# Documentación
# Tarea 2 - ELO329: Simulador Gráfico de la evolución de una pandemia. (Versión 2.1)

**Escrito por:** **Miguel Andrade**

**Manuel Cruces**

**Kevin Pizarro**

**Pablo Troncoso**

**Junio 13, 2021**

## Explicación breve de la solución:

Se nos solicita implementar una interfaz gráfica de usuario (GUI) para la evolución de una pandemia. La GUI permite recibir parámetros iniciales a través de un archivo de configuración en formato txt y luego pueden ser modificadas dentro de una ventana del simulador llamada **settings**. Por otro lado, la interfaz permite iniciar, pausar, acelerar y ralentizar la simulación a gusto. Por último, para poder interpretar y visualizar fácilmente la evolución de la pandemia se provee una sección para un gráfico de áreas apiladas, el cual se actualiza en tiempo “real”. Para esto (utilizando el código de ayuda), se tienen 8 clases:

- Comuna.java
- ComunaView.java
- Pedestrian.java
- PedestrianView.java
- Simulator.java
- SimulatorConfig.java
- SimulatorMenuBar.java
- StageX.java

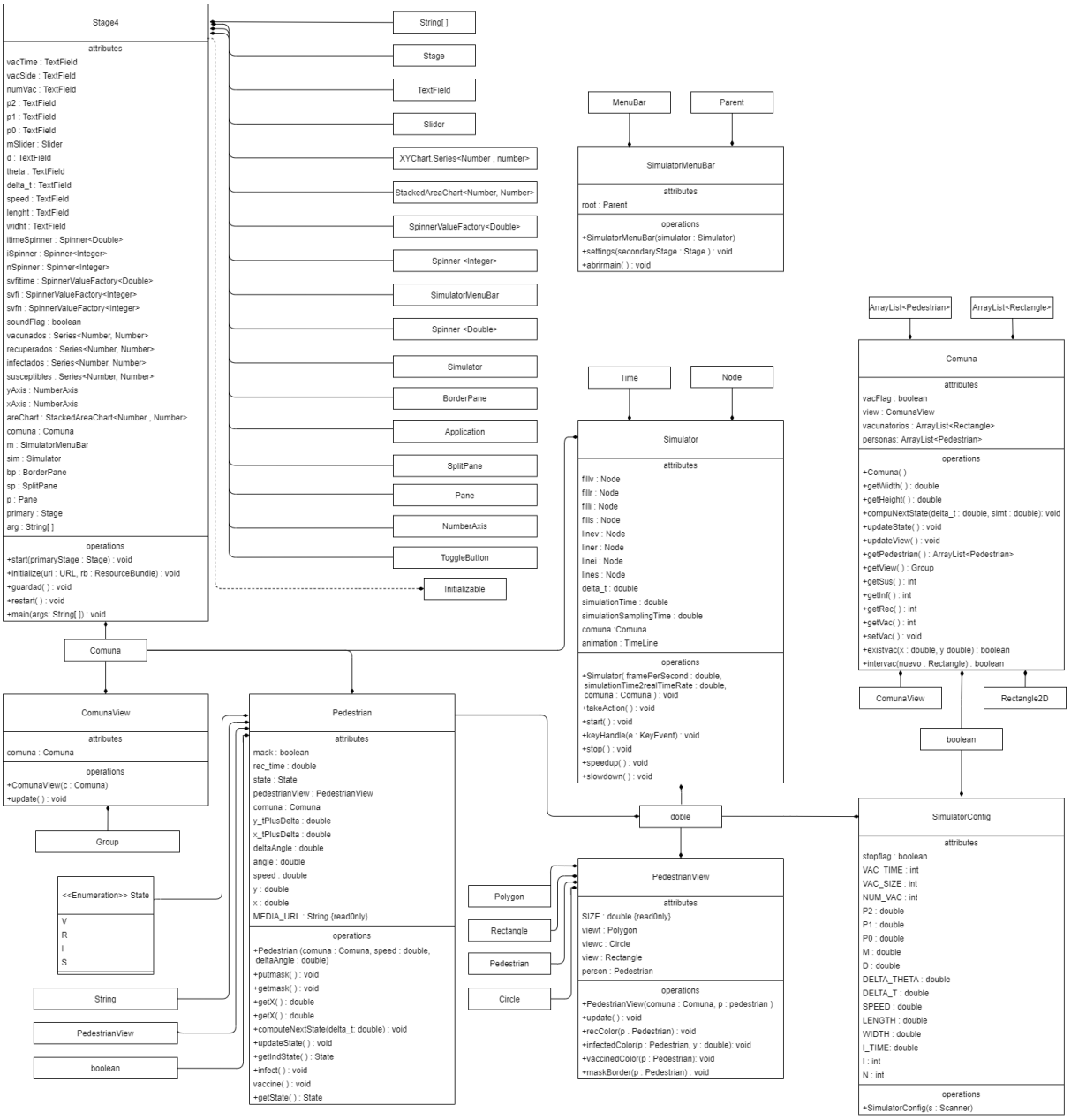
Stage X es la clase donde se encuentra el main y cuyo trabajo es inicializar simulador y comuna, así como recibir los parámetros del archivo de configuración (utilizando las librerías Scanner y Locale para leer el archivo, para este punto está la clase SimulatorConfig que se encarga de almacenar los valores leídos, IOException para evitar errores y File para trabajar en los archivos) los cuales se entregan al Simulator, que se encargará principalmente de la simulación en sí, es decir, el manejo de los tiempos de la simulación e invocación de los métodos para actualizar el estado de los individuos además de crear y manejar el gráfico de áreas apiladas. Esto lo logra trabajando con tiempos discretos, es decir, por cada instante de tiempo $\Delta t$ se “detiene” a calcular los cambios en los individuos para luego reportarlos y agregarlos a sus variables correspondientes, además de otorgar los valores necesarios a comuna para que esta última se encargue de la creación de los individuos y vacunatorios.

Para los menús dentro de la interfaz se encarga SimulatorMenuBar, que crea los menús “Controls” y “Settings”, con “Controls” se tienen los submenus “Start” y “Stop” que se encargan de comenzar y detener la simulación, “Settings” se encarga de mostrar otra ventana en donde se podrán modificar los datos entregados por el archivo de entrada.

En Comuna se inicializa el “espacio” y arreglo de los individuos que se mueven en este, llamando a las actualizaciones de estado de los mismos según lo indique el simulador para pasarlos a ComunaView. Luego se definen los vacunatorios y el método para ubicarlos en el espacio de la comuna. En Pedestrian se parametriza el comportamiento y estados de estos; si está infectado, vacunado, recuperado o es susceptible a infectarse. Luego se tienen los cálculos propios de su movimiento “aleatorio”, el cual está determinado por los parámetros entregados por el archivo de configuración, todos estos cálculos serán enviados a PedestrianView.

Para la parte visual utilizaremos principalmente ComunaView el cual actualizará todos los componentes al interior de la comuna, es decir, el arreglo de Pedestrian y los vacunatorios. Para ayudar a representar gráficamente también usaremos PedestrianView el cual transforma la información de Pedestrian, creando cuadrados para los estados susceptible y recuperado, círculos para los infectados, polígonos triangulares para los vacunados y otorganodole un borde negro a las figuras ya mencionadas si el individuo pose máscara.

Se adjunta el diagrama de clases UML.




**1: Diagrama de clases UML para Extra 1.**

## Dificultades encontradas:

1. Creación del menú settings.

**Solución:** Se realiza un archivo fxml creado mediante SceneBuilder, luego se le asigna el archivo StageX.java como controlador, en vez de SimulatorMenuBar.java y un Controller.java (controlador de prueba) como se realiza en primeras instancias.

2. Generación de formas para representar gráficamente al individuo.

**Solución:** Se crean todas las formas (rectangle, circle y polígono de 3 lados) paralelamente para el individuo, sin embargo se muestra sólo aquél correspondiente a su estado (vacunado, infectado u otro) mientras los demás quedan escondidos visualmente. Cabe mencionar que todas las formas están coordinadas para que el individuo quede en el centro de estas.

3. Resetear a tiempo 0 [s]

**Solución:** Se guardan todas las variables necesarias para iniciar todo desde 0, en el momento antes de iniciar nuevamente la simulación se ejecuta un Stop() para que no haya conflictos ni superposiciones de simulaciones anteriores.

4. Creación del Makefile

**Solución:** Se agregó una segunda variable llamada PATH2, en la que se coloca el directorio principal de la tarea, para modificar el classpath que utiliza java a la hora de correr los códigos, permitiendo correr el código del package desde dentro de la carpeta, como si se encontrara fuera de ella.

## Gráficos de Salida:

Para obtener los siguientes gráficos se simuló con la configuración dada por defecto en el repositorio de GitLab.

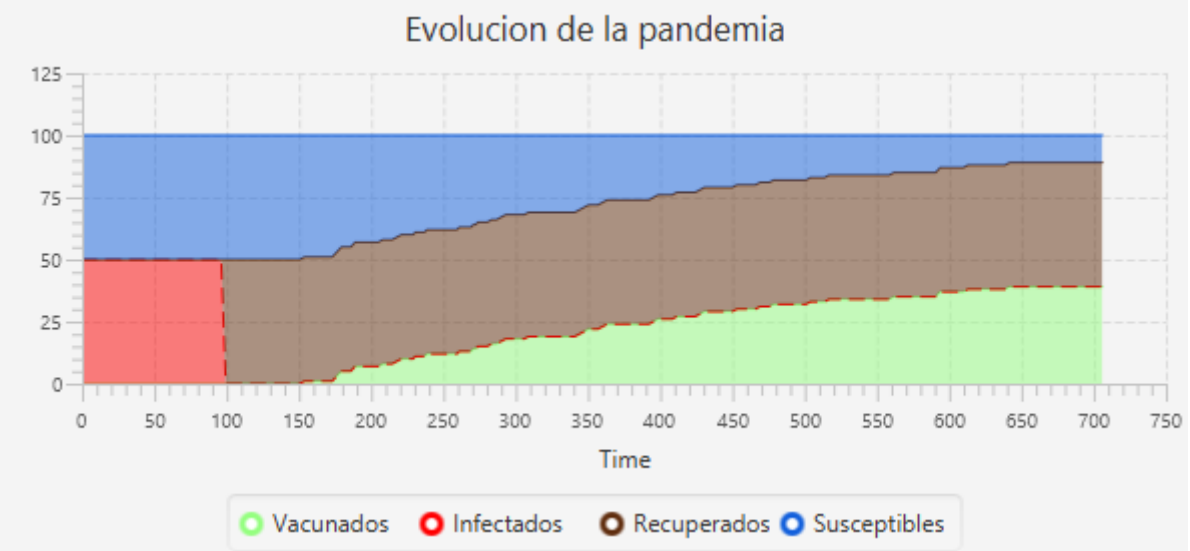



**Gráfico 1: Salida de Stage 4.**