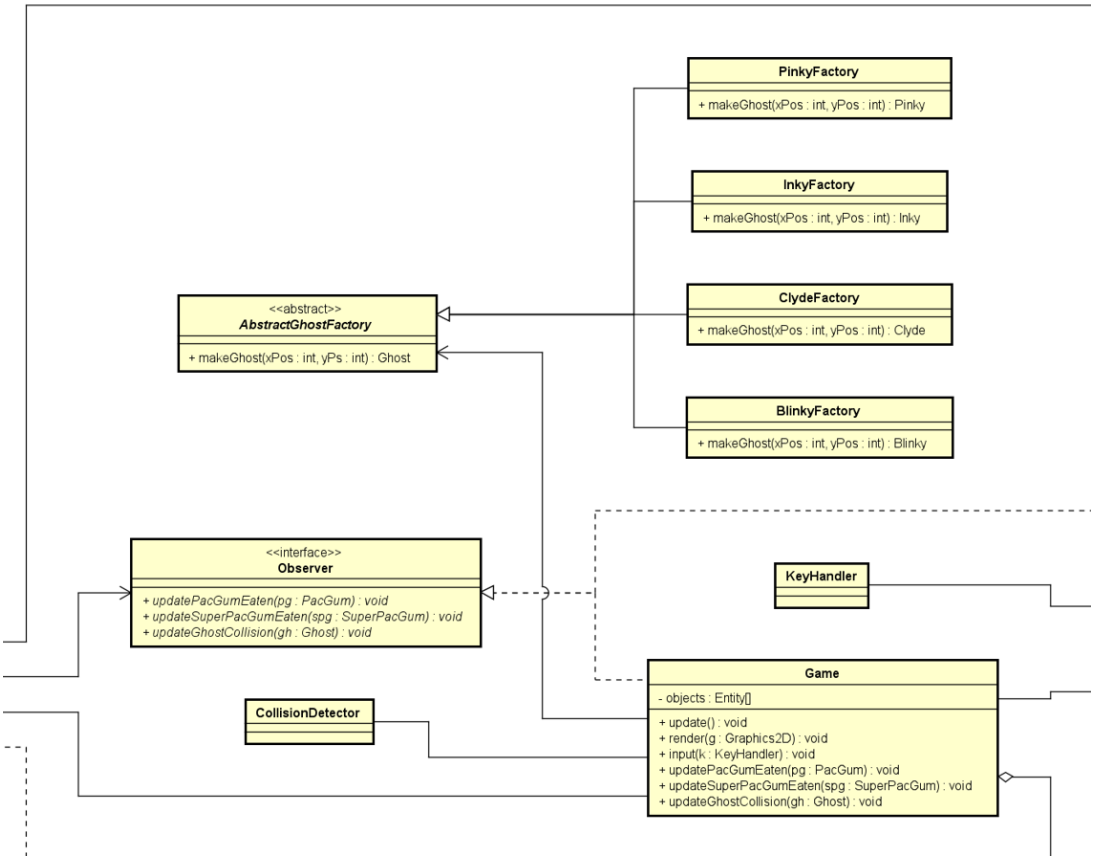
Taller 5

El proyecto que elegí es una implementación del juego Pac-Man, el cual está desarrollado completamente en java con librerías como java swing.

https://github.com/lucasvigier/pacman

Patrón Factory

En primer lugar, el patrón de diseño Factory es un patrón creacional que se utiliza para crear objetos de un tipo específico sin especificar explícitamente su clase concreta. Proporciona una interfaz para crear objetos, pero delega la responsabilidad de la creación a las subclases. En el contexto de los videojuegos como Pac-Man, el patrón Factory es especialmente útil porque permite la creación de objetos de juego de manera flexible y extensible.



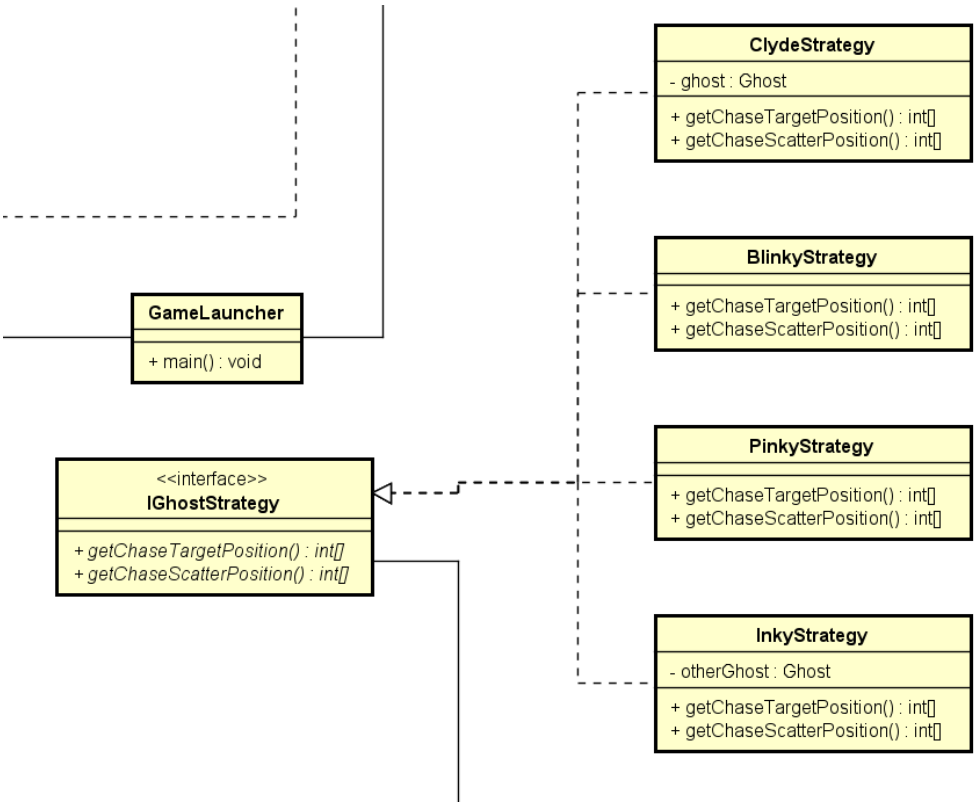
En este caso se puede ver como el juego tiene una clase “AbstractGhostFactory” la cual es la que implementa el patrón Factory al instanciar entidades de cada uno de los distintos fantasmas: pinky, inky, clyde, blinky. En este caso tiene un método que crea el fantasma con una posición en X y una posición en Y.

```java
package game.ghostFactory;

import game.entities.ghosts.*;

//Abstract Factory pour créer les différents fantômes concrets à partir de constructeurs différents
public abstract class AbstractGhostFactory {
    public abstract Ghost makeGhost(int xPos, int yPos);
}
```

El patrón factory le otorga flexibilidad en la creación de objetos ya que permite agregar nuevos tipos de elementos al juego Pac-Man sin modificar el código existente de igual modo evita duplicar código a la hora de instanciar objetos.

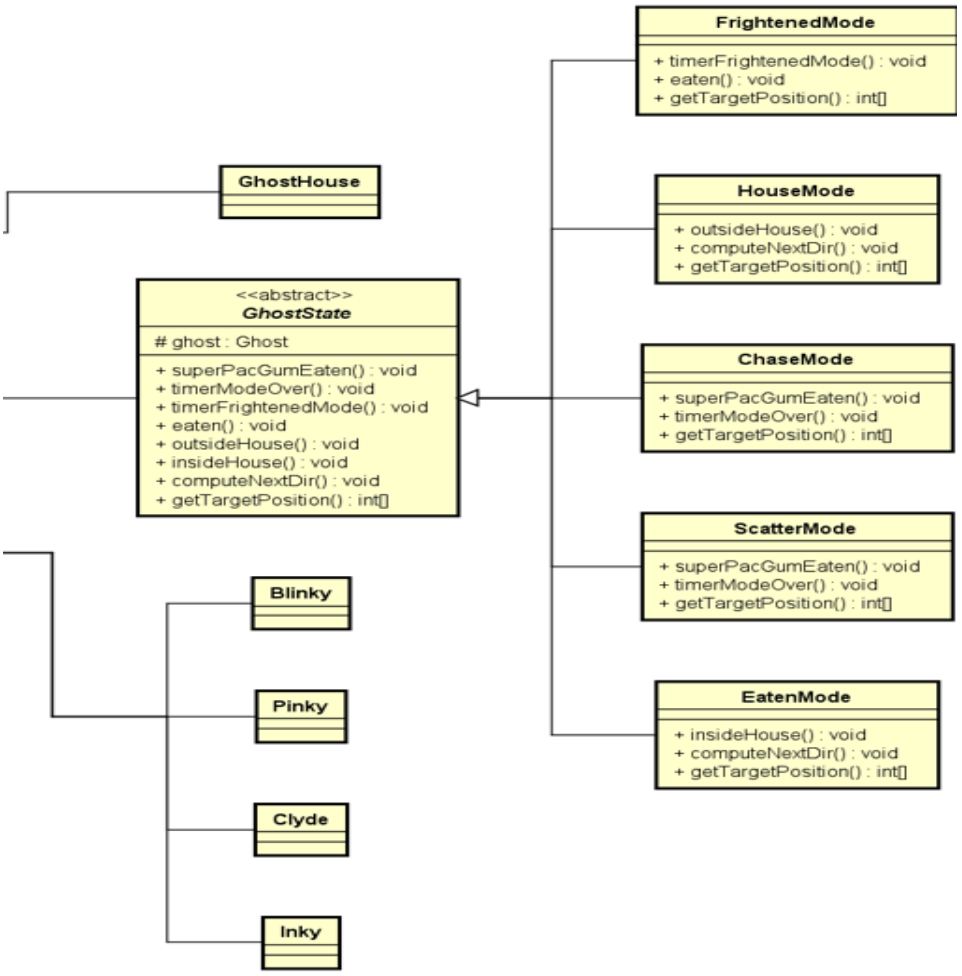
De igual modo este patrón simplifica mucho el código lo brinda un mejor entendimiento del código a la hora de realizar ajustes o al momento de que otra persona lea la implementación del proyecto.

Patrón Strategy



Por otra parte, el proyecto también implementa el patrón de estrategia el cual proporciona una forma de encapsular diferentes algoritmos o estrategias. En este caso es útil para para definir los comportamientos de los fantasmas, en este caso la acción de perseguir.

Patrón State



Por último, también implementa el patron State, el cual permite que un objeto cambie su comportamiento en función de su estado. En este caso se encuentra la clase GhostState y las clases ChaseMode, ScatterMode, EatenMode, FrighteneMode Y House Mode la cual establece los distintos comportamientos de Pac-Man y de los fantasmas.

Este proyecto se podría haber realizado con una aproximación diferente sin usar patrones, pero esto dificultaría mucho el desarrollo del mismo, ya que los patrones ayudan a simplificar el código como lo puede ser en el caso de Factory el cual si no se implementa se tendrían que generar las distintas entidades instanciándolas una por una, de igual modo con State el cual se tendría que definir completamente cada comportamiento por aparte y lo mismo pasaría con Strategy lo cual le quitaría dinamismo al juego.