# STATE MACHINE STRIKE BACK

#### SI ON ADAPTAIT LES COMMANDES... EN ACTIONS

#### MIAGE M2 - QUALITÉ DU SI - THOMAS HAESSLÉ

```
interface Face {
    type: "face";
    direction: Direction;
interface Start {
    type: "start";
interface Stop {
    type: "stop";
type Action =
      Face
      Start
      Stop
const stop: () => Stop = () => ({ type: "stop" });
const start: () => Start = () => ({ type: "start" });
const face: (d: Direction) => Face = (d: Direction) => ({ type: "face",
direction: d });
```

Rien de surprenant, on met simplement les paramètres (direction) dans l'interface Face, Start et Stop sont des types (vs des constructeurs en OCaml), on peut créer nos constructeurs de valeurs





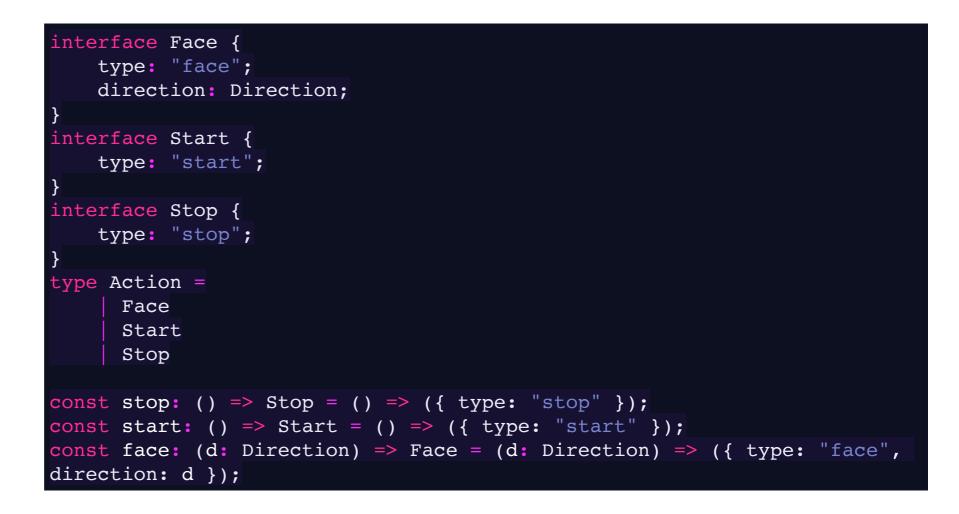




## STATE MACHINE STRIKE BACK

SI ON ADAPTAIT LES COMMANDES... EN ACTIONS





Rien de surprenant, on met simplement les paramètres (direction) dans l'interface

Face, Start et Stop sont des types (vs des constructeurs en OCaml), on peut créer nos constructeurs de valeurs

## STATE MACHINE STRIKE BACK

#### DONC ON VEUT STOCKER L'ÉTAT



```
interface Idle {
    type: "idle";
}
interface Moving {
    type: "moving";
}
type State = Idle | Moving

const idle : () => Idle = () => ({
    type: "idle"
});
const moving : () => Moving = () => ({
    type: "moving"
});
const initialState : State = idle();
```