Conception d'un système interactif

1. Liste des événements
2. Liste des actions
3. Matrice d'états/événements
4. Event-handlers

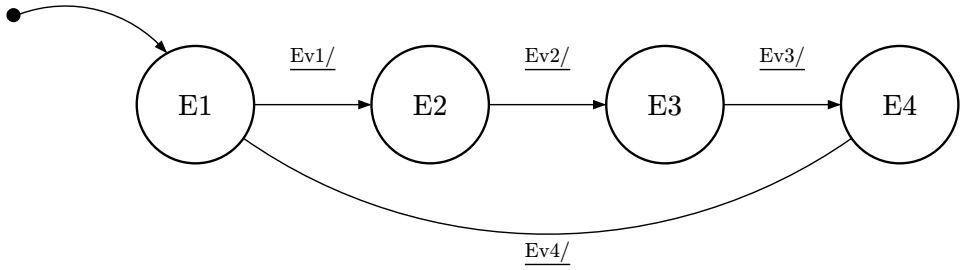
Exemple de cours

1) Système simple :

- $E = \{E_1, E_2, E_3, E_4\}, s_0 = E_1$
- $\mathrm{Ev} = \{\mathrm{Ev}_1, \mathrm{Ev}_2\}$,
- $A = \{A_1, A_2, A_3, A_4\}$,

- $f : E \times \mathrm{Ev} \to A$
  $f(E_1, \mathrm{Ev}_1) = A_1$
  $f(E_2, \mathrm{Ev}_1) = f(E_3, \mathrm{Ev}_1) = A_2$
  $f(E_4, \mathrm{Ev}_2) = f(E_3, \mathrm{Ev}_2) = A_3$
  $f(E_2, \mathrm{Ev}_2) = A_4$

- $g : E \times \mathrm{Ev} \to E$,
  $g(E_1, \mathrm{Ev}_1) = g(E_3, \mathrm{Ev}_2) = E_2$
  $g(E_2, \mathrm{Ev}_1) = g(E_4, \mathrm{Ev}_2) = E_3$
  $g(E_3, \mathrm{Ev}_1) = E_4$
  $g(E_2, \mathrm{Ev}_2) = E_1$



	Ev1	Ev2	Ev3	Ev4
E1	État "E1"	Interdit	Interdit	Interdit
E2	Interdit	État "E3"	Interdit	Interdit
E3	Interdit	Interdit	État "E4"	Interdit
E4	Interdit	Interdit	Interdit	État "E1"

Exercices de TP

Quatre boutons (*Reprise de l'exercice du cours*)

- **Événements :** (*4 boutons*) `cb1`, `cb2`, `cb3`, `cb4`
- **Actions :** $\emptyset$
- **Automate :**

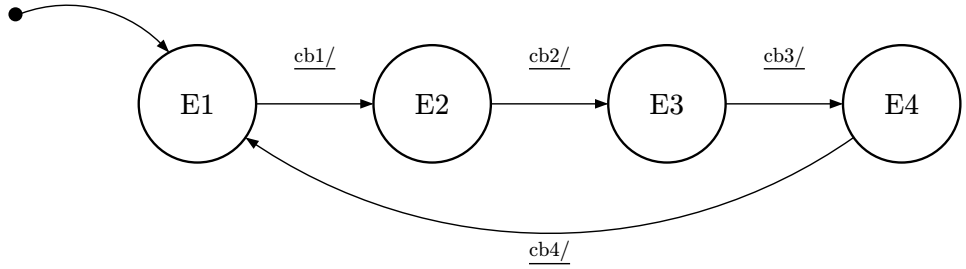


Figure 2: Automate 4 boutons

	`cb1`	`cb2`	`cb3`	`cb4`
E1	État "E2"	Interdit	Interdit	Interdit
E2	Interdit	État "E3"	Interdit	Interdit
E3	Interdit	Interdit	État "E4"	Interdit
E4	Interdit	Interdit	Interdit	État "E1"

Table 2: Matrice 4 boutons

- **Code :** `FourButtons.java`

Figure 3: Capture d'écran de l'app FourButtons

Quatre boutons deux à deux

- **Événements :** (*3 boutons*) `cb1`, `cb2`, `cb3`, `cb4`
- **Actions :** ∅
- **Automate :**

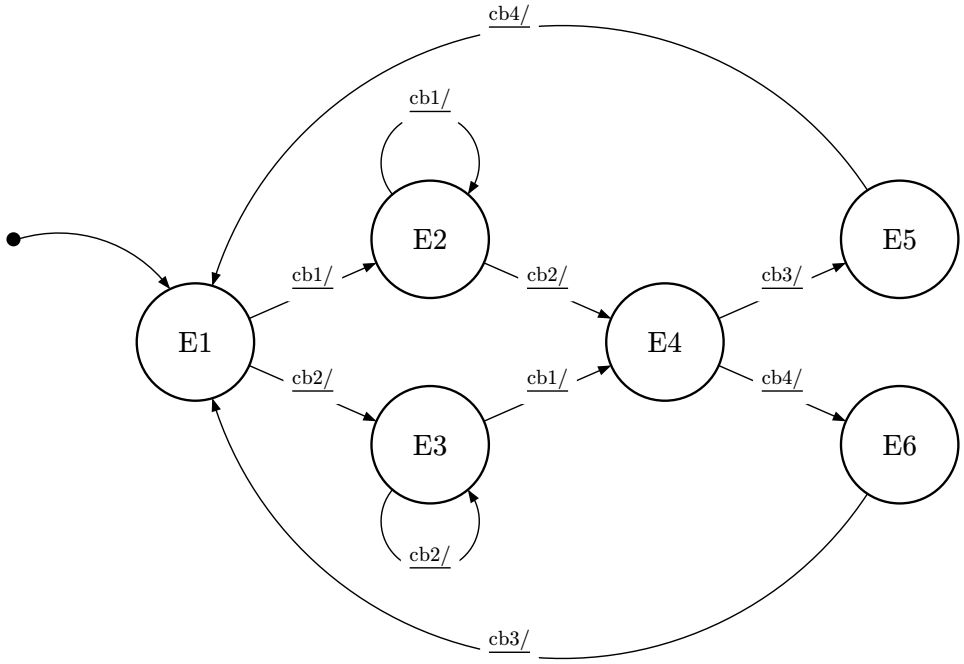


Figure 4: Automate 4 boutons deux à deux

	`cb1`	`cb2`	`cb3`	`cb4`
E1	État “E2”	État “E3”	Interdit	Interdit
E2	État “E2”	État “E4”	Interdit	Interdit
E3	État “E4”	État “E3”	Interdit	Interdit
E4	Interdit	Interdit	État “E5”	État “E6”
E5	Interdit	Interdit	État “E5”	État “E1”
E6	Interdit	Interdit	État “E1”	État “E6”

Table 3: Matrice 4 boutons deux à deux

- **Code :** `FourButtons2.java`

Figure 5: Capture d’écran de l’app FourButtons 2

Compteur

- **Événements :**
  - (*3 boutons*) `cbStart`, `cbStop`, `cb+1`
- **Actions :** `init()`, `A1`, `A2`
- **Automate :**

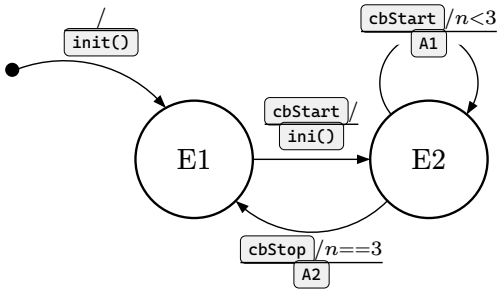


Figure 6: Automate compteur

	`cbStart`	`cbStop`	`cb+1`
E1	`init()` État "E2"	Interdit	Interdit
E2	Interdit	`A2` État "E4"	`A1` État "E2"

Table 4: Matrice compteur

- **Code :** `Counter.java`

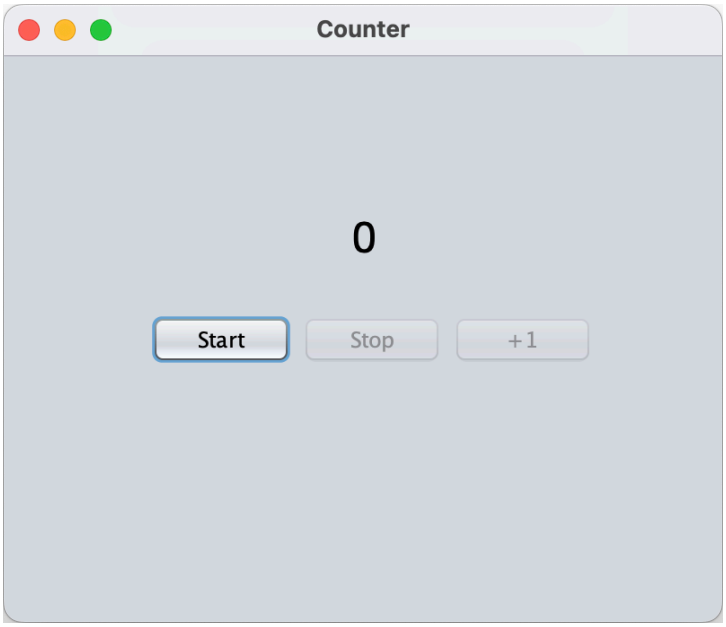


Figure 7: Capture d'écran de l'app Counter

Compteur simple (avec timer)

- **Événements :**
  - (*3 boutons*) `cbStart`, `cbStop`
  - (*1 Timer*) `Ttick`
- **Actions :** `init()`, `A1`, `A2`
- **Automate :**

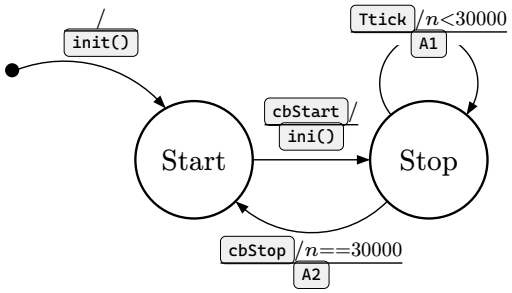


Figure 8: Automate compteur

	`cbStart`	`cbStop`	`Ttick`
Start	`init()` État “Stop”	Interdit	Interdit
Stop	Interdit	`A2` État “Start”	`A1` État “Stop”

Table 5: Matrice compteur

- **Code :** `CounterSimple.java`

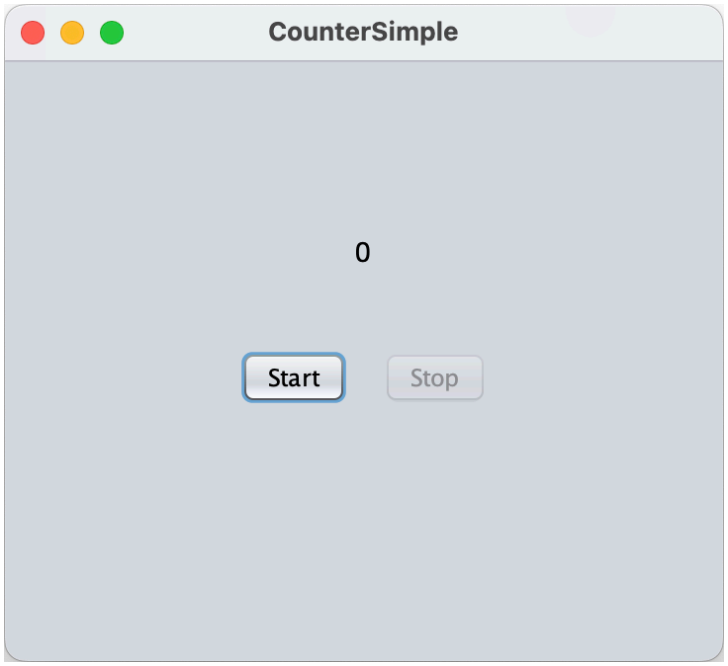


Figure 9: Capture d’écran de l’app Counter Simple

Compteur avancé

- **Événements :**
  - (*3 boutons*) `cbStart`, `cbRecule`, `cbAvance`, `cbStop`
  - (*2 Timer*) `timerAvanceActive`, `timerReculeActive`
- **Actions :** `init()`, `A1`, `A2`
- **Automate :**

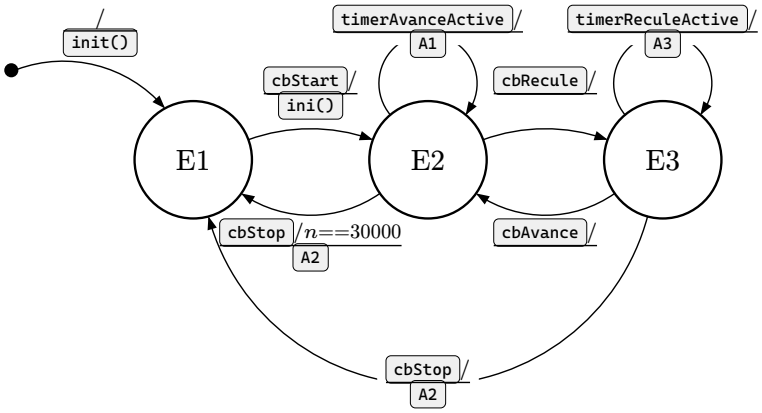


Figure 10: Automate compteur avancé

	`cbStart`	`cbStop`	`cbRecule`	`cbAvance`	`timerReculeActive`	`timerAvanceActive`
START E1	`init()` État "E2"	Interdit	Interdit	Interdit	Interdit	Interdit
RECULE_STOP E2	Interdit	`A2` État "E1"	Interdit	État "E3"	Interdit	`A2` État "E2"
AVANCE_STOP E3	Interdit	`A2` État "E1"	État "E2"	Interdit	`A2` État "E3"	Interdit

Table 6: Matrice compteur

- **Code :** `CounterAdvenced.java`

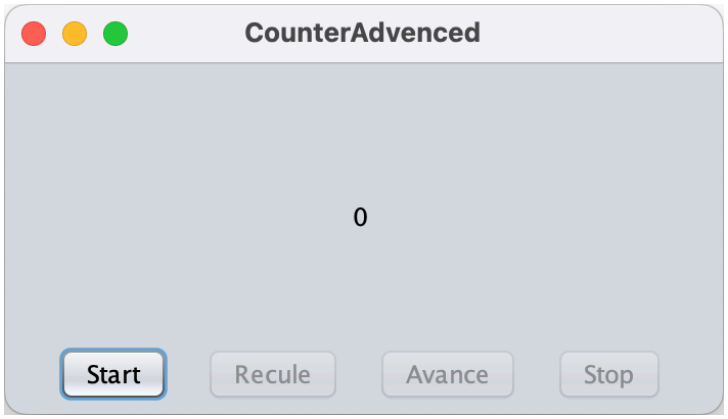


Figure 11: Capture d'écran de l'app Counter Advenced