Controlador es estatico (o cada clase tiene una referencia al controlador)

Cada clase se encarga de su info, cuando se le llama una función, por lo general generaran un Evento, por ejemplo cuando el cajero le pasa un reloj al relojero

Cuando una clase hace algo, por si genera un evento, lo que hace es determinarle la hora de fin, y agregarlo a la lista de futuros eventos del controlador, por ej, cuando el relojero empieza a trabajar en un reloj genera un evento “fin reparación”

Los eventos los guarda el controlador en una lista, y los guarda ordenados

Hay eventos de inicio y fin de simulación

El controlador en cada evento se encarga de preguntarle a los objetos (cajero, relojero, etc) que datos tienen, para cargar la tabla. Esos datos se guardan en una tupla de strings (o algo asi) si son null, se guarda cadena vacia

A la lista de próximos eventos, hay que agregar los eventos en orden, y después obtenerlos con un pop (que es un get junto con el borrado)

No hay un evento distinto para cada necesidad del cliente, lo que quiere el cliente se determina en el momento que llega (primero e determina si quiere comprar, encargar, o retirar, y si quiere retirar, hay que elegir a un cliente random de los que están como pendientes, tal vez sea por medio de los relojes o de los clientes, y hay que ver cual es el estado de su reloj, y si no esta listo, se produce un “error”, no exactamente un error, sino que ocurrio algo que no tendría que haber pasado)

La clase cliente capas que no es necesaria…

|  |  |
| --- | --- |
| **Eventos** |  |
| Fin atención cliente |  |
| Fin reparación |  |
| Llegada cliente |  |
| ~~Llega Cliente (compra)~~ |  |
| ~~Llega Cliente (encargo)~~ |  |
| ~~Llega Cliente (retiro)~~ | ~~No siempre es posible, depende de si hay relojes en reparacion o listos para retirar~~ |

|  |
| --- |
| Reloj |
| Estado: enReparacion, esperandoReparacion, reparado |
| ComenzarReparacion(): evento (fin reparacion)  TerminarReparacion(): void  ObtenerTiempoReparacion(): float (tiempo) |

|  |
| --- |
| Relojero |
| Estado: reparando, libre |
| Método q hace q el relojero llame a los métodos en el reloj (tal vez reparar(Reloj), o comenzar y terminar de reparar) |