gráfico de datos respectivamente.

En concreto, “Pandas es una herramienta de fuente abierta fácil de usar, poderosa y flexible para el análisis y manipulación de datos [...]” (Pandas, 2022), y fue utilizada para el procesamiento previo de los datos utilizados por el juego y, posteriormente, para el tratamiento y almacenamiento de la información obtenida a partir del mismo. En particular, esta librería facilitó notablemente el procesamiento de los datos necesarios para el software, que primero se había realizado con el módulo CSV propio del lenguaje de programación Python.

Por otra parte, una de las herramientas utilizadas para este proyecto fue PySimpleGUI, una librería que, al igual que Pandas, es de código abierto y permite la implementación de una interfaz gráfica de manera sencilla. Es gracias a esta

herramienta, que se instrumentó todo lo visible para el usuario dentro del software *Figurace*.

Adicionalmente, en el presente Trabajo se utilizó Matplotlib para exponer aquellos resultados y conclusiones, obtenidas de la implementación de Pandas para el análisis de la información de juego. Según su desarrollador, John Hunter, “Matplotlib intenta que sean fáciles las cosas fáciles, y posibles las cosas difíciles” (John Hunter, s.f.). Es una librería que resulta sencilla de utilizar, sin necesidad de un gran entendimiento de su funcionamiento.

Hay que mencionar además, el uso de un entorno virtual para el desarrollo del trabajo. Para esto, se utilizó la librería *virtualenv*, la cual “es una herramienta usada para crear un ambiente aislado de Python. Este ambiente tiene su propio directorio de instalación que no comparte librerías con otros ambientes virtualenv” (Seminario de Lenguajes - Opción Python, 2022). Esta herramienta resulta una gran ventaja para el trabajo, ya que permite instalar las librerías necesarias para el proyecto, aislandolo de cualquier otro con el que se esté trabajando. De esta manera, se pueden evitar problemas de compatibilidad por posible manejo de distintas versiones de las librerías compartidas por diversos proyectos.

Adicionalmente, se han utilizado distintas técnicas para obtener, entre otros, una estructura de directorios y una forma de pensar los módulos, para la correcta división de tareas y partes a lo largo del trabajo. También, es importante destacar que se utilizó GitLab, una herramienta que permitió a cada programador desarrollar el software desde sus dispositivos, permitiendo controlar los cambios y obtener las modificaciones realizadas por los distintos miembros del equipo. En particular, esta herramienta ofrece una forma de visualizar los aspectos y tareas pendientes del proyecto, y asignarlas a los integrantes del grupo, que permitió una mejor organización del trabajo.

Problemas y soluciones surgidas durante el desarrollo

En un principio, a la hora de plantear el presente Trabajo, una parte de las pantallas fueron desarrolladas de forma individual y distinta entre cada una. Esto, a medida que la complejidad del mismo aumentó, trajo inconvenientes tanto de comprensión entre los integrantes del equipo y ayudantes como también problemas de comunicación entre módulos. Por ello, una vez identificada esta cuestión, se optó por el desarrollo de un módulo que cree ventanas, donde las instrucciones y variables entre las mismas sean sus argumentos, permitiendo así llevar la misma lógica pero también adaptar las pantallas de acuerdo a lo requerido.

No obstante, luego de ciertos avances en el Proyecto, este módulo recibió ciertas modificaciones y ajustes que le permitieran continuar cumpliendo con su función. Esto también fue posible gracias a la fragmentación de entregas para el Trabajo, incluyendo las pantallas más simples en la primera y las que más desarrollo precisaban en la segunda.

Uno de los grandes desafíos que el trabajo dejó en evidencia, fue comprender y aprender a utilizar las funciones propias que ofrece Python, tales como lambda, map, filter, zip, reduce, entre otras, teniendo en cuenta que mayormente nuestros desarrollos de código habían sido en Pascal. Si bien estas funciones fueron de gran utilidad y permitieron un código más sencillo, corto y legible, poder establecer dónde era adecuado emplearlas no siempre resultaba una labor sencilla. Fue con la continua escritura y consulta, tanto con los ayudantes como entre compañeros, que se pudo realizar la reimplementación de varias secciones de código con tales funciones, revelando una mejora notable con respecto a su utilización entre la primera y la segunda entrega del Trabajo Integrador.

Por otra parte, el espacio de consulta con el que se dispuso fue de gran ayuda para la constante mejora del Trabajo. A lo largo de las consultas, el trabajo fue progresando y superando los obstáculos que se presentaron. Así, por ejemplo, tanto la ventana de la configuración como la de puntajes migraron de ser varias ventanas, a una sola con todos sus atributos. La pantalla principal de puntajes contaba con tres botones, uno para cada dificultad, y una vez elegido alguno era otra ventana la que se mostraba con sus respectivos datos. Esto fue comentado por los ayudantes, y se recomendó unificar las mismas en una única pantalla con las tres dificultades. Notando así, la gran importancia del aprovechamiento de estos espacios para el progreso del software.

Consideraciones éticas sobre el desarrollo

Partiendo de la definición de software libre que indica que, “A grandes rasgos, [...] los usuarios tienen la libertad de ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software” (Free Software Foundation, s.f.), se cuenta con varias ventajas al momento de utilizarlo. En primer lugar, al contar con acceso al código resulta más sencillo entender el funcionamiento de ciertas funciones y/o aspectos del mismo. Por ejemplo, como se mencionó en el apartado anterior, uno de los retos de empezar a trabajar con el lenguaje, fue comprender sus funciones ya definidas, pero al tener disponible su implementación se facilita el entendimiento de las mismas y, por tanto, su adecuado uso.

Habiendo terminado el trabajo y reflexionando en sus alcances, podemos concluir que el software cuenta con algunas restricciones en cuanto su accesibilidad. El juego no responde a las necesidades de, por ejemplo, condiciones como el daltonismo, aspecto que podría mejorarse para generar una mayor inclusión al mismo. Si bien los desarrolladores creen que la paleta de colores no definirá el desempeño del usuario en el juego, este aspecto no fue considerado al momento de la elección de colores del mismo.

Respecto de la dependencia de un sistema operativo específico, se puede afirmar que *Figurace* está pensado para ser jugado tanto en Windows como en Linux. Gracias a las sugerencias de la cátedra, se ha logrado realizar la implementación del código de modo que sea ejecutable en los sistemas operativos mencionados. Esto es mediante el uso del módulo `os`, la cual “provee una manera versátil de usar funcionalidades dependientes del sistema operativo.” (Python Software Foundation, 2022)

Conclusiones y trabajos futuros

En líneas generales, y repasando lo que fue el desarrollo de este Trabajo Integrador, creemos que el mismo nos introdujo a lo que es el mundo real del software y los Proyectos, ya que no contábamos con la experiencia de realizar el desarrollo de un programa desde cero y en equipo. Esto implicó que cada uno de los integrantes estudiemos, investiguemos y probemos distintas herramientas, soluciones y funcionalidades para llegar al resultado final de *Figurace*.

Trabajar con una herramienta que permita el intercambio del Trabajo entre cada uno de los integrantes del equipo, con la posibilidad de contar con los cambios de los mismos a la hora de ponerse a trabajar en el software, fue de gran utilidad y ayudó mucho en los aspectos organizativos. A su vez, trabajar siguiendo la metodología “Kanban” de letreros y tarjetas, con las actividades, prioridades y fechas límite de cada demanda, fue fundamental para el correcto ordenamiento del equipo.

Al principio, la implementación del juego nos resultó muy intuitiva, hasta que nuestros ayudantes nos propusieron otra estructura para nuestro directorio, y por ende, una diferente organización del código. Si bien al comienzo esa organización, y en particular el módulo para la creación de ventanas, nos confundió, luego de una mejor comprensión de su funcionalidad, concluimos que era mejor que la forma que teníamos antes de su corrección.

Debido a la falta de tiempo y al establecimiento de ciertos aspectos con mayor prioridad que otros para la entrega del trabajo, quedan para su futura implementación ciertas características que ampliarían el proyecto en varios aspectos.

Entre ellos, se encuentra la inclusión de la totalidad de los datasets propuestos por la cátedra, con el fin de no limitar a solo tres categorías al usuario. Por otra parte, la adición de distintos atributos al juego, como un mejor gráfico de las tarjetas o pantallas, efectos de sonidos, música tanto en el menú como en el desarrollo del juego, un sistema de ayuda para adivinar la tarjeta, opción para elegir la paleta de colores del mismo, entre otros.

Finalmente, todas estas consideraciones y las sugerencias que puedan llegar por parte de los usuarios, constituirían un mejor y más completo desarrollo de *FiguRace*.

Referencias

pandas - Python Data Analysis Library [On-line]. Disponible en:
<https://pandas.pydata.org/docs/>

Matplotlib - Visualization with Python [On-line]. Disponible en: <https://matplotlib.org/>

Entornos virtuales - Python 2022 [On-line]. Disponible en:
https://python-unlp.github.io/2022/guias/07_entornos_virtuales/

¿Qué es el Software Libre? - Proyecto GNU - Free Software Foundation [On-line].
Disponible en: <https://www.gnu.org/philosophy/free-sw.es.html>

os — Interfaces misceláneas del sistema operativo — documentación de Python -
3.10.6 [On-line]. Disponible en: <https://docs.python.org/es/3/library/os.html>

Anexo 1: guía de usuario

Como se puede observar en la Figura 1, la pantalla inicial del juego presenta el menú principal de *FiguRace*, donde se visualizan los distintos botones que permiten a los jugadores acceder al resto de las pantallas o salir del juego. Se muestra, o bien la dificultad, el dataset y el usuario seleccionados por el jugador, o los determinados por defecto en caso de que no hayan sido elegidos. Tanto la categoría de dataset como el nivel pueden ser escogidos por el usuario en la pantalla *Configuración*.

Figura 1. Captura pantalla de Inicio



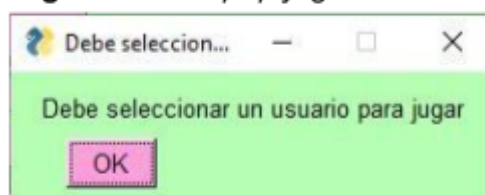
Antes de iniciar con el juego, es recomendable leer la explicación disponible en la pantalla *Cómo Jugar* (Figura 1.1), ventana con una breve exposición de cada una de las funcionalidades y pantallas presentes en el juego.

Figura 1.1. Captura pantalla Cómo Jugar



Para comenzar a jugar, es necesario haber seleccionado previamente un usuario, o alternatively, crearlo. En caso de que se intente jugar sin haberlo hecho, se mostrará una ventana emergente (en inglés Popup) en pantalla (Figura 1.2) y será redireccionado a la pantalla *Perfil* (Figura 2) para la creación o selección de un usuario.

Figura 1.2. Popup jugar sin selección de usuario



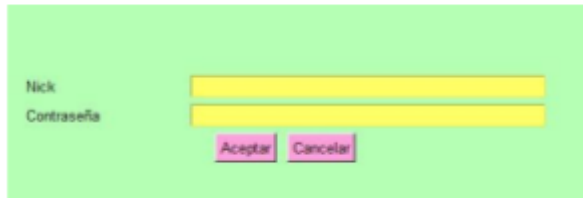
Una vez en la pantalla *Perfil*, se presentan distintas opciones con respecto al mismo: selección, creación y edición. La Figura 2 presenta las opciones mencionadas anteriormente.

Figura 2. Captura pantalla Perfil



Si se desea seleccionar un usuario, se le pedirá el nombre del mismo y su contraseña, quedando éste así seleccionado (Figura 2.1). Por otra parte, para la creación de un nuevo perfil se requiere el ingreso de determinados datos (Figura 2.2). Información que, luego es utilizada para un análisis de datos y gráfico de estadísticas sobre el juego. Estos datos son: alias o nick (abreviatura de *nickname*, en inglés) del jugador, la edad, el género autopercebido y una contraseña. En caso de que el nick ingresado ya exista entre los perfiles creados, no se permitirá generar un usuario con ese alias. Por último, se encuentra la opción de editar un perfil previamente seleccionado, siendo posible modificar únicamente la edad y el género autopercebido. Como se muestra en la Figura 2.3, se visualizan los datos del usuario a editar permitiendo cambiar aquel que desee.

Figura 2.1. Selección de usuario



Form for user selection. It has a light green background. On the left, the labels "Nick" and "Contraseña" are in black. To their right are two yellow input fields. Below the fields are two pink buttons: "Aceptar" and "Cancelar".

Figura 2.2. Creación de usuario



Form for user creation. It has a light green background. At the top, the text "Ingrese los siguientes datos:" is in black. Below it are four labels: "Nick", "Edad", "Género autopercebido", and "Contraseña". To the right of these labels are four yellow input fields. The "Género autopercebido" field has a small downward arrow on its right side. Below the fields are two pink buttons: "Crear" and "Cancelar".

Figura 2.3. Edición de usuario



Form for user editing. It has a light green background. At the top, the text "Figuracer: Emi" is in black. Below it, the text "Ingrese nuevos datos:" is in black. There are two labels: "Edad:" and "Género autopercebido:". To their right are two yellow input fields. The "Edad" field contains the number "19". The "Género autopercebido" field contains the word "Femenino" and has a small downward arrow on its right side. Below the fields are two pink buttons: "Aceptar" and "Cancelar".

Una vez seleccionado el perfil con el que se va a jugar, no es necesaria la elección ni de la categoría ni del nivel, ya que se cuenta con unos establecidos por defecto. Ahora ya es posible comenzar con el juego.

Como muestra la Figura 3, en la parte izquierda de la pantalla *Jugar* se ubica la información de la partida: el alias del jugador, la dificultad, el puntaje actual, la categoría de dataset y el tiempo restante. Por otro lado, se encuentra la tarjeta con características para la identificación del objeto o persona dada como opción en la misma.

Tanto la cantidad de datos mostrados en la tarjeta, el tiempo por ronda como los puntos obtenidos en cada respuesta, depende de la dificultad seleccionada, la cual es posible modificar en la pantalla de configuración. La tarjeta cuenta también, con la opción de pasar de ronda, dándose por perdida la misma.

Figura 3. Captura pantalla Jugar



Al finalizar la partida, se obtiene un puntaje de acuerdo con las respuestas dadas por el jugador. Si este se encuentra dentro de los mejores veinte puntajes de ese nivel, se podrá visualizar en la pantalla *Puntaje*. Como se puede observar en la Figura 4, la pantalla cuenta con dos tablas para cada dificultad: una con los mejores puntajes y otra con los mejores puntajes promedio, ambas ordenadas descendentemente y presentando, como máximo, veinte.

Figura 4. Captura pantalla Puntaje



Si bien el juego cuenta con una configuración por defecto para cada dificultad, ésta puede ser modificada por el jugador en la pantalla *Configuración*. La Figura 5 expone la disposición de la misma. Se muestra por un lado, los parámetros de la dificultad seleccionada y permite modificarlos, haciendo efectivo el cambio presionando el botón “Modificar dificultad”.

Figura 5. Captura pantalla Configuración

Configuración

Seleccionar dataset: Spotify Lagos Películas

Seleccionar dificultad: Facil Media Dificil

Ingrese el tiempo limite (en segundos): 30

Ingrese la cantidad de rondas por juego: 10

Ingrese el puntaje sumado por cada respuesta correcta: 1

Ingrese el puntaje restado por cada respuesta incorrecta: 0.5

Ingrese la cantidad de características a mostrar: 3

Modificar dificultad: Media

Seleccionar configuración por defecto Volver al menú

Por otra parte, la pantalla permite: volver a la configuración por defecto del nivel si se requiere, seleccionar y modificar otra dificultad, o escoger otra categoría de dataset.

Anexo 2: guía para el desarrollador

El presente software, desarrollado en Python 3.10.4, fue desarrollado por Julia Caro, María Emilia Romero y Francisco Venegas Naboulet, para la asignatura “Seminario de Lenguajes – Opción Python”

El directorio se encuentra orientado de la siguiente manera:

- common: contiene las funciones comunes, que son transversales al proyecto
- data: comprende los archivos generados para el procesamiento de datos dentro del juego
- helpers: incluye los archivos con lógica particular para una o dos pantallas específicas
- static: contiene aquellos archivos que son estáticos al proyecto
- windows: en esta carpeta se incluyen los archivos en los que se dividen las pantallas del juego
- figurace.py: es el archivo main del proyecto

Adicionalmente, para el presente desarrollo se han utilizado las librerías Pandas y PySimpleGUI. Recomendamos que, la instalación y uso de las mismas dentro de este proyecto, sean las utilizadas durante nuestro desarrollo. Por ello, además de la estructura de directorios descripta, encontrará un archivo “requirements.txt”, el cual, ejecutando el comando “pip install requirements.txt” le instalará en su dispositivo las versiones utilizadas, y, en caso de preferir una instalación manual, pueda revisar las versiones allí.

Una vez mencionado esto, es fundamental el entendimiento de la lógica del módulo hacer_ventana.py, ya que es lo que permite el intercambio entre pantallas, así como la interacción con el usuario.

Podemos encontrar este módulo dentro del directorio “common”. Allí, podemos ver que el mismo se encuentra dividido en tres funciones. La primera, “default_update”, que permite evitar errores cuando no se cuenta con código particular dentro de cierto procesamiento. En segundo lugar, tenemos la función “crear_ventana”, la cual será la encargada de crear las distintas ventanas con los atributos que se le indiquen por parámetros, y creará un bucle que permita la interacción con la misma. Por último, contamos con un “pasar_ventana” el cual permitirá la correcta transferencia entre pantallas, “escondiendo” la pantalla previa y “abriendo” la siguiente.

Adicionalmente, los módulos dentro de la carpeta “windows” contienen una estructura lógica que permite brindarle un criterio uniforme al programa y un mayor entendimiento al desarrollador. Dentro de esta carpeta, se podrá ver todo lo que lleva cada una de las pantallas que integra el juego, con su respectivo procesamiento e implementación de la parte gráfica.

Por otra parte, dentro de las carpetas “common” y “helpers” se pueden encontrar funciones definidas durante el desarrollo, con sus respectivos *docstrings* para el entendimiento de su funcionamiento. Las mismas pueden ser reutilizadas en el caso que se quiera ampliar el funcionamiento de cualquiera de las pantallas del presente juego.