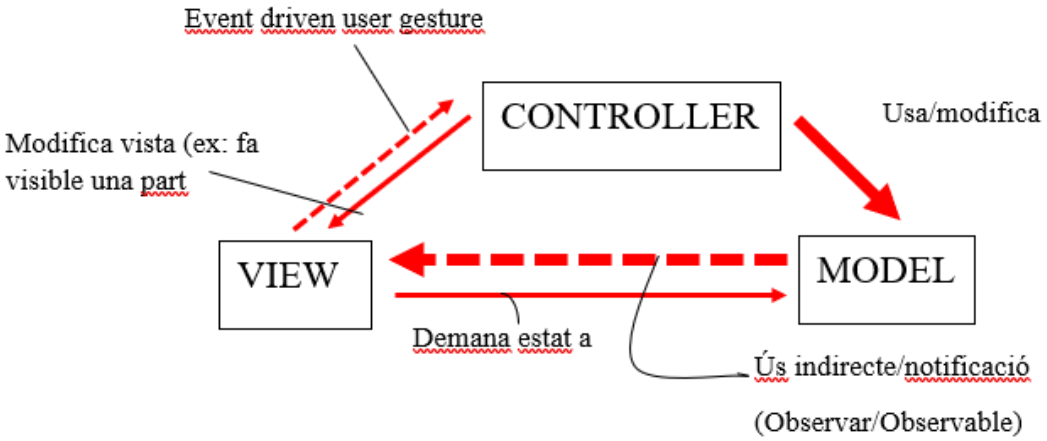
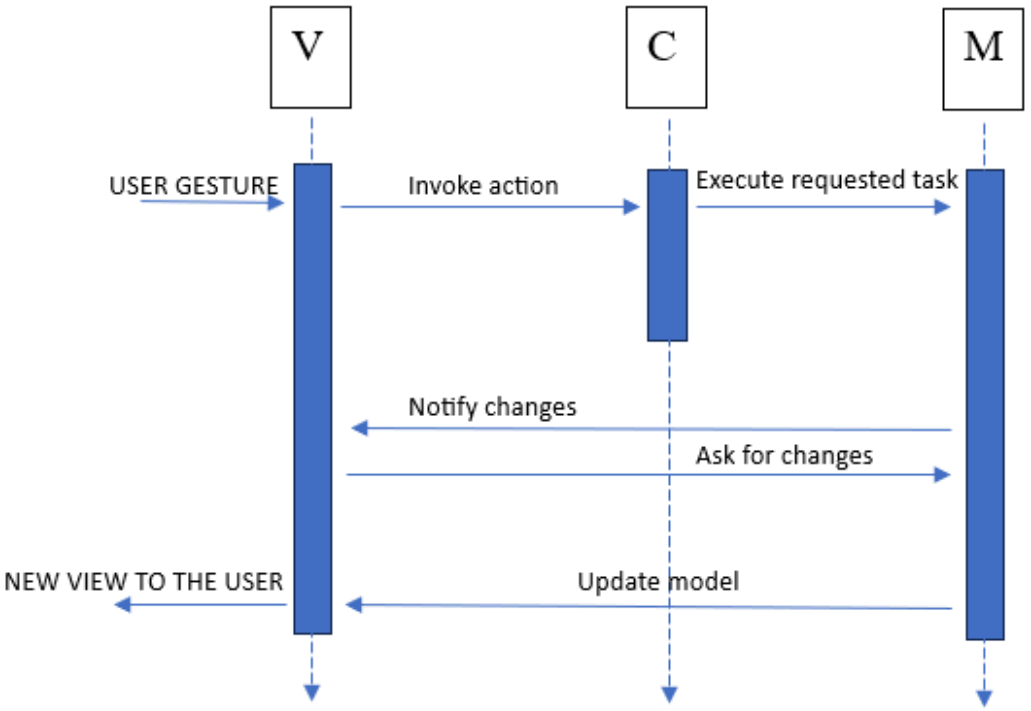
MVC

- Interessa abstreure la presententació (vista) del model de dades.
- Idealment canvis en el model comporten automàticament actualització de la presentació
- **Model:** L'estat de l'aplicació és el que canvia
- **View:** Presentació de l'aplicació. Un model pot tenir-ne més d'una
- **Controller:** Gestor d'esdeveniments d'Input. Invoca canvis d'estat del model



- Implementat per primer cop el 1979 a Smalltalk-80 (gran influencia en el disseny de GUIS posteriors)



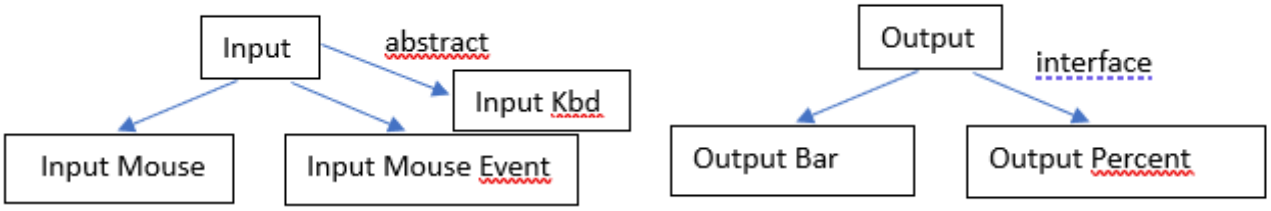
MVC Patró Observer/Observable

MVC desacopla els tres elements: podem canviar la vista.
El model modifica la vista dels seus canvis, actualitzant la seva presentació. El model no te perque coneixer els detalls de la vista/vistes [Line (M), Edita.. (C), Console (V)].

Patró Observer/Observavbe: Els observers (vistes) es registren amb l'Observavle (model9¡). El model informa les vistes amb notifyObservers quan el seu estat canvia. Internament manté una llista d'observadors registrats notifyObservers invoca update de cada vista per actualitzar-la.

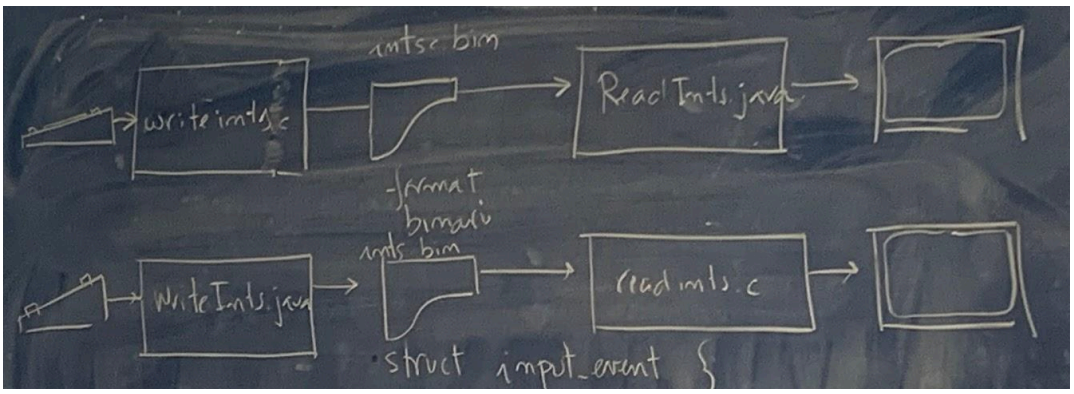
```
class Observable
{
        void addObserver(Observer o)
        void notifyObservers();         //pull
        void notifyObservers(Object arg);     //push
        void setChanged();
}

interface Observer
{
        void update(Observable o, Object arg);
}
```

(Observació personal): el patro MVC es fixa en la presentació, però no te en compte l'input.
Patró handle abstracte:



Programar el ratolí obrint /dev/input/event5 com a arxiu de Java (open)
En UNIX "everything is a file"
Llegeixo valors que són structs C

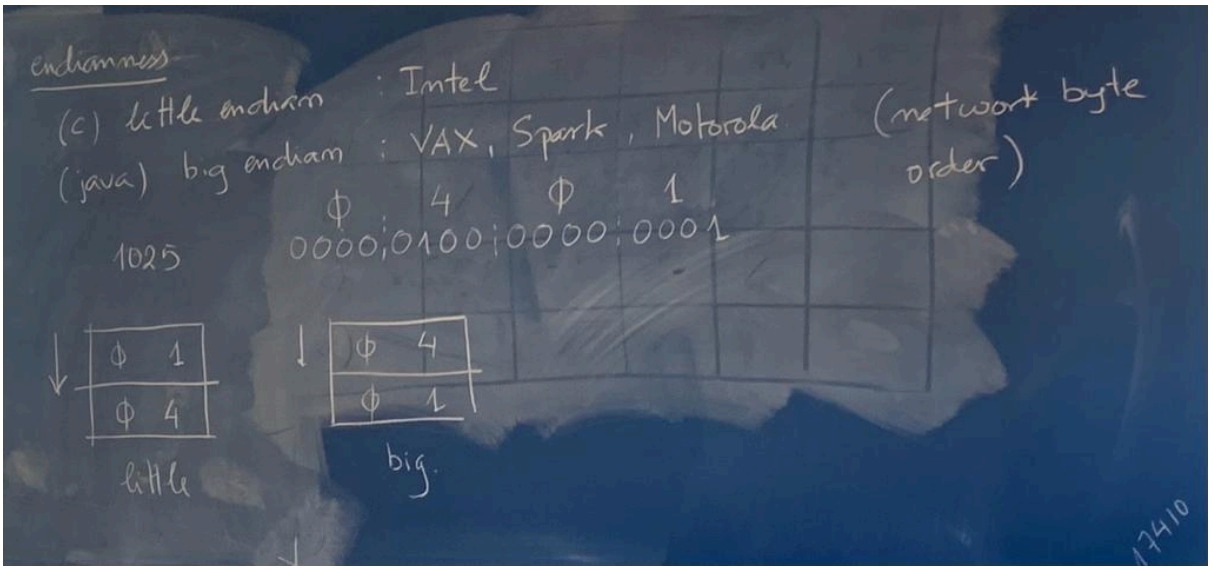
```
struct input_event{
        struct timeval time;
        unsigned short type code;
        unsigned int value;
}
```



Java	C	fd (file descriptor)
System.in	stdin	0 (STDIN_FILENO)
System.out	stdout	1 (STDOUT_FILENO)
System.err	stderr	2 (STDERR_FILENO)

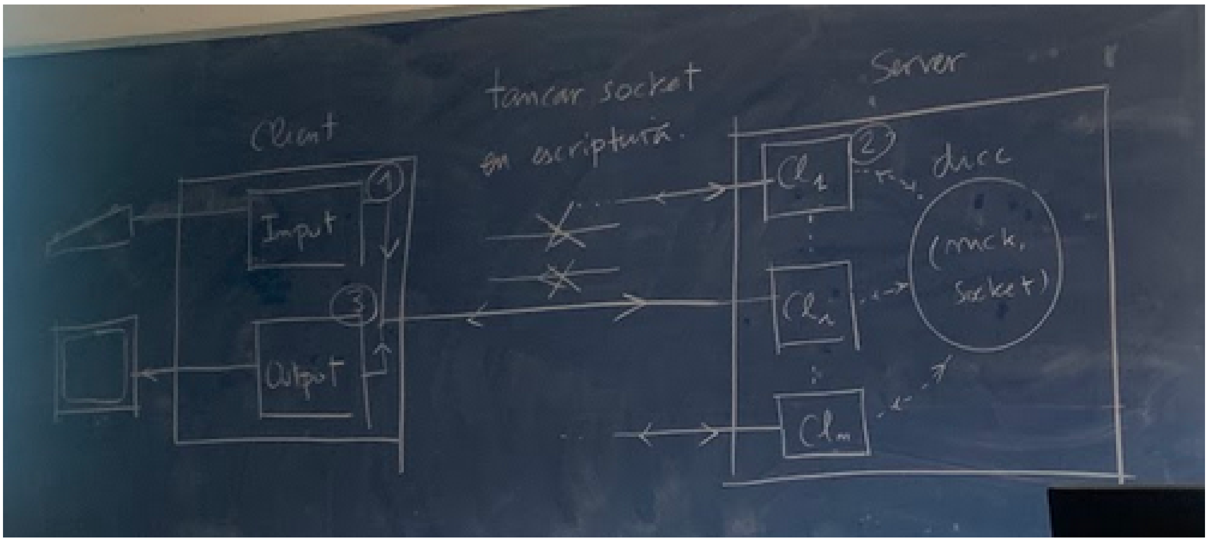
cc -o writeints.com writeints.c
consola:
./writeints.com > intsc.bin

endianess:
(c) little endian
(java) big endian



Programar amb el mètode match de parsing
1- caracter a caracter guardant un prefix
2- amb els mètodes de Reader

PRÀCTICA 2



El monitor que programàvem a AST ja està fet a ReentrantWriteLock.

PRÀCTICA 3

Swing	Android
Event Dispatch Thread (EDT) no s'ha de bloquejar	
Init thread (main) l'EDT - - - - - - - - - - → threads auxiliars (worker threads) →SwingWorker	UI thread (executa Activities) Async Task
El codi gràfic no està preparat per ser executat per varis threads (not thread-safe, not reentrant)	
Hi ha un mecanisme de delagació de codi del thread auxiliar al EDT. Es passa un objecte Runnable SwingUtilities: - invokeLater(Runnable) AsyncTask (asíncron) - invokeAndWait(Runnable) (síncron)	runOnUiThread

widget->JTextField widget.add.AdctionListener(ActionListener) -> actionPerformed Observable.addObserver(Observer) -> update	view-> EditText view.setOnEditor ActionListener() OnActionListener( )
Als widgets, views (elements gràfics) se'ls pot associar un model new JList(users->ListModel)	ListView.setAdapter(list->ListAdapter)

l'EDT té una cua d'events, però la concurrencia d'aquesta ja ve feta, no s'ha de programar
Biblioteca Gimnp ToolKit (GTK)
El codi GUI està ple de variants de MVC!

```
JList usrArea = new JList(users);
…
users.addElement("pepe");
…
users.removeElement("juan");
```