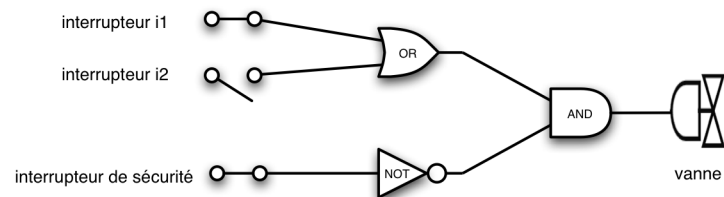


Polytech Lille IMA4 SC  
Conception Logicielle à Objets

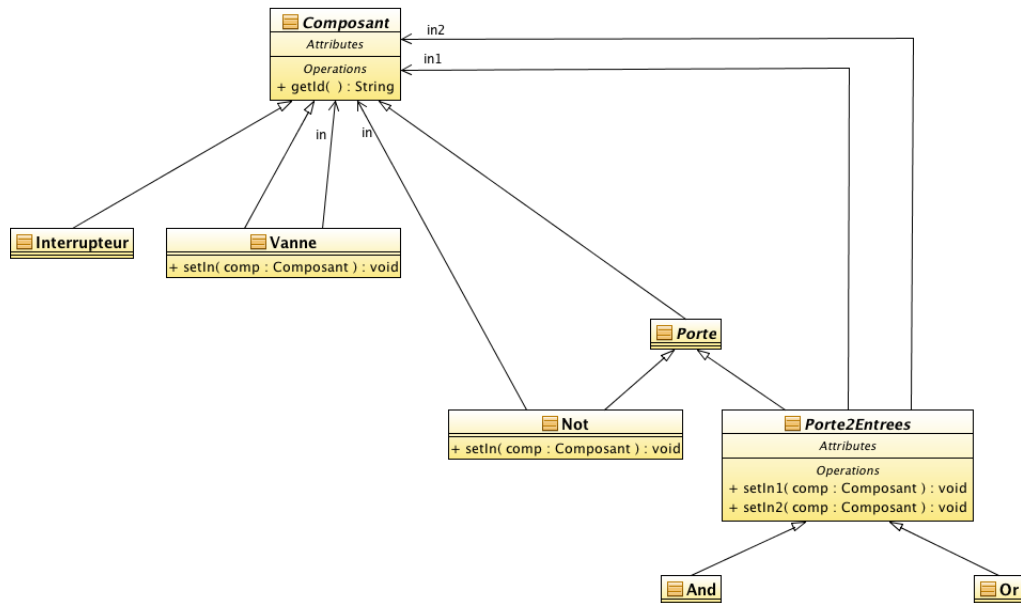
TP2

On considère des circuits constitués de portes logiques and, or ou not connectés en entrée sur des interrupteurs et en sortie sur des appareils. Par exemple :



## Composants

La hiérarchie de classes suivante permet de représenter les types de composants et sera complétée au fur et à mesure des TPs :



- `Composant` est la classe racine de la hiérarchie. Elle est abstraite et fournit une méthode `getId()` qui permet d'identifier le composant.
- un `Interrupteur` n'a pas d'entrée
- une `Vanne` est connectée en entrée sur un composant (in)
- `Porte` est la classe abstraite racine des portes logiques
- un `Not` a une connection en entrée sur un composant (in)
- `Porte2Entrees` est abstraite et factorise les portes logiques à 2 entrées (`And`, `Or`, ...) connectées en entrée sur 2 composants (in1 et in2)

On donne (`~bcarre/public_html/ima4-FSC/circuits/composants`) :

```
public abstract class Composant {
    public String getId() {
        return super.toString(); // class@numero renvoye par Object
    }
}

public class Interrupteur extends Composant {
}

public class Vanne extends Composant {
    protected Composant in;
    public void setIn(Composant comp) {
        in = comp;
    }
}

public abstract class Porte extends Composant {}

public class Not extends Porte {
    protected Composant in;
    public void setIn(Composant comp) {
        in = comp;
    }
}
```

## 1 Classes de portes

- Copier le répertoire  
`~bcarre/public_html/ima4-FSC/circuits/composants`  
 qui contient les classes précédentes et le squelette d'une classe `TestCircuits` (`main`).
- Ajouter les classes `Porte2Entrees`, `And` et `Or`.

### Test

- Tester en complétant la classe `TestCircuits` :
- Dans la section `//Construction` du `main`
    - créer un tableau `composants` de `Composant`
    - instancier les composants du circuit exemple et les ranger dans ce tableau
  - Section `//Connexions` inchangée pour l'instant
  - Programmer une méthode `printIds` paramétrée par un tableau de composants qui affiche leur `id`, et l'appliquer dans la section `//Affichage` du `main` sur le tableau `composants`.

## 2 Description des composants

Programmer une méthode (polymorphe) `public String description()` dans les classes de composants qui fournit une chaîne de caractères formée de :

- leur identifiant (donné par `getId()`)
- pour les composants disposant d'entrée(s), les identifiants (`getId()`) des composants correspondants ou la chaîne de caractères “`non connecte`” si l'entrée n'est pas connectée.

Par exemple pour un and (`And@48d6c16c`) non connecté en entrée 1, connecté en entrée 2 sur un not (`Not@5abb7465`) :

`And@48d6c16c in1: non connecte in2: Not@5abb7465`

### Test

Tester en complétant la classe `TestCircuits` :

- Dans la section `//Connexions`, établir quelques connexions (essayer différentes configurations)
- Programmer une méthode `descriptions` paramétrée par un tableau de composants qui affiche leur description, et l'appliquer dans la section `//Affichage` du `main` sur le tableau `composants`.