**Tarea 1: Revisar Gloss Familiarízate con la biblioteca Gloss para el desarrollo de gráficos 2D y aplicaciones interactivas en Haskell. Debes estudiar los conceptos fundamentales incluyendo: tipos básicos (Picture, Color, Display), transformaciones geométricas (Translate, Rotate, Scale), los tres modos de visualización (display, animate, play), y el sistema de manejo de eventos de teclado y ratónV**

Gloss es una biblioteca de Haskell para crear gráficos 2D y animaciones.

**Tipos básicos:**

-Picture: Representa una imagen o figura 2D.

| **Constructor** | **Parámetros** | **Descripción** | **Ejemplo de uso** | **Resultado visual** |
| --- | --- | --- | --- | --- |
| Blank | — | Imagen vacía (no dibuja nada). | Blank | No se muestra nada. |
| Polygon | [(Float, Float)] | Polígono definido por sus vértices. | Polygon [(0,0),(50,100),(100,0)] | Triángulo. |
| Line | [(Float, Float)] | Línea formada por varios puntos unidos. | Line [(0,0),(100,100)] | Línea diagonal. |
| Circle | Float | Círculo relleno con radio dado. | Circle 80 | Círculo de radio 80. |
| ThickCircle | Float (radio), Float (grosor) | Círculo hueco o anillo. | ThickCircle 60 10 | Anillo de radio 60, grosor 10. |
| Arc | Float (inicio), Float (fin), Float (radio) | Arco entre dos ángulos (grados). | Arc 0 180 80 | Semicírculo. |
| Text | String | Dibuja texto en pantalla. | Text "Hola Gloss" | Muestra texto. |

-Color:Define el color de una figura (Picture).

* Colores predefinidos: white, black, red, green, blue, yellow, orange, purple, grey, etc.
* Colores personalizados: makeColor r g b a, donde cada componente va de 0 a 1.

-Display**:** Indica cómoy dónde se mostrará la ventana.

| **Constructor** | **Ejemplo** | **Descripción** |
| --- | --- | --- |
| InWindow | InWindow "Título" (ancho, alto) (x, y) | Ventana normal. |
| FullScreen | FullScreen | Pantalla completa. |
| InPlace | (raro de usar) | Renderizado sin ventana. |

**Transformaciones geométricas y combinación de imágenes:**

Las transformaciones permiten modificar o combinar figuras ya existentes.

| **Función** | **Parámetros** | **Descripción** | **Ejemplo de uso** | **Efecto visual** |
| --- | --- | --- | --- | --- |
| Translate | Float (x), Float (y), Picture | Desplaza una figura horizontal y verticalmente. | Translate 100 50 (Circle 30) | Mueve el círculo 100 px a la derecha y 50 px hacia arriba. |
| Rotate | Float (grados), Picture | Rota la figura un ángulo (en grados, sentido antihorario) respecto al origen (0,0). | Rotate 45 (RectangleSolid 100 50) | Gira el rectángulo 45°. |
| Scale | Float (x), Float (y), Picture | Escala la figura en el eje X e Y. | Scale 2 0.5 (Circle 40) | Estira el círculo al doble de ancho y la mitad de alto. |
| Pictures | [Picture] | Combina varias figuras en una sola imagen. | Pictures [Circle 50, RectangleWire 100 100] | Pictures [Circle 50, RectangleWire 100 100] |

\*\* Las transformaciones se pueden combinar aplicándose de aplican de derecha a izquierda.

Ejemplo: figura1 = Translate 100 0 $ Rotate 45 $ Circle 30

**Modos de visualización**

-Display: crea una imagen estática.

-Animate: crea animaciones basadas en el tiempo. La función recibe el tiempo (en segundos) y devuelve un Picture distinto en cada frame.

-Simulate: permite mantener y actualizar un estado

**Play**

Permite crear aplicaciones interactivas, con eventos de teclado/ratón y actualizaciones del estado cada frame.

play

:: Display -- Tipo de ventana

-> Color -- Color de fondo

-> Int -- Número de frames por segundo (FPS)

-> world -- Estado inicial del juego

-> (world -> Picture) -- Función de dibujo (renderizado)

-> (Event -> world -> world) -- Función de manejo de eventos

-> (Float -> world -> world) -- Función de actualización del estado

-> IO ()

Un evento representa una acción que puede cambiar el estado del programa.

-EventKey: evento de teclado y ratón. Para describir el contenido de un EventKey, Gloss usa otros tipos:

-Char:Tecla de carácter alfanumérico.

-SpecialKey: Tecla especial del teclado.

-MouseButton: Botón del ratón.

Tarea 2: Desarrollar un juego simple con monigote Implementa un juego básico usando Gloss donde aparezca un personaje (monigote) que se pueda controlar mediante el teclado. El monigote debe poder desplazarse horizontalmente (izquierda/derecha) usando las teclas A/D o las flechas del teclado, y realizar un salto con física realista al presionar W o flecha arriba. El juego debe incluir: un tipo de datos para representar el estado del juego (posición, velocidad, dirección), una función de dibujo que represente gráficamente al personaje de forma lateral o frontal, un manejador de eventos que capture las pulsaciones del teclado, y una función de actualización que aplique física básica (gravedad, detección de suelo, límites de pantalla). El personaje debe animarse al caminar y voltear su orientación según la dirección del movimiento. Ver video de ejemplo en la carpeta de la tarea.

Tarea 3: Implementar Applicative y usar fmap Refactoriza los tipos de datos de tu juego HAUS para que implementen la clase de tipos Applicative (y por tanto también Functor). Identifica en tu código todas las oportunidades donde se puedan usar las funciones fmap (o su operador <$>), <\*>, y pure para hacer el código más declarativo y funcional. Esta tarea se centra en aplicar los conceptos de functores aplicativos que se impartirán en la próxima clase. Debes modificar las funciones existentes para aprovechar estas abstracciones cuando sea posible, por ejemplo al actualizar el estado del juego, transformar coordenadas, o aplicar funciones a valores dentro de contextos. La implementación debe respetar las leyes de Functor y Applicative. Nota Esta tarea requiere comprender los conceptos que se enseñarán en la próxima clase sobre Functores y Applicatives. Se recomienda revisar material previo sobre estas type classes si deseas adelantar trabajo. ¿cual es más fácil?