

Le fonctionnel par la pratique

Live Coding d'un jeu Web

Quentin Duval @quduval
Guillaume Eveillard @geveillard

Protagonistes

QUENTIN

- 6 ans à Murex
- C++ le jour
- Haskell & Clojure

OBJECTIF

Live Coder

Ne pas se planter

Protagonistes

GUILLAUME

- 5 ans à Murex
- Java le jour
- Recherche language

OBJECTIF

Live Coder

Ne pas se planter

Clojure (Script)

- (((LISP)))
- Functional
- JVM: Clojure
- JS: ClojureScript





35 min de Talk

5 min Q/R

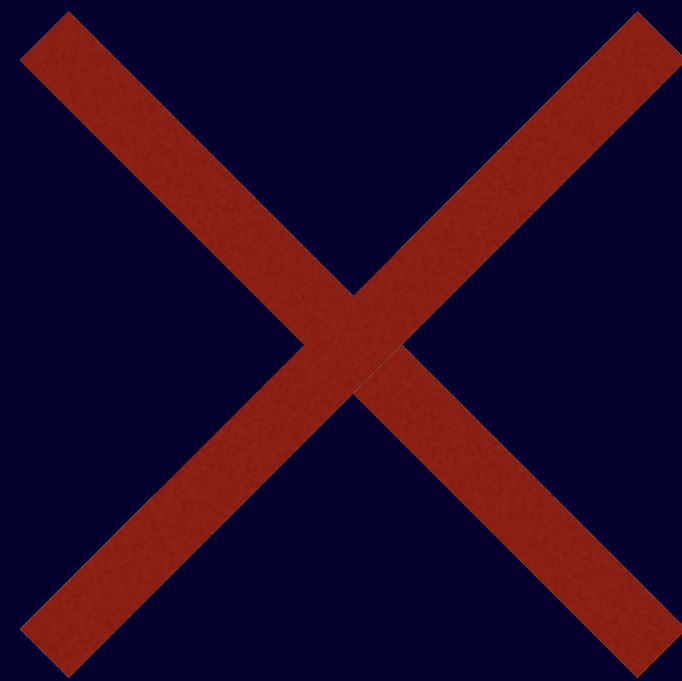


Clojure en 5 min 28

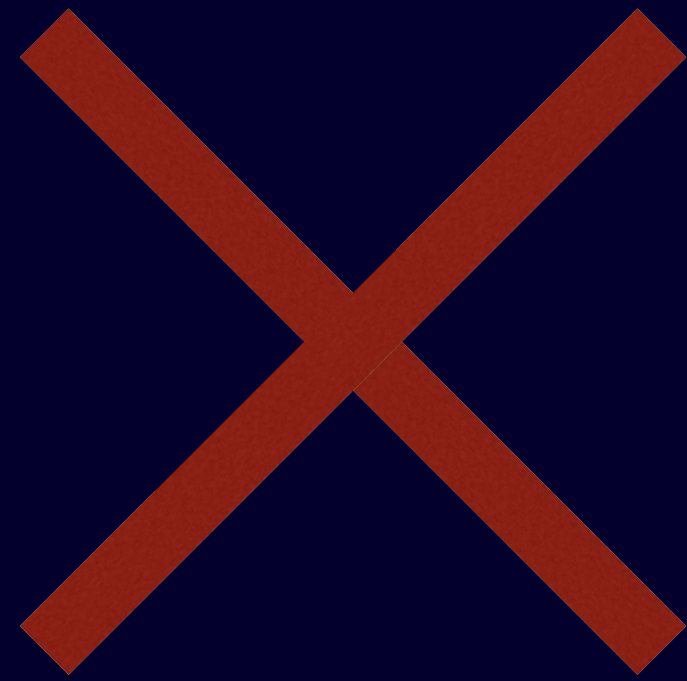
Demo REPL

TicTacToe en 25 min 17

Owner



Owner



:owner/cross

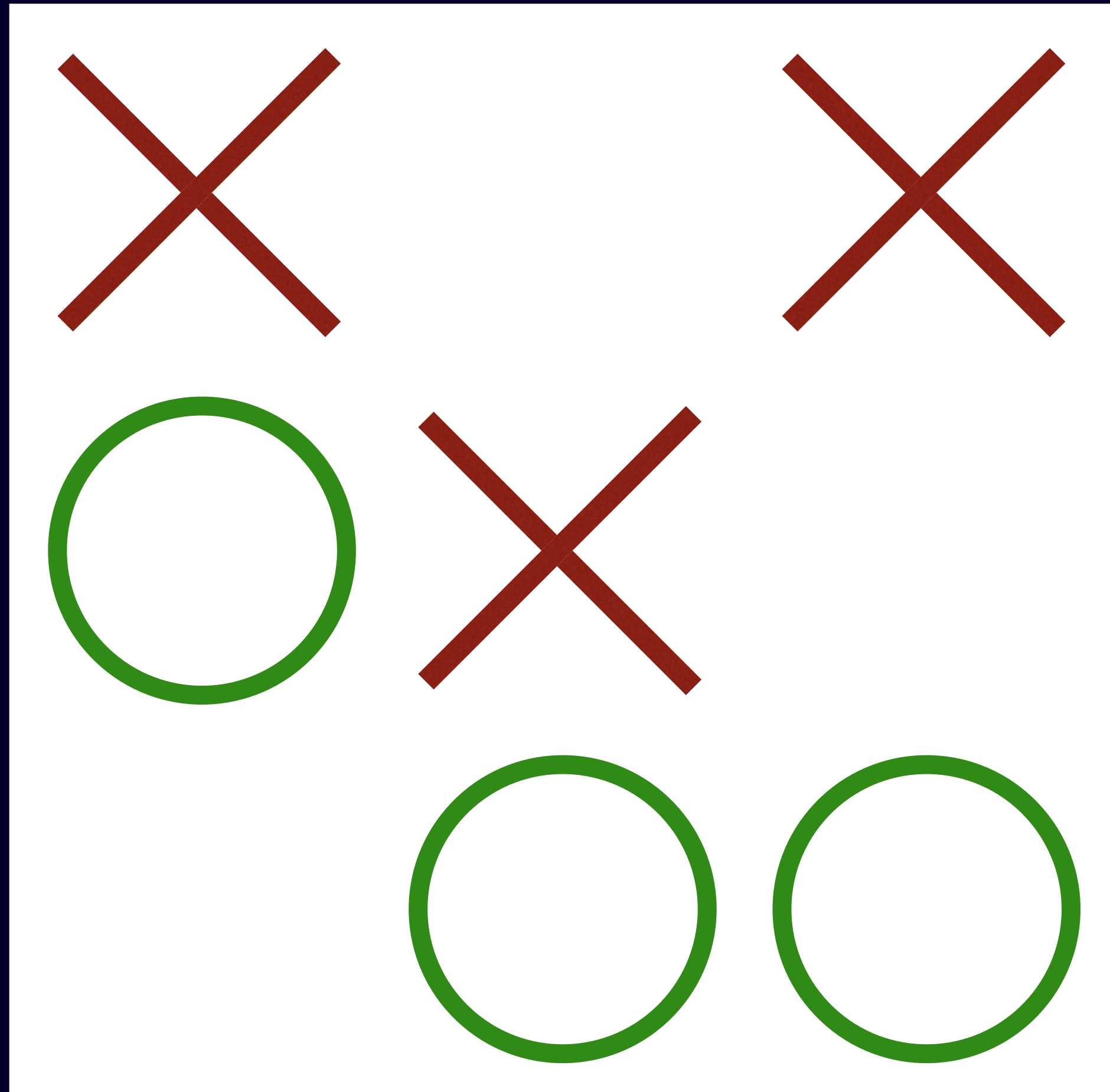


:owner/circle

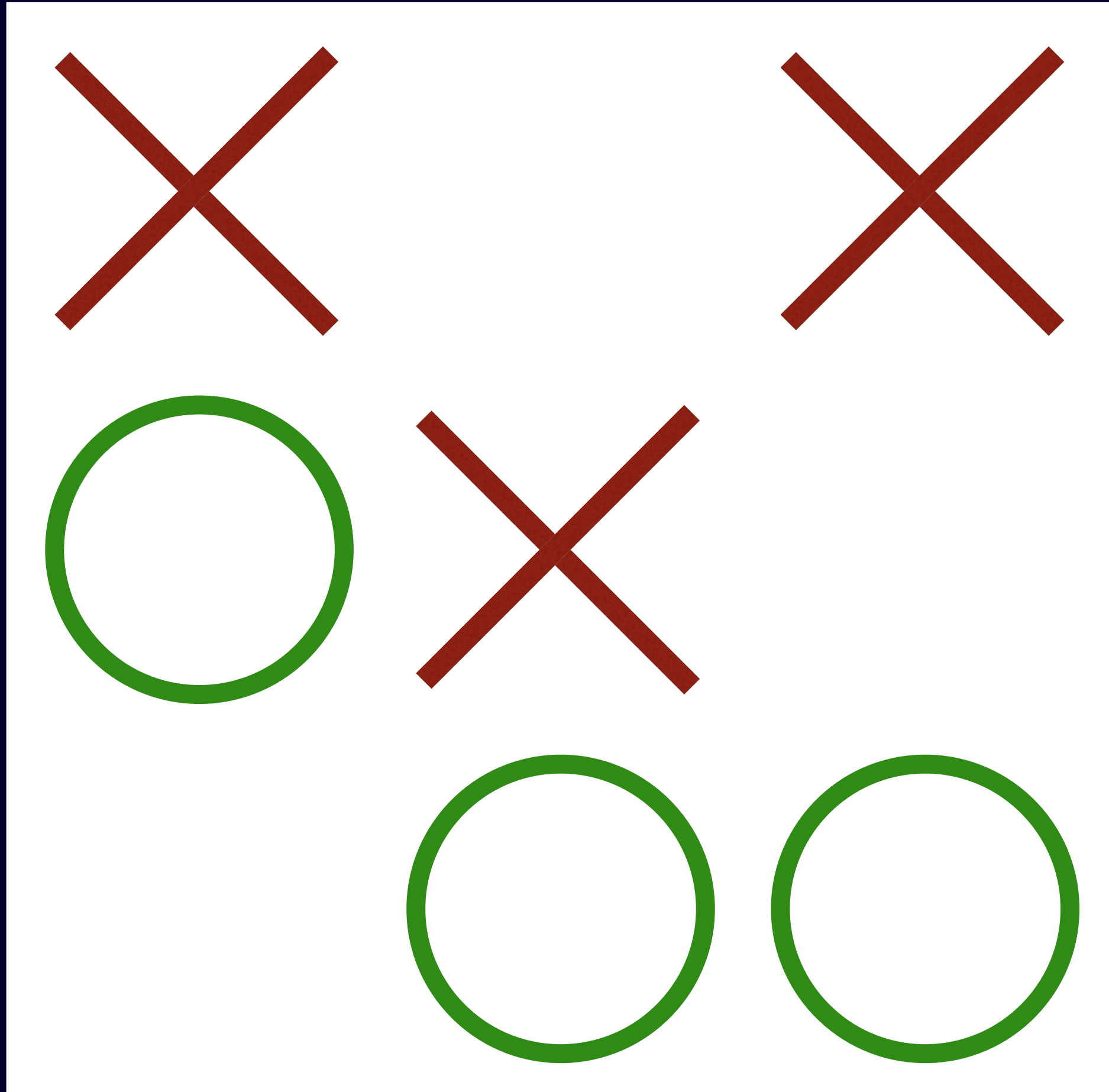


:owner/none

Board

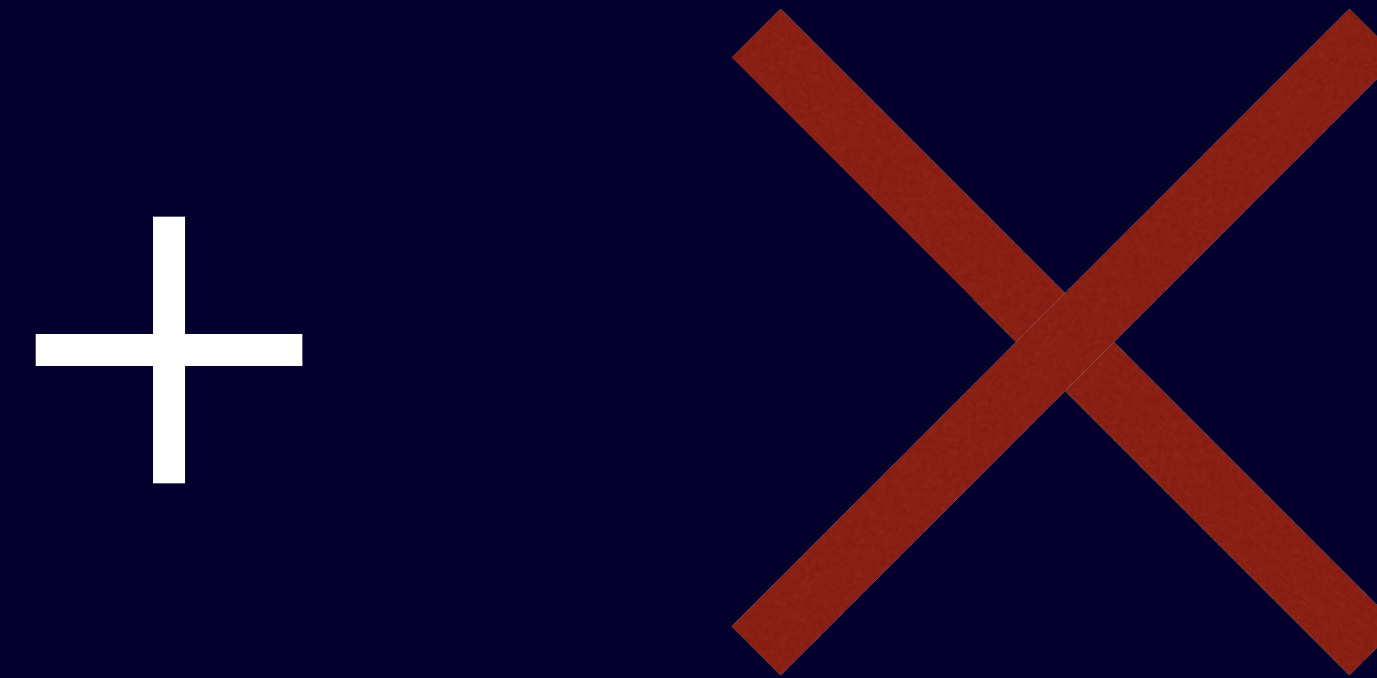
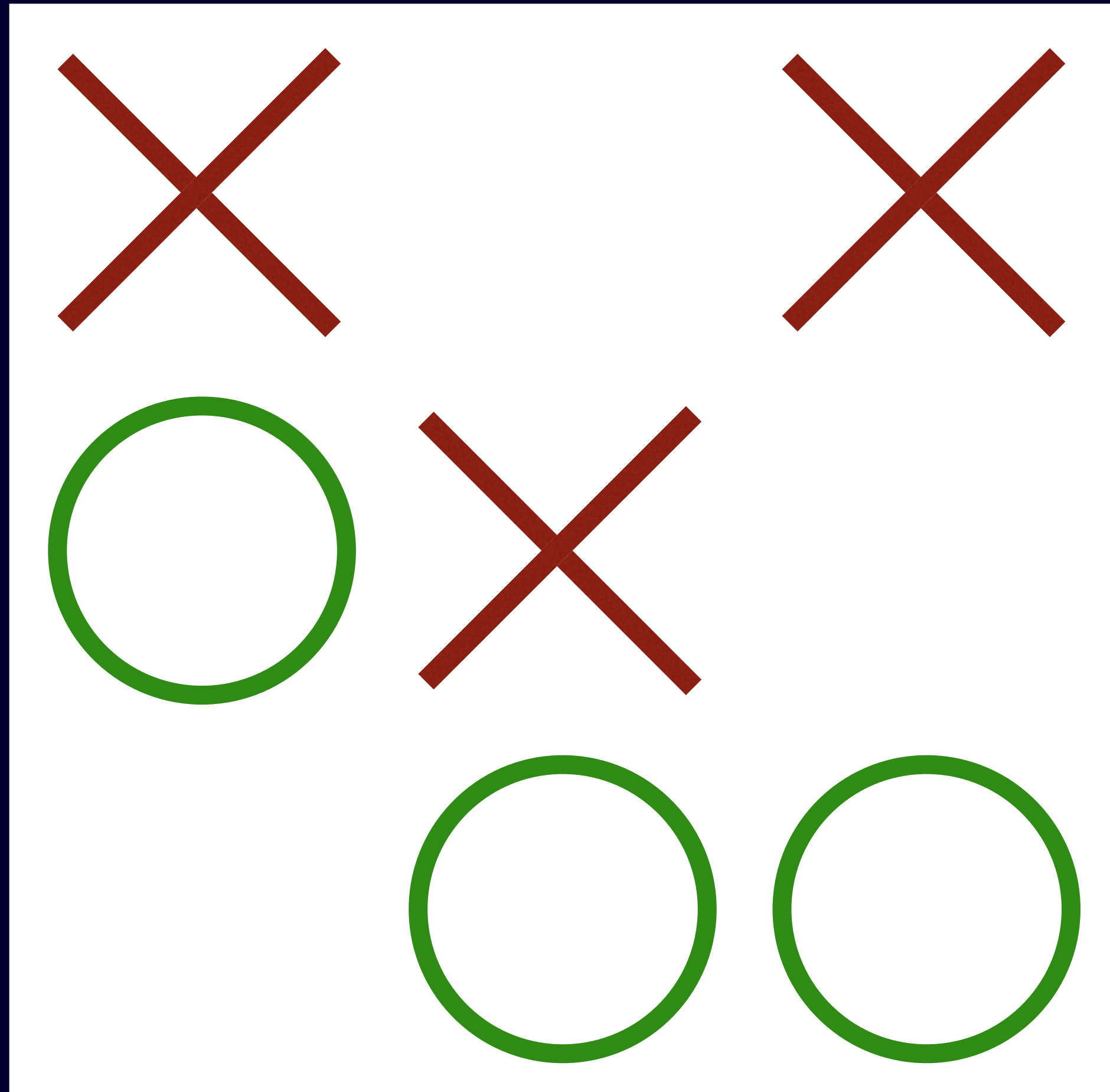


Board



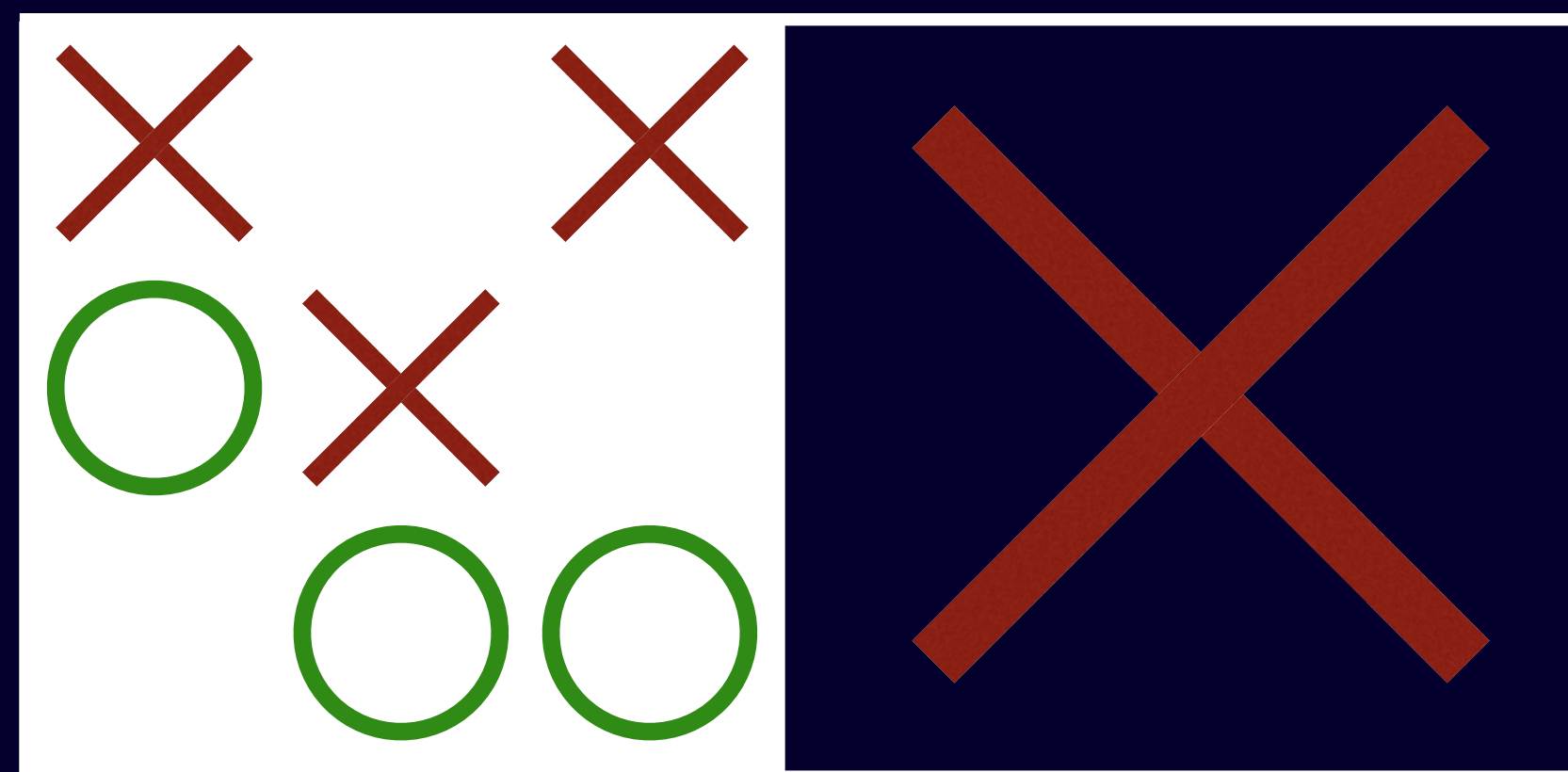
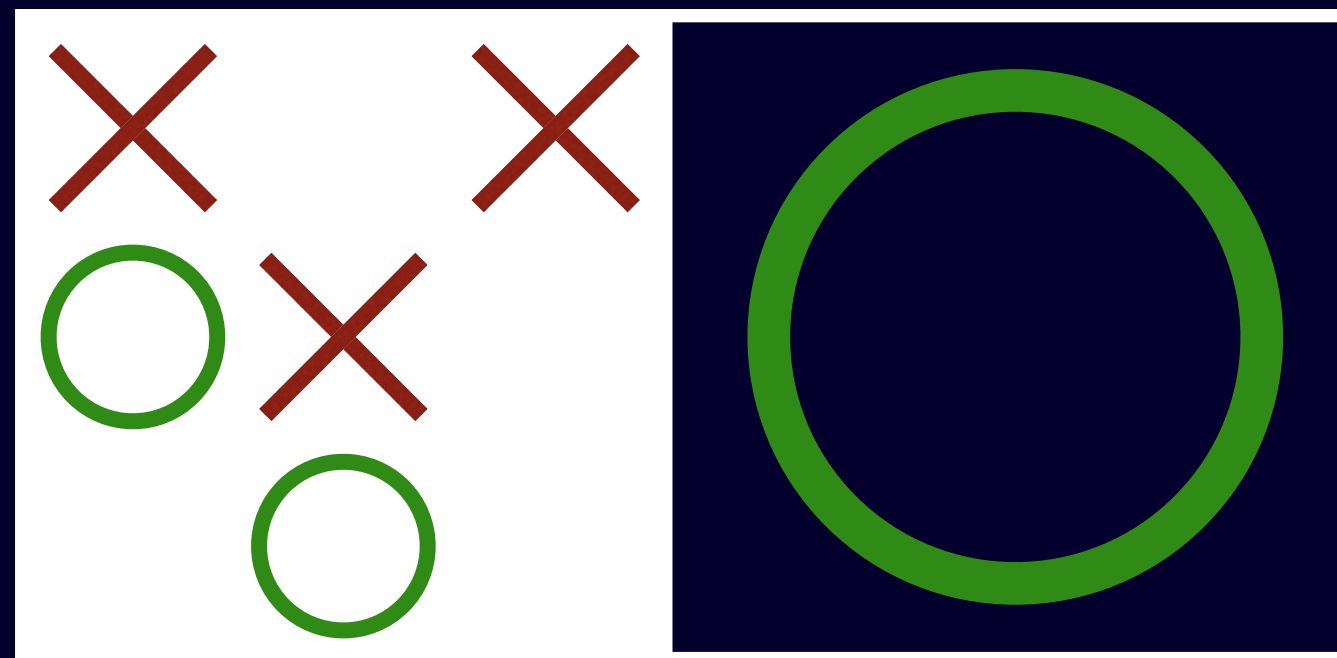
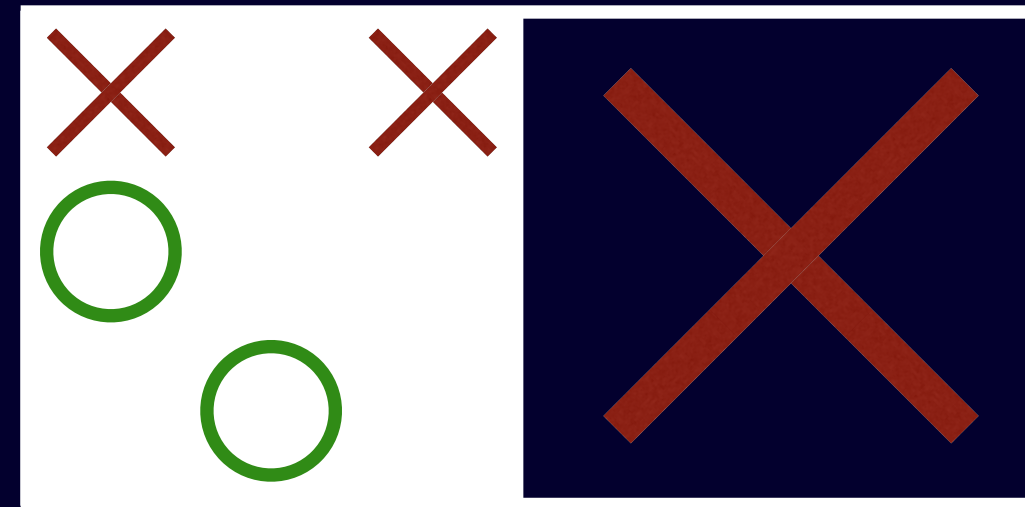
```
{ [0 0] :owner/cross  
  [1 0] :owner/circle  
  [2 0] :owner/none  
  ... }
```

Turn

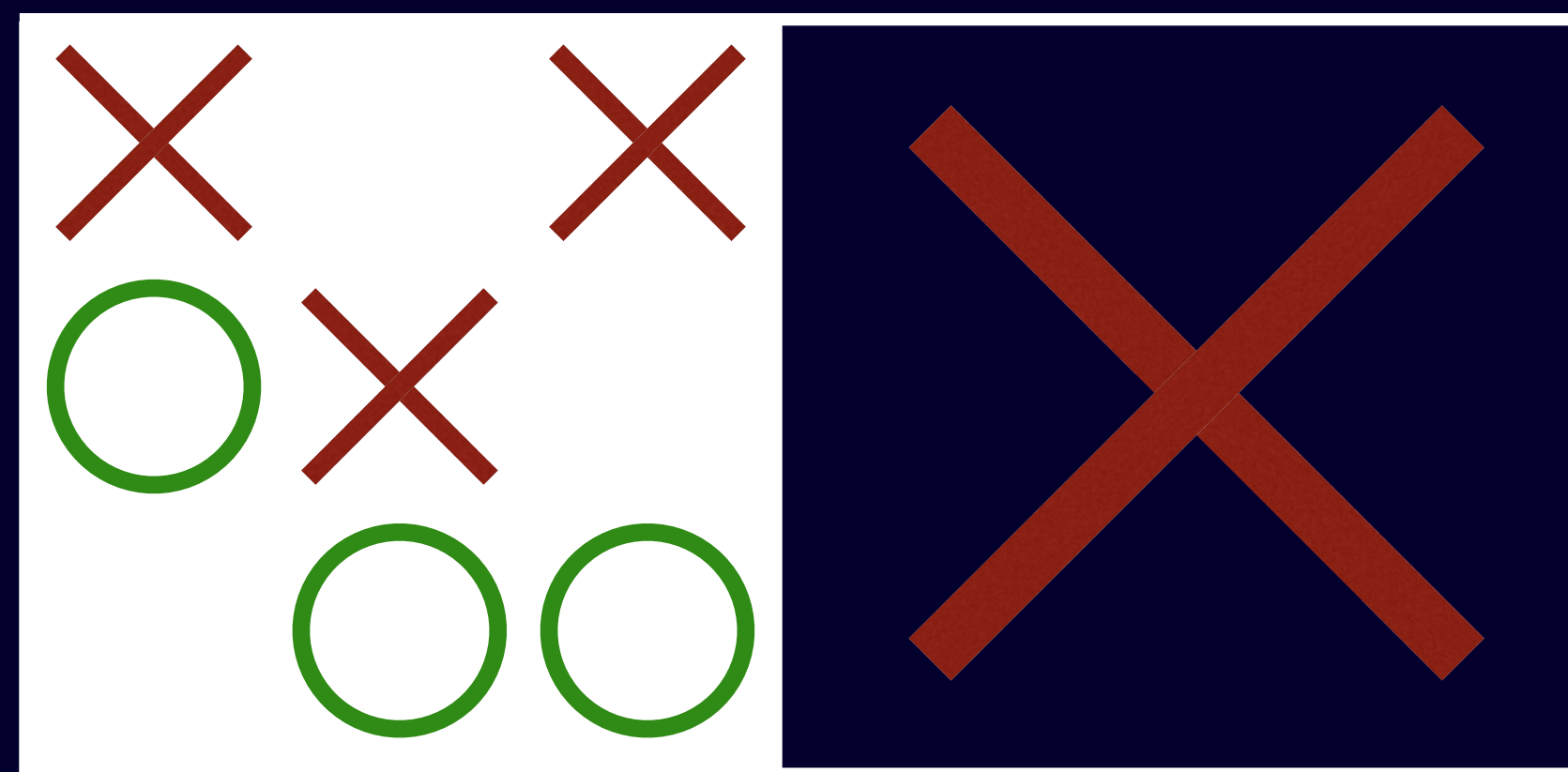
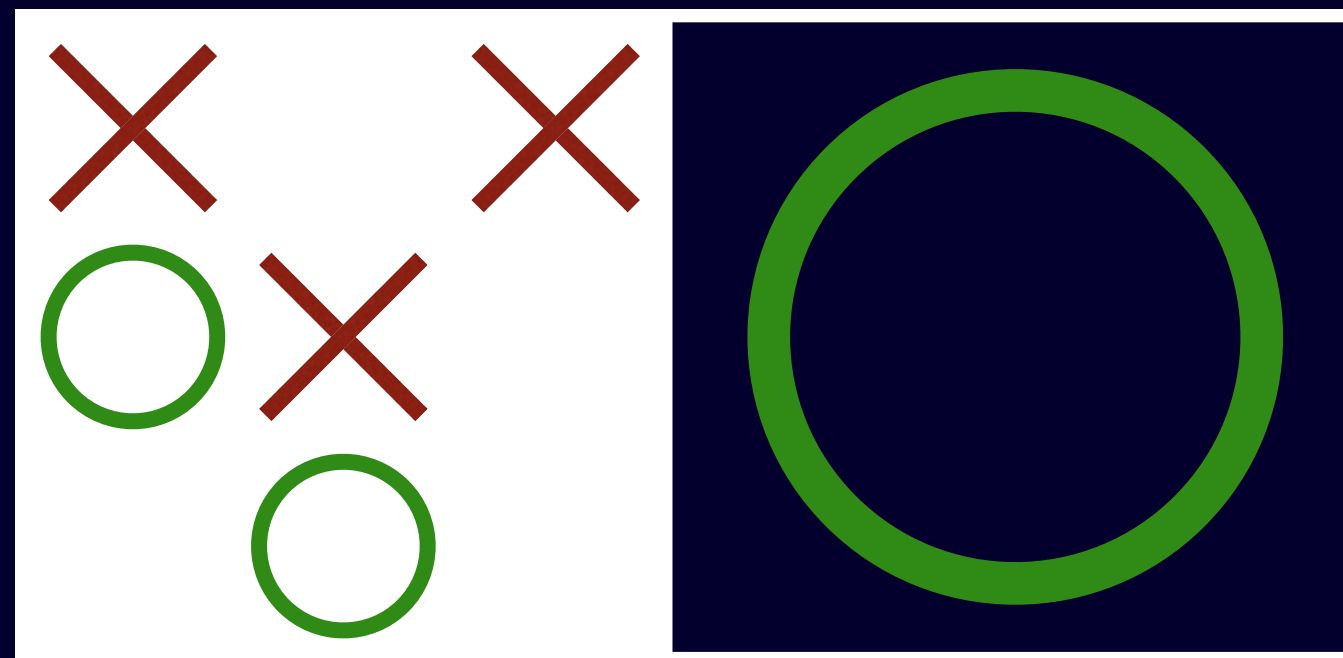
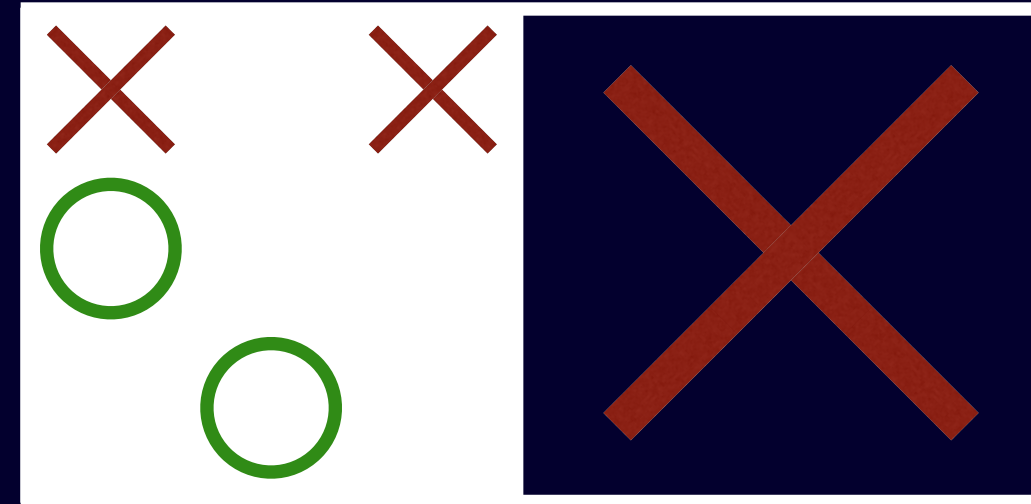


```
{ :board ...  
  :player ... }
```


Game



Game



[turn-0

turn-1

...

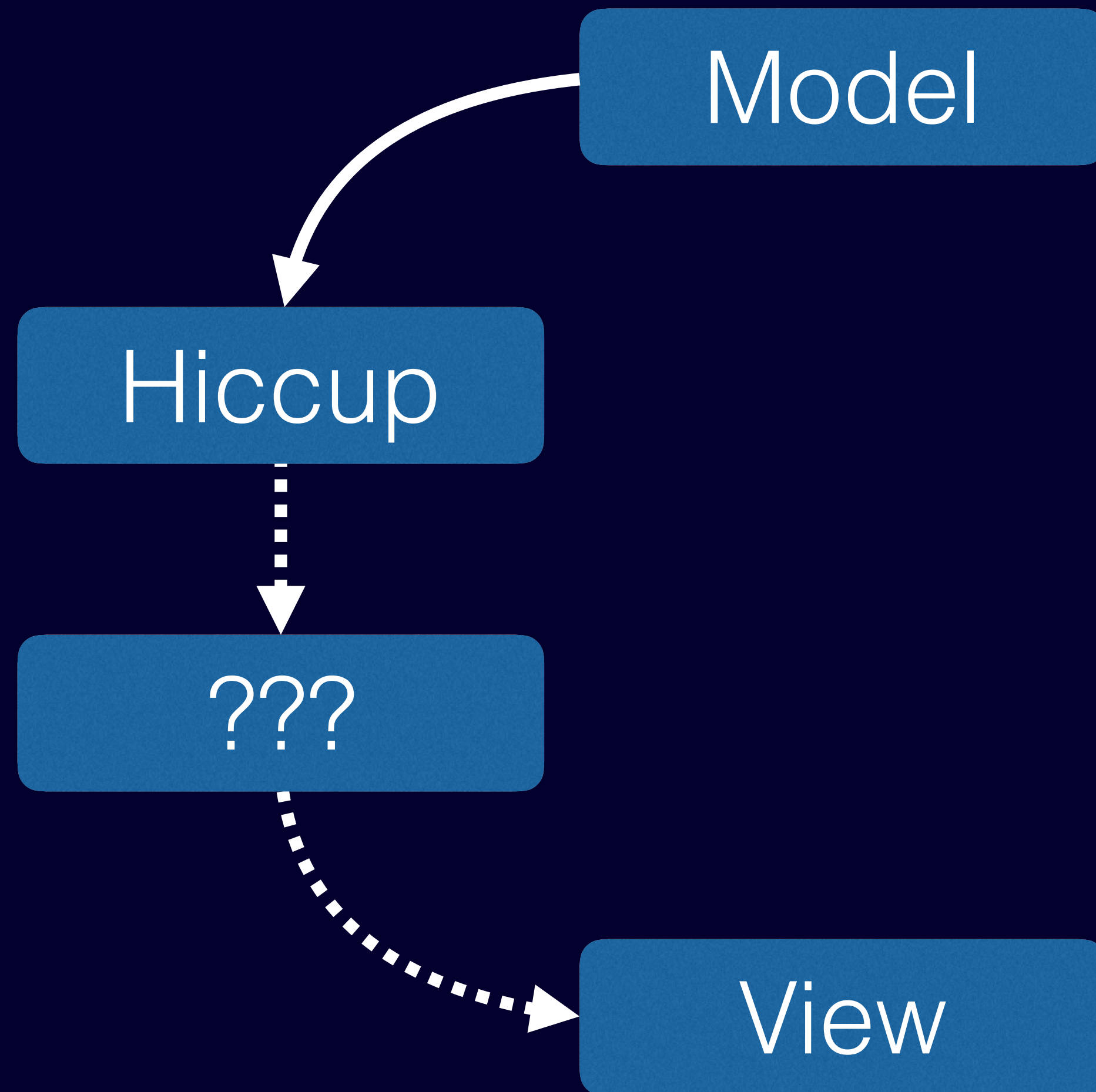
previous-turn

current-turn]

Design

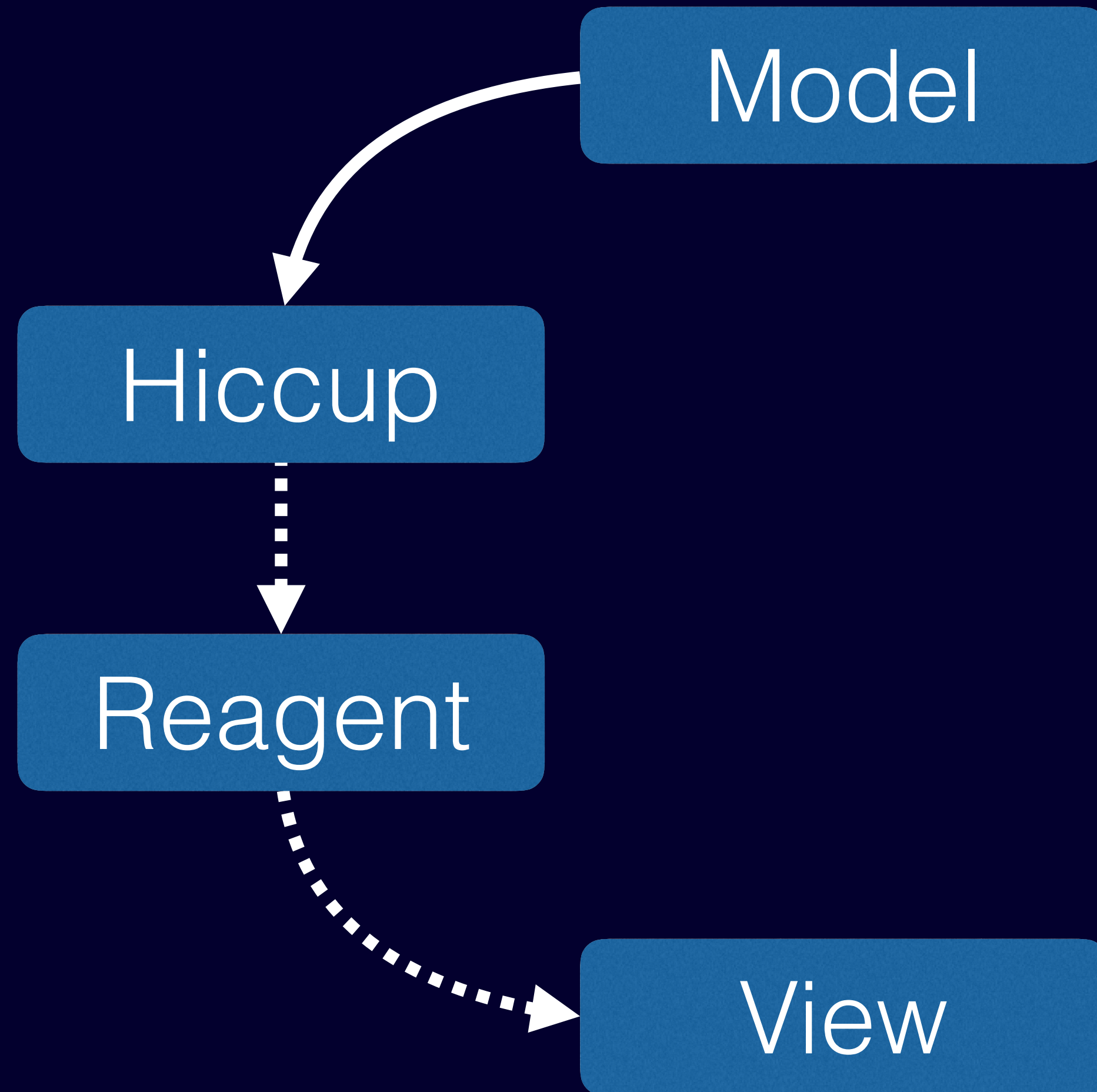


Design



Live Code

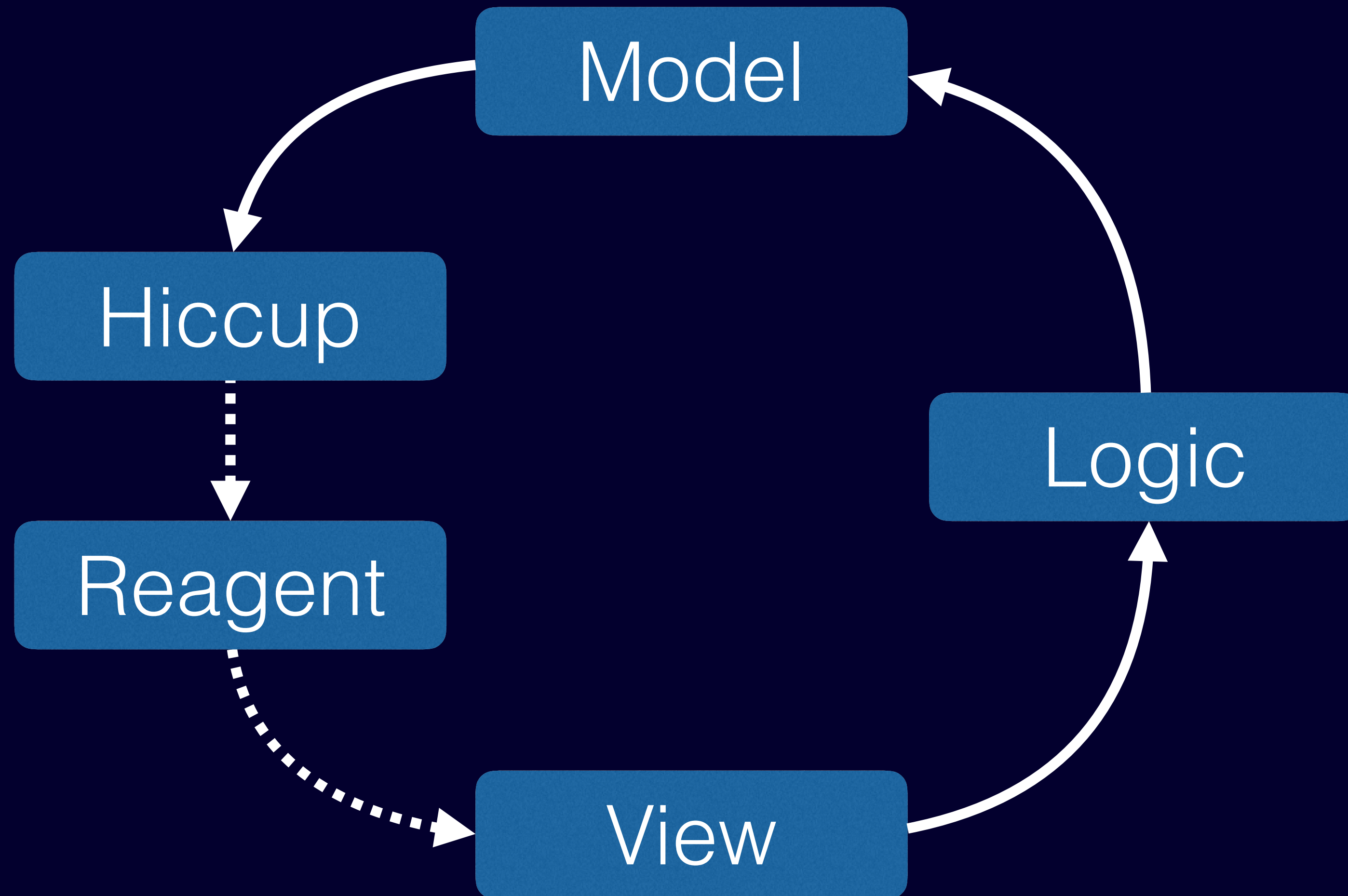
Reagent



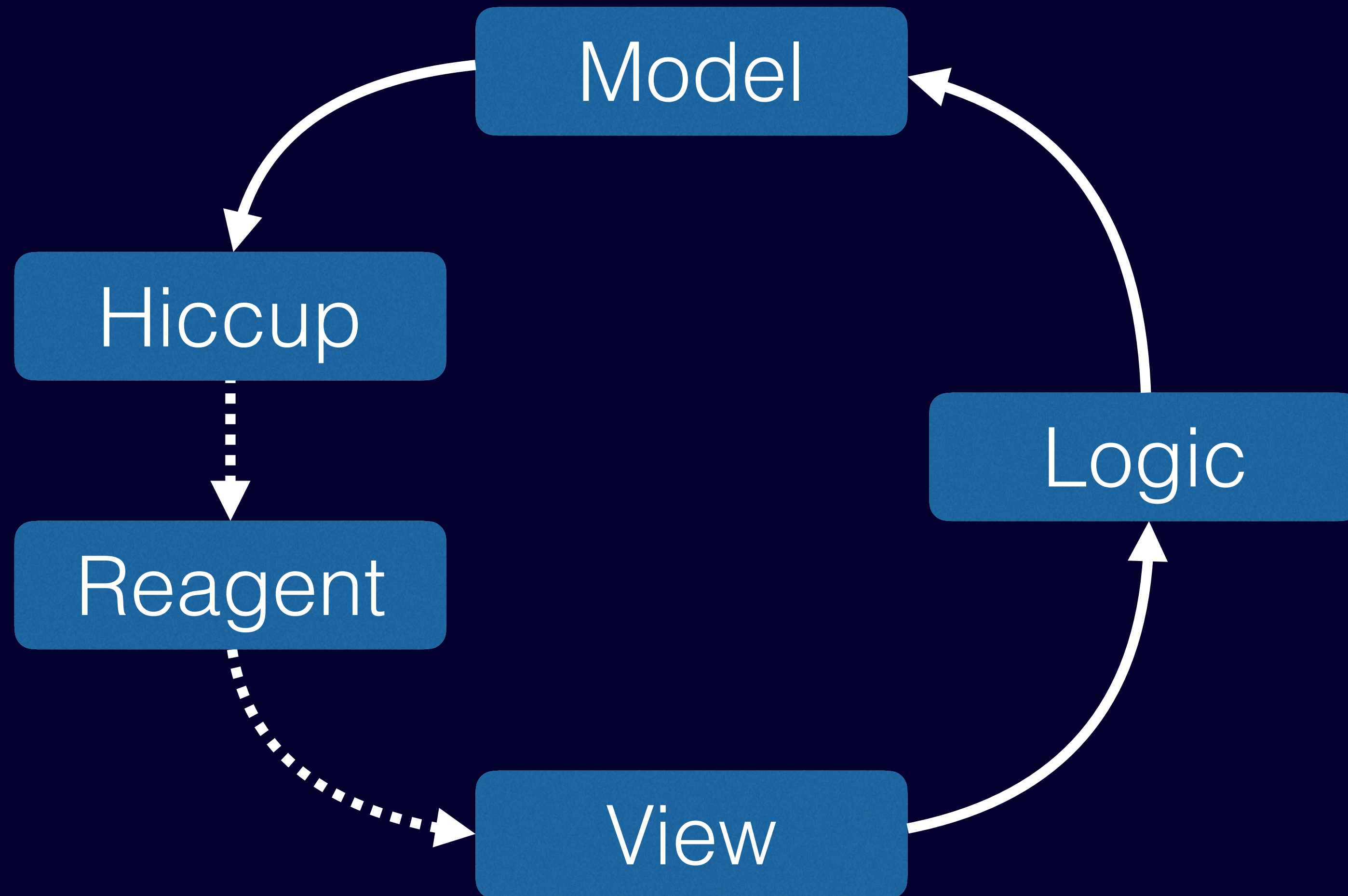
- Surcouche sur React
- Intégré dans le language
- Pas de templating

Live Code

Design

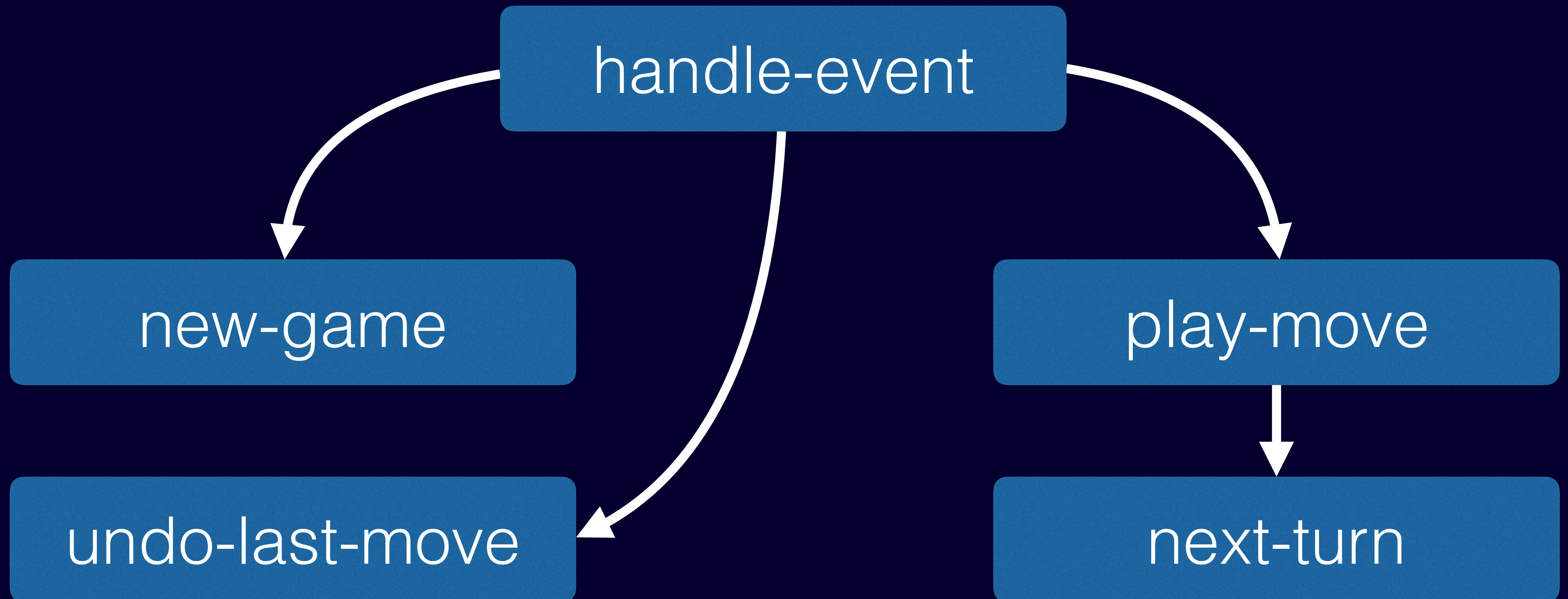


Design



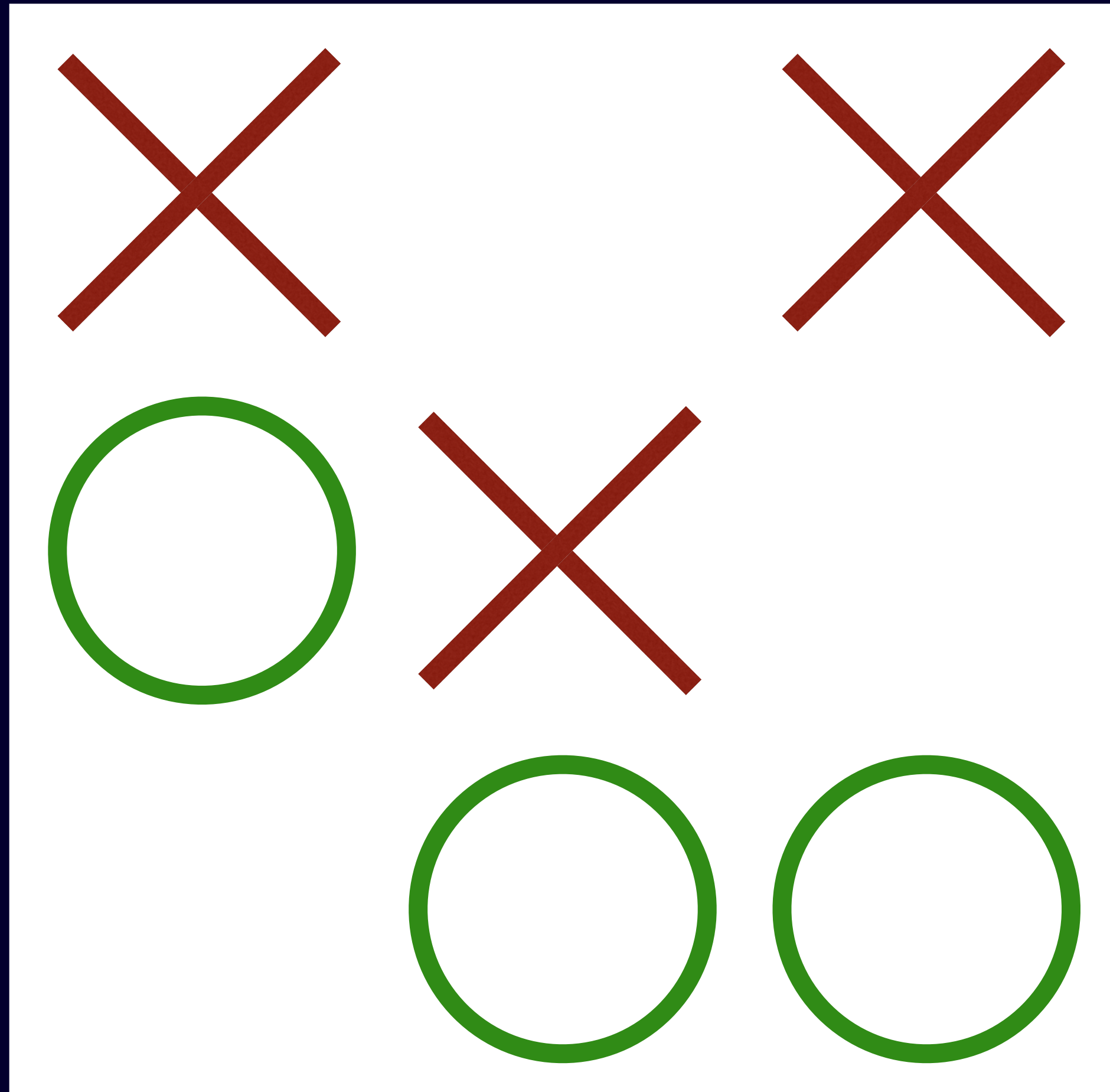
Modèle
+
Evènement
=
Nouveau modèle

Handling events

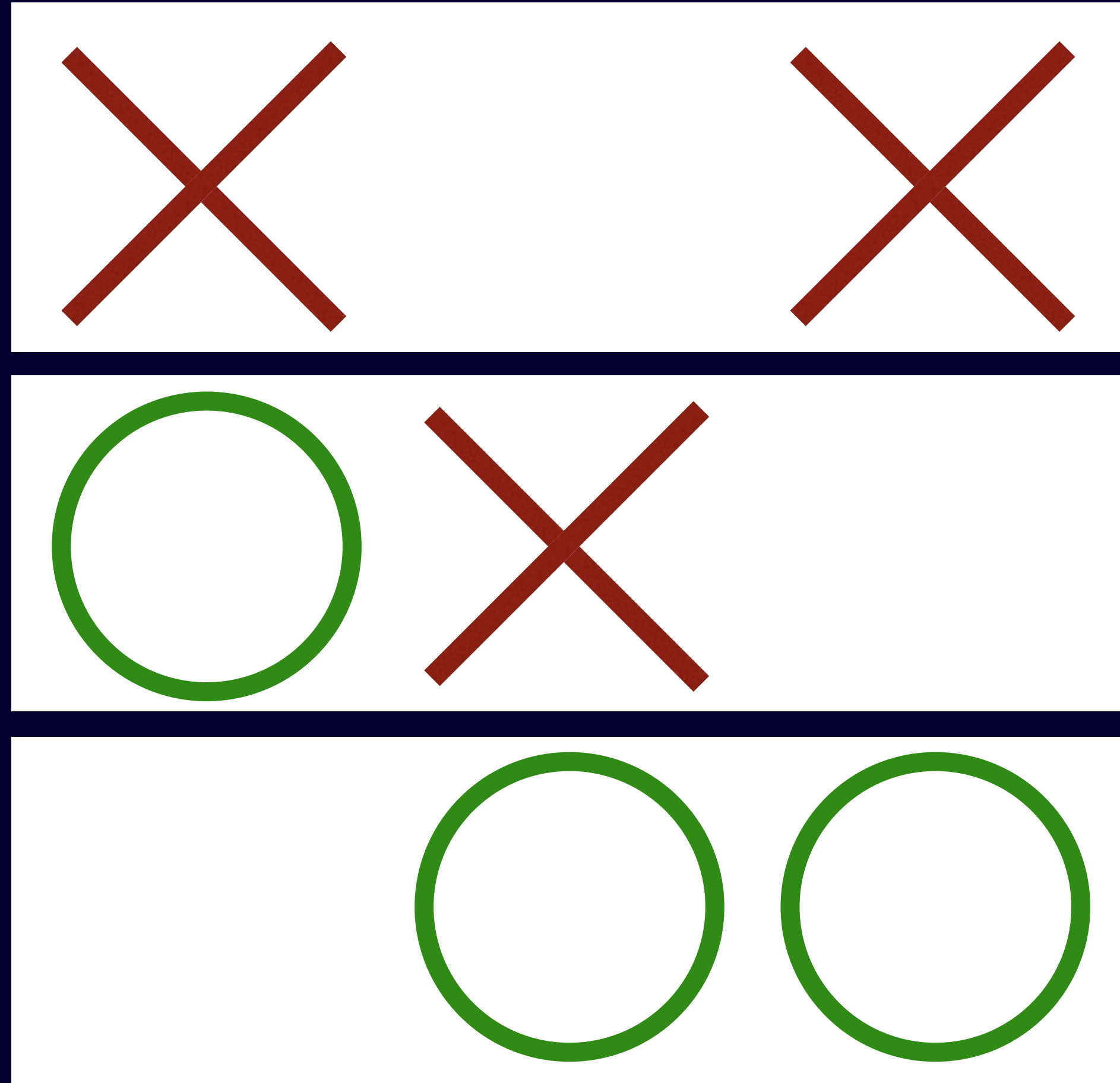


Live Code

Conditions de victoire



Conditions de victoire

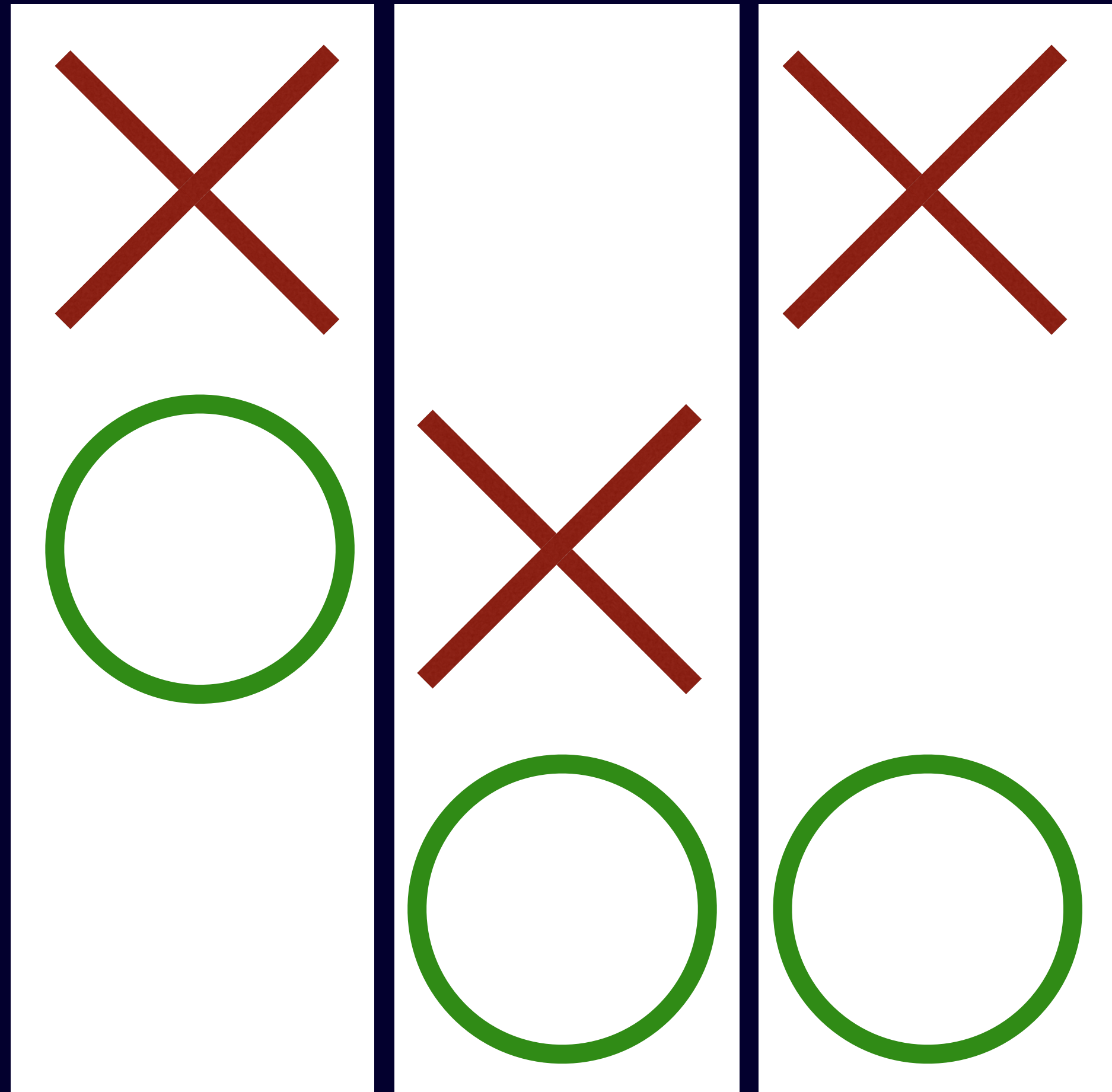


$[[0 \ 0] \ [0 \ 1] \ [0 \ 2]]$

$[[1 \ 0] \ [1 \ 1] \ [1 \ 2]]$

$[[2 \ 0] \ [2 \ 1] \ [2 \ 2]]$

Conditions de victoire

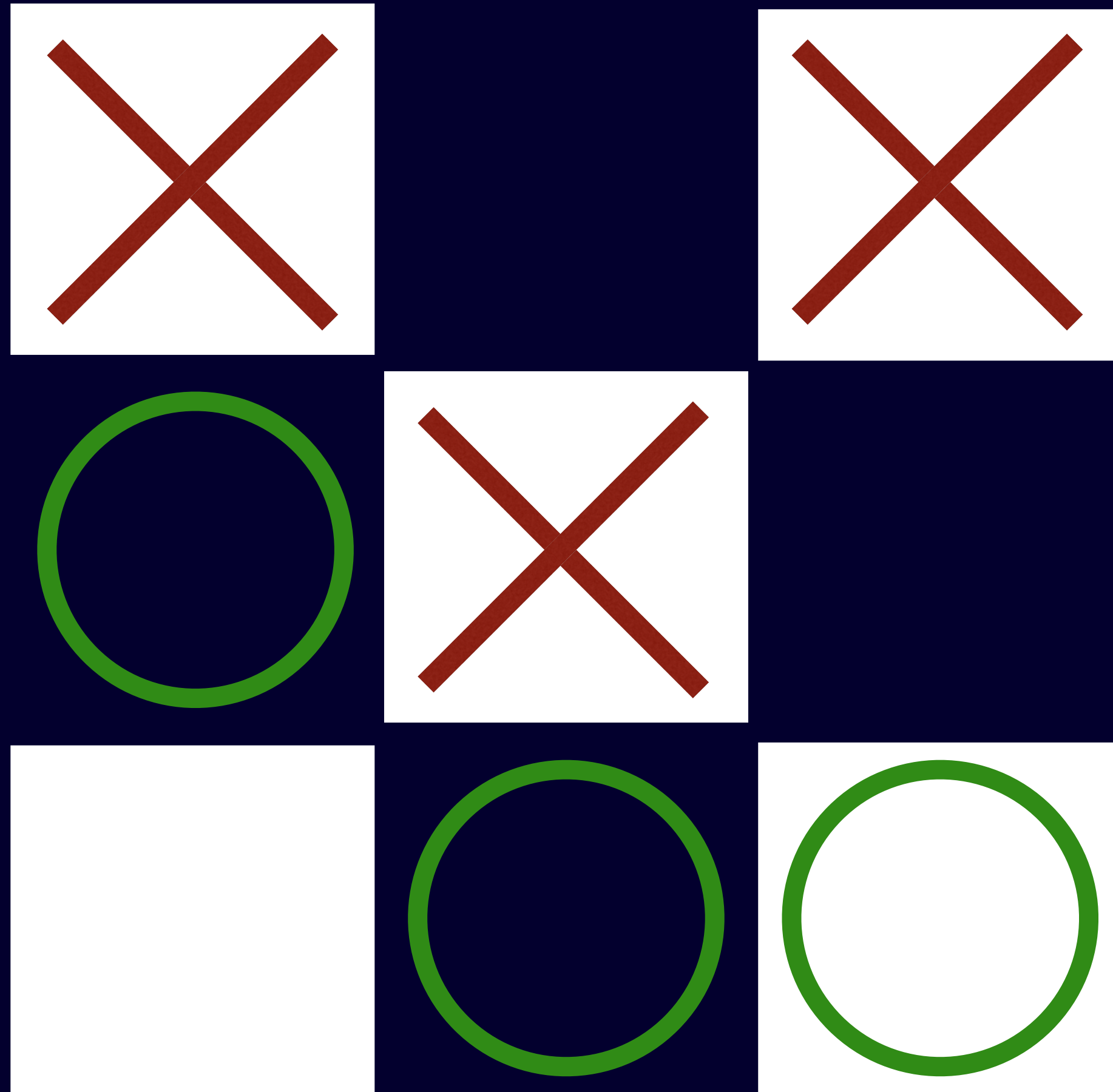


$[[0 \ 0] \ [1 \ 0] \ [2 \ 0]]$

$[[0 \ 1] \ [1 \ 1] \ [2 \ 1]]$

$[[0 \ 2] \ [1 \ 2] \ [2 \ 2]]$

Conditions de victoire

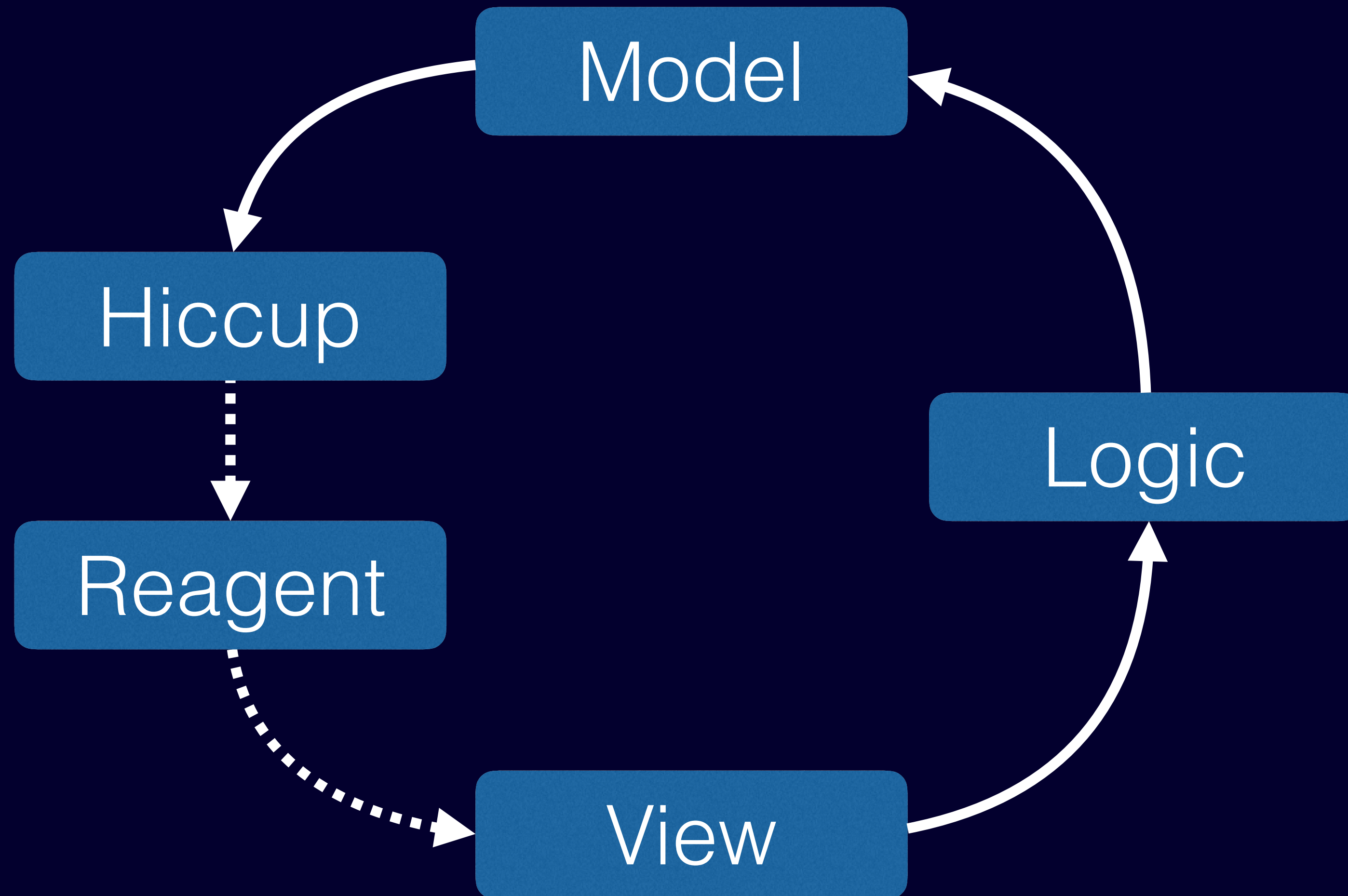


$[[0 \ 0] \ [1 \ 1] \ [2 \ 2]]$

$[[0 \ 2] \ [1 \ 1] \ [2 \ 0]]$

Live Code

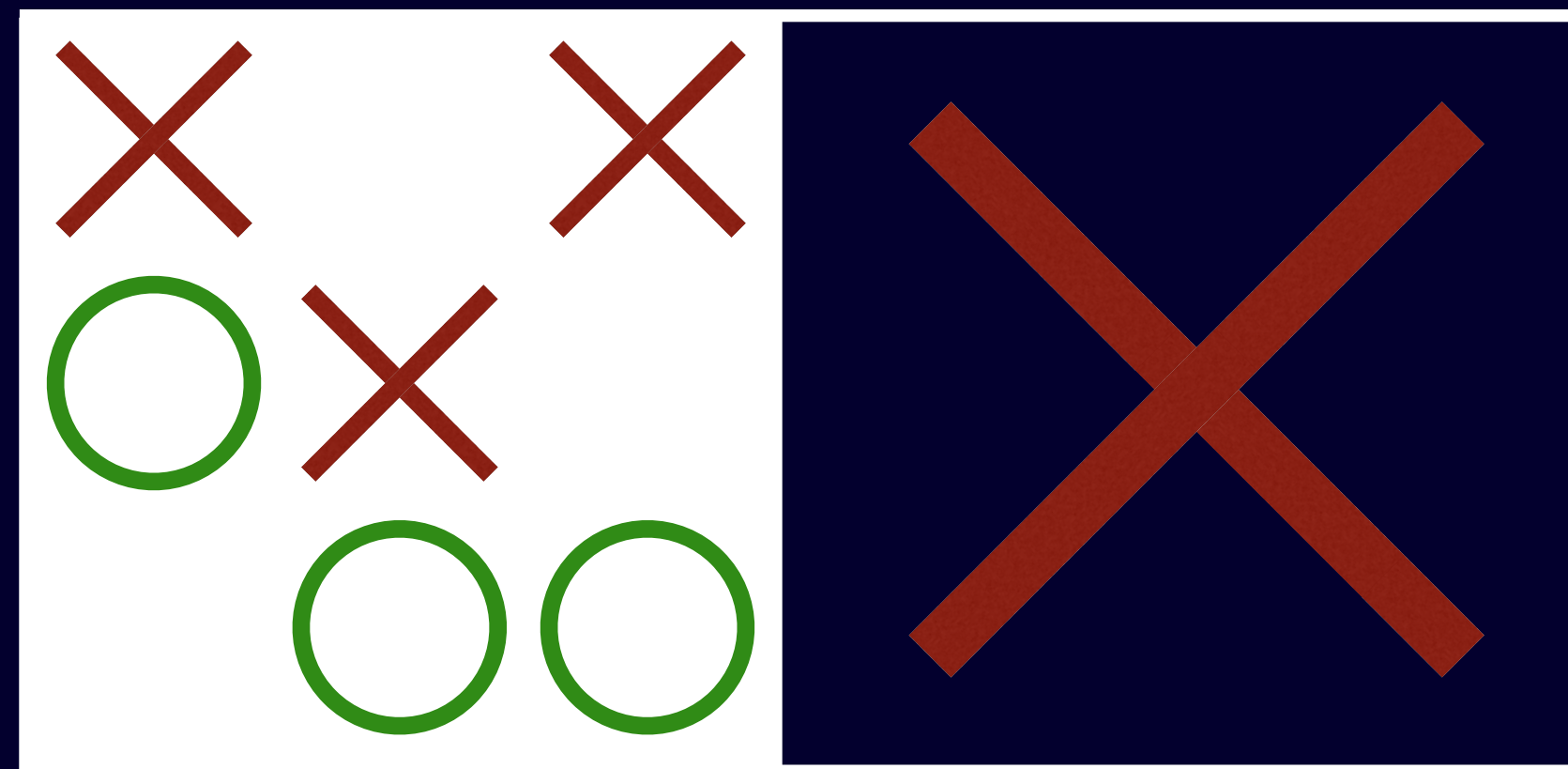
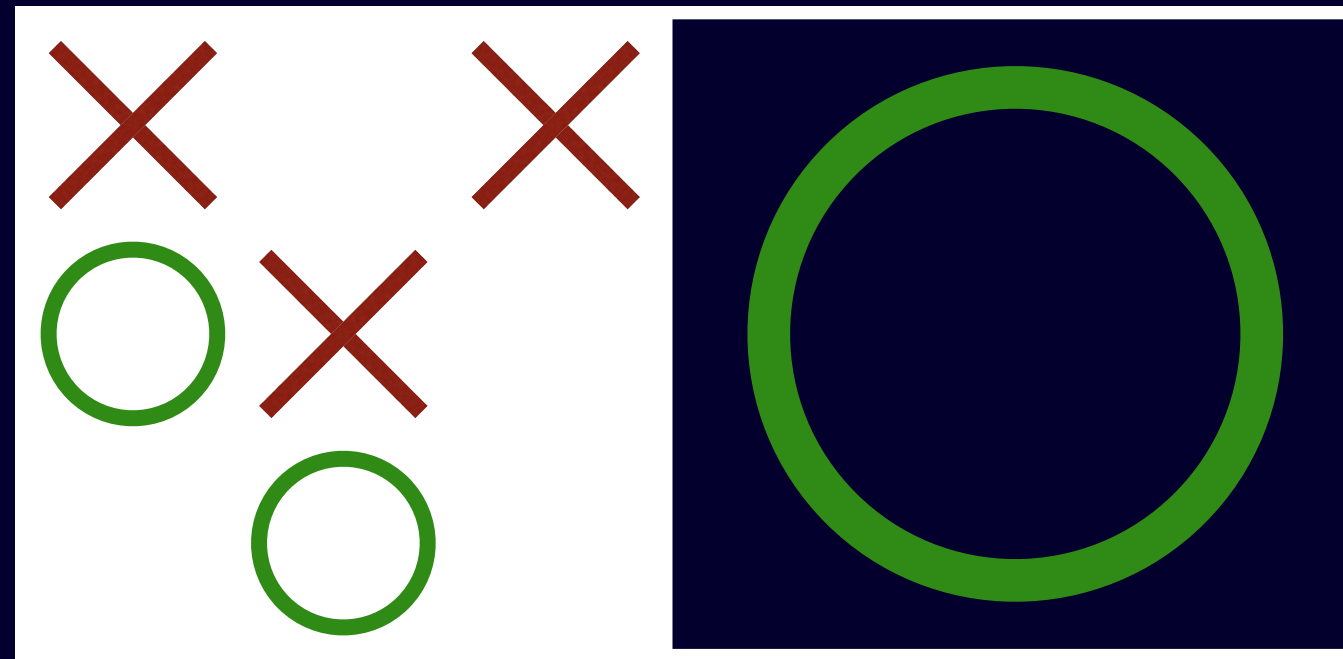
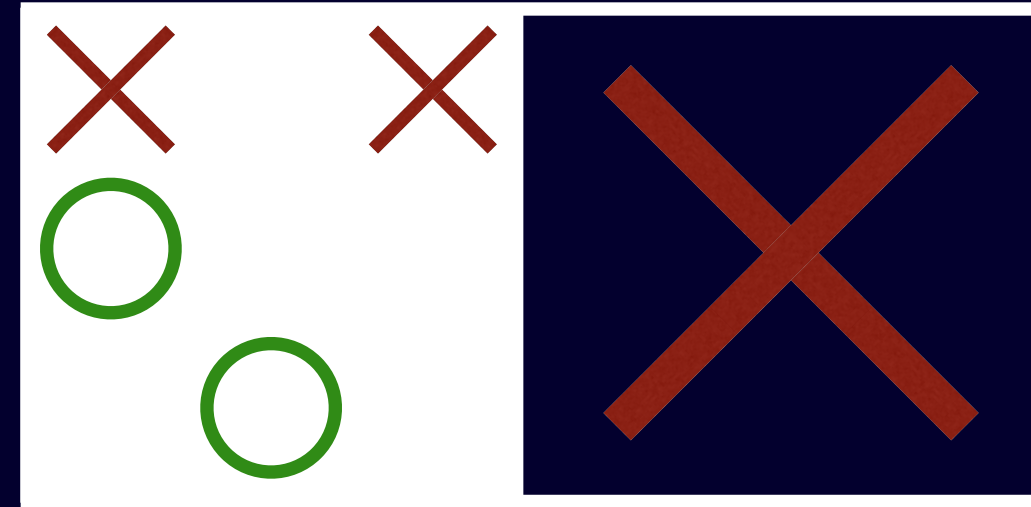
Design



- Simple
- Un seul d'état
- Testable
- Évènements métier

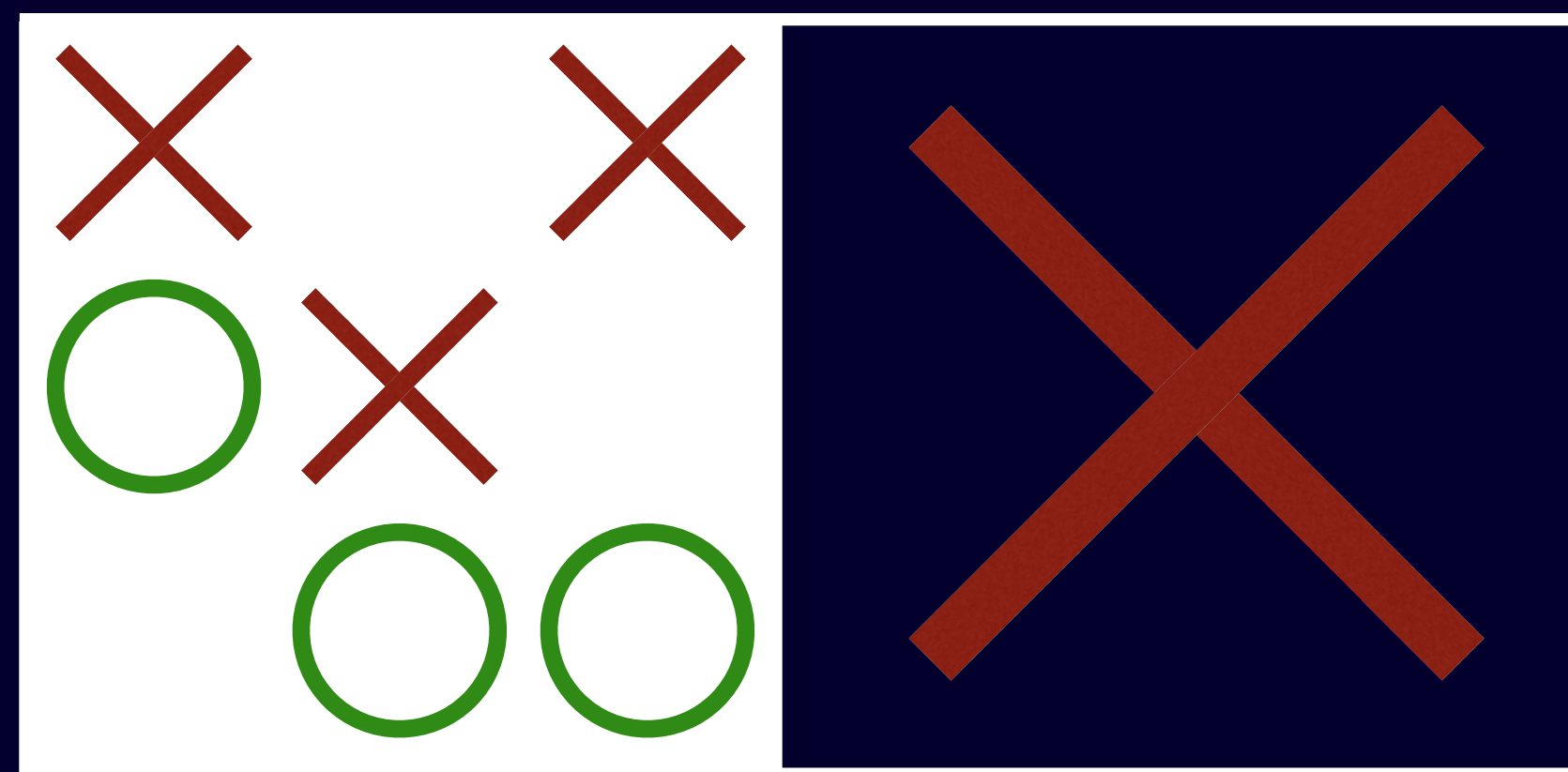
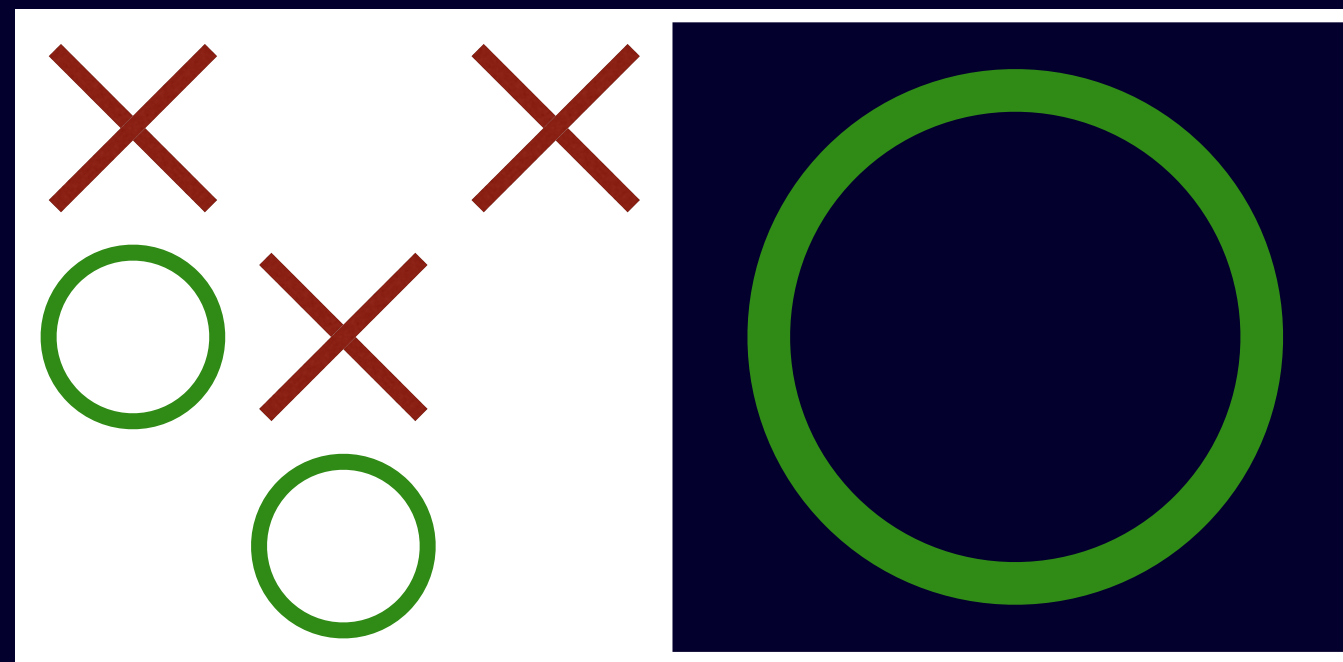
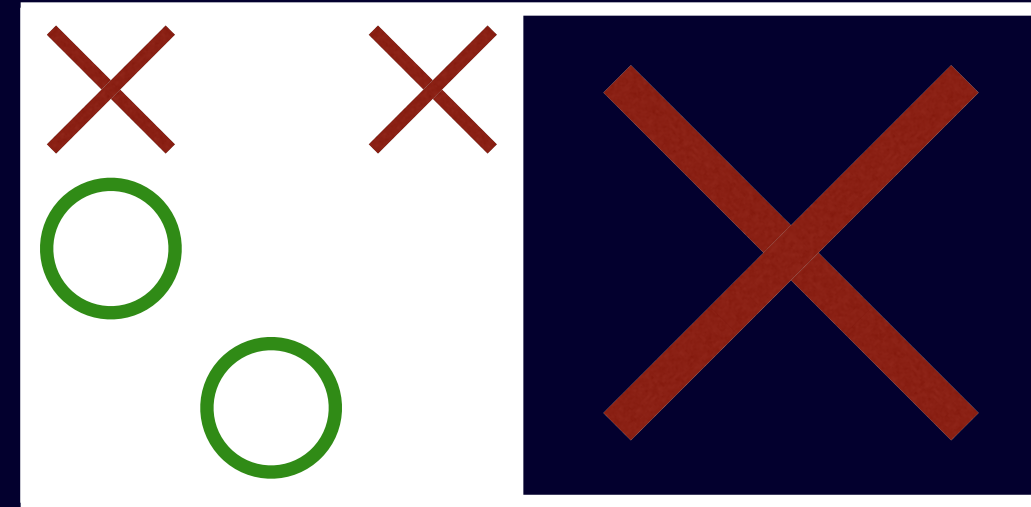
Le CTRL-Z

Etat = Succession de Valeurs



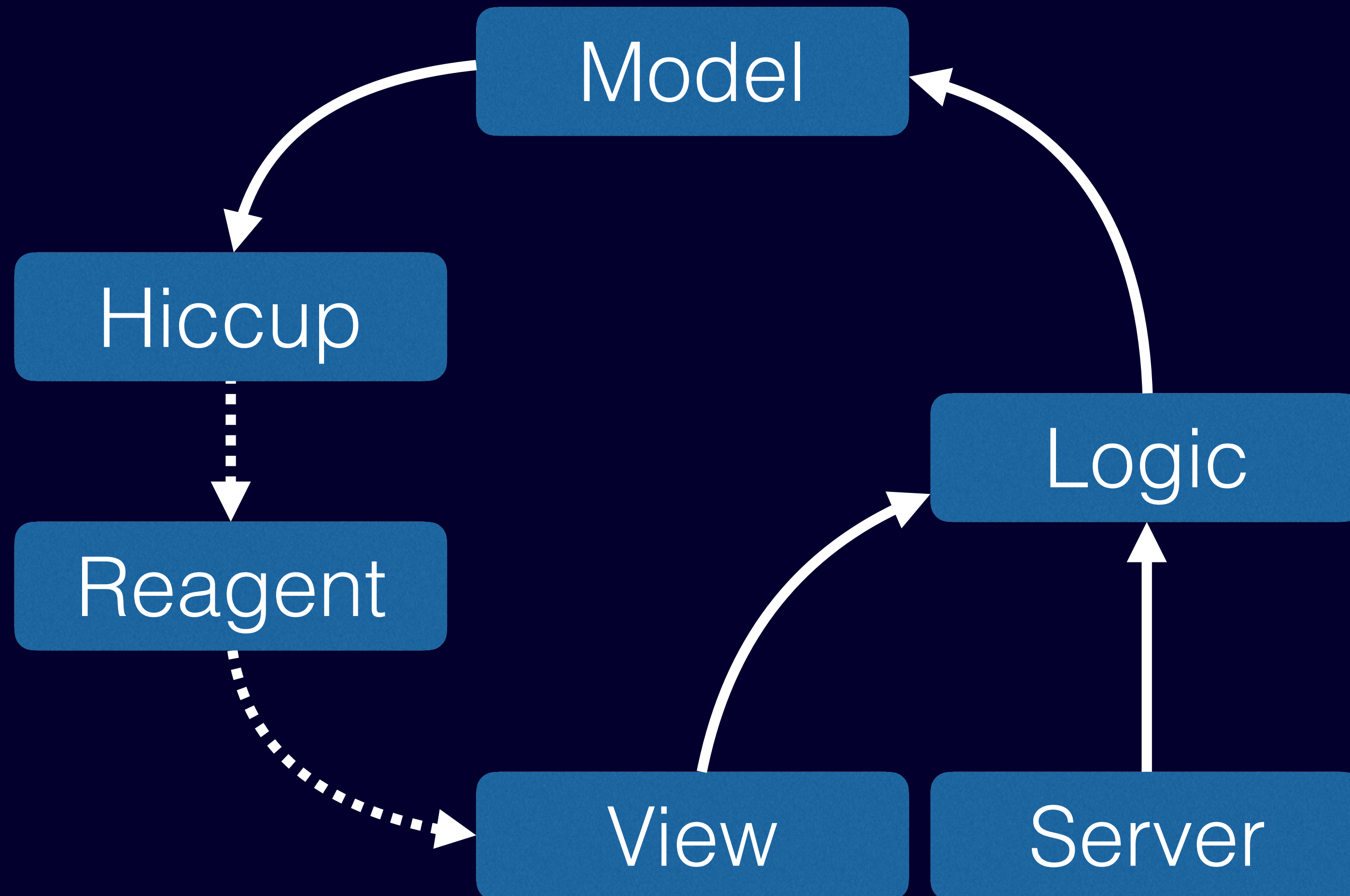
Live Code

Etat = Succession de Valeurs



- Une catégorie entière de problèmes en moins
- Rejouable
- Observable
- Efficient

Un pattern efficace



- Évènement serveur
- Découplé de la source
- Scalable
- Thread-safe

Le fonctionnel c'est...

- Accessible
- Concret
- Simple et efficace
- Source d'inspiration



Merci

Jouez au jeu:

<https://quentinduval.github.io/tictactoe>

Présentation et ressources:

<https://github.com/QuentinDuval/TicTacToeDevoxx/tree/master>

Blog post dédié:

<https://deque.blog/2017/03/03/building-a-clojurescript-game-architecture-poc>