





**EXCEPTIONS**

RAISE/THROW ARE UGLY GOTOS

MAGNET 2-DUALS - THOUGHTS

```
type weapon
type target
type impacted = | Impacted

let arm_your_bow : weapon = raise (Failure "NoMoreArrow")
let targeted_monster : target = raise (Failure "TooMuchFog")
let hit_monster : weapon -> target -> impacted =
    fun w t -> Impacted
```

```
class Weapon {
    Weapon(){
        throw new Exception("NoMoreArrow");
    }
}

class Target {
    Target(){
        throw new Exception("TooMuchFog");
    }
}

class Impacted{}

public class Main {
    Weapon armYouBow = new Weapon();
    Target targetMonster = new Target();
    Impacted hitMonster(Weapon w, Target t) {
        return new Impacted();
    }
}
```

On remarquera le manque d'expressivité de Java



# EXCEPTIONS

RAISE / THROW ARE UGLY GOTOS

```
class Weapon {
    Weapon(){
        throw new Exception("NoMoreArrow");
    }
}
class Target {
    Target(){
        throw new Exception("TooMuchFog");
    }
}
class Impacted{}

public class Main {
    Weapon armYouBow = new Weapon();
    Target targetMonster = new Target();
    Impacted hitMonster(Weapon w, Target t) {
        return new Impacted();
    }
}
```

```
type weapon
type target
type impacted = | Impacted

let arm_your_bow : weapon = raise (Failure "NoMoreArrow")
let targeted_monster : target = raise (Failure "TooMuchFog")
let hit_monster : weapon -> target -> impacted =
    fun w t -> Impacted
```

On remarquera le manque d'expressivité de Java

# EXCEPTIONS

RAISE / THROW ARE UGLY GOTOS LOST IN TIME OF ASYNC / FUTURE

```
class Weapon {
    Weapon(){
        throw new Exception("NoMoreArrow");
    }
}
class Target {
    Target(){
        throw new Exception("TooMuchFog");
    }
}
class Impacted{}

public class Main {
    Weapon armYouBow = new Weapon();
    Target targetMonster = new Target();
    Impacted hitMonster(Weapon w, Target t) {
        return new Impacted();
    }
    FutureTask<Impacted> attack = new FutureTask(hitMonster(armYouBow, targetMonster));
}
```