Paradigmas de Programación

Práctica II - Curso 2020/21

Deslizator (II)

1. Definición y requisitos

Esta práctica es una continuación de la práctica anterior en la que vamos a aprovechar las capacidades de las Interfaces Gráficas de Usuario (GUI) para crear una aplicación orientada a entorno de ventanas que implemente el juego descrito en la primera práctica. Las reglas del juego son las mismas, salvo que ahora se puede establecer un número de filas distinto a 12 y el número de colores de las piezas es de 3.

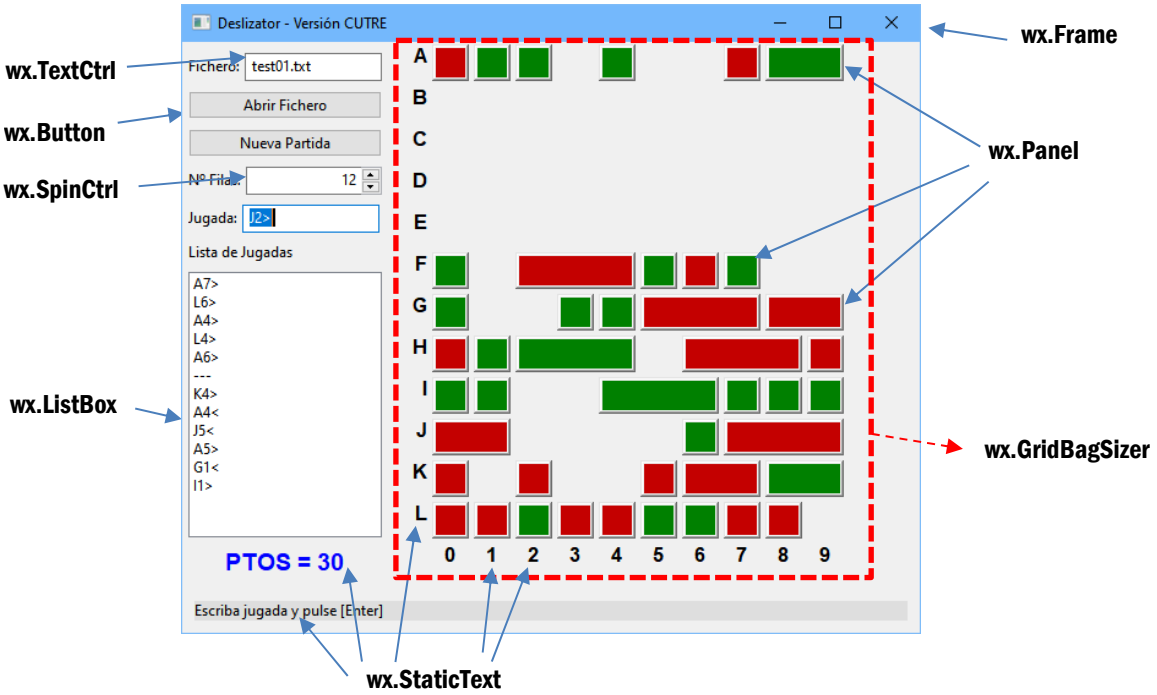
La librería de Python que se va a utilizar para implementar la interfaz gráfica es **wxPython** (https://wxpython.org/). Es posible apoyarse en otras herramientas (como **wxGlade**) para el diseño de la interfaz, pero el código debe estar basado exclusivamente es esa librería.

Los requisitos de la aplicación se van a clasificar en **imprescindibles** y **opcionales**. No se evaluarán prácticas que no incorporen los requisitos imprescindibles. Una práctica perfecta que cumpla únicamente los requisitos imprescindible tendría 5 puntos (sobre 10). Si cumple además todos los requisitos opcionales puede llegar a calificarse con 12 puntos sobre 10.

Requisitos imprescindibles

- Que muestre **gráficamente** el estado del tablero en todo momento.
- Que disponga de controles para que el usuario pueda indicar la ruta del fichero de líneas de entrada (definido exactamente igual que en la práctica anterior) y lleve a cabo su lectura. **Atención**: Hasta que no se haya leído un fichero no debe ser posible empezar a jugar.
- Que disponga de controles para poder cambiar el número de filas del tablero (si el cambio se produce en medio de una partida se reinicia el juego)
- Que disponga de controles para poder iniciar, en cualquier momento, una nueva partida
- Que muestre una lista de las jugadas realizadas hasta el momento en la partida actual
- Que muestre la puntuación obtenida hasta el momento
- Que disponga de controles para que el usuario pueda indicar la jugada a realizar

En la página siguiente se muestra un ejemplo del aspecto que podría tener una aplicación que solo cumple los **requisitos imprescindibles**, fijaros en que el usuario tiene que indicar la jugada escribiendo su código en un cuadro de texto, en vez de realizarla mediante pulsaciones de ratón. Se han etiquetado alguno de los controles y *sizers* utilizados para que sirvan de guía (por supuesto tenéis libertad para escoger el aspecto que queráis para vuestra aplicación siempre que sea compatible con los requisitos y con el buen gusto):



Se puede apreciar que este es un enfoque de mínimo esfuerzo y la aplicación resultante no aprovecha las posibilidades de la interfaz gráfica. Si se sigue este enfoque la puntuación máxima que se puede obtener es de 5 puntos.

Esa puntuación base se ira incrementando a medida que se implementen los siguientes requisitos opcionales (no es necesario implementarlos todos, se contabilizan individualmente):

Requisitos opcionales

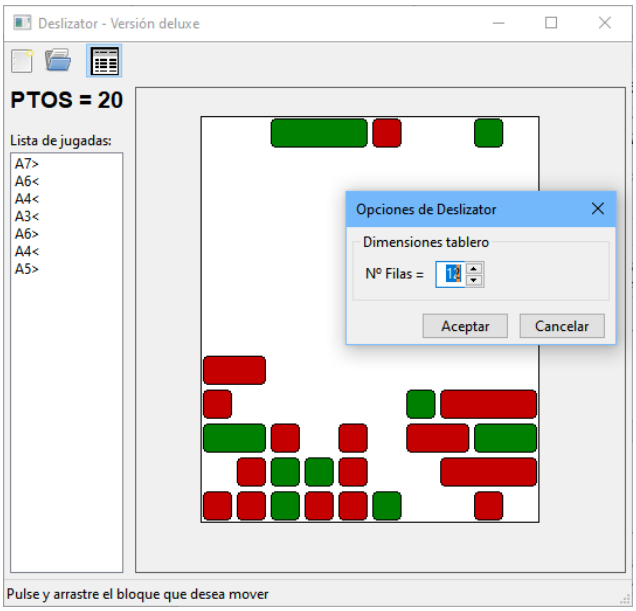
- **Indicación de jugada mediante ratón** (hasta **2.5 puntos**): El movimiento de los bloques se debe indicar pulsando sobre el bloque que se desea mover y *arrastrando* con el ratón hacia la izquierda o la derecha para indicar el sentido. Si se pulsa sobre una zona del tablero donde no hay bloques se entenderá que no se desea mover ninguno (el equivalente a la jugada "---" en modo texto).
- **Animación** de los movimientos y caídas de los bloques (hasta **2.5 puntos**).
- Utilización de **diálogos** (ya sean estándar y/o creados por vosotros) para los apartados de apertura de fichero y cambio del número de líneas (hasta **1 punto**).
- **Adaptación** inmediata del tamaño de los bloques y del tablero cuando se cambia de tamaño la ventana (hasta **1 punto**).

En general existen dos alternativas principales para representar el contenido del tablero:

- Usar controles estándar (paneles, imágenes, botones, etc.) para representar los bloques. La posición de estos controles se puede realizar mediante un *sizer* (se recomienda *GridBagSizer*) o bien no utilizar *sizers* y posicionarles de forma absoluta mediante llamadas al método *SetPosition* (y *SetSize*).
- Usar un panel y **dibujar** el contenido del tablero en él (ver la clase **wx.DC**)

En el mundo real prácticamente nunca se utilizaría la primera alternativa, pero en esta práctica podéis usar el método que queráis (cuentan igual). El segundo método es más sencillo de implementar (y se adapta mejor al proceso de animación) pero requiere un poco más de preparación previa para conocer la forma en que se trabaja con la clase DC.

A continuación se muestran dos imágenes de aplicaciones que implementan todos los requisitos opcionales, cada una de ellas implementa una alternativa distinta para representar el tablero:



2. Presentación y Evaluación de la práctica

La práctica se realizará **individualmente** y su evaluación se divide en dos etapas:

1. Presentación electrónica de los ficheros que componen la práctica: Ficheros ***.py** y todos aquellos (imágenes, etc.) que sean necesarios para su ejecución. Se habilitará en el Aula Virtual de la E.T.S. Informática (www.inf.uva.es -> menú Aula Virtual) una tarea de subida de ficheros cuya fecha límite será el **domingo 16 de mayo a las 23:59.** Al principio de todos los ficheros debe aparecer un comentario con el nombre de quien la han desarrollado.
2. Evaluación **presencial**, en laboratorio, ante el profesor. Se realizará en el lugar, día y hora correspondiente al horario de prácticas del subgrupo al que pertenezca durante la semana del **17 al 20 de mayo**.

En la evaluación de la práctica se tendrá en cuenta, entre otros, los siguientes factores:

- Autoría y participación en la misma.
- La correcta resolución del problema, así como la modularidad, documentación y robustez de la solución presentada. El uso de las técnicas impartidas hasta ese momento en la asignatura tiene una influencia positiva en la evaluación.