Un exemple: l'appel de méthodes en Java

On va regarder l'interaction entre deux traits de Java:

- L'*héritage* (sous-classes);
- Les *packages* (visibilité).

```
package P1;

public class A {
   ...
}

public class C
   extends P2.B {
   ...
}
```

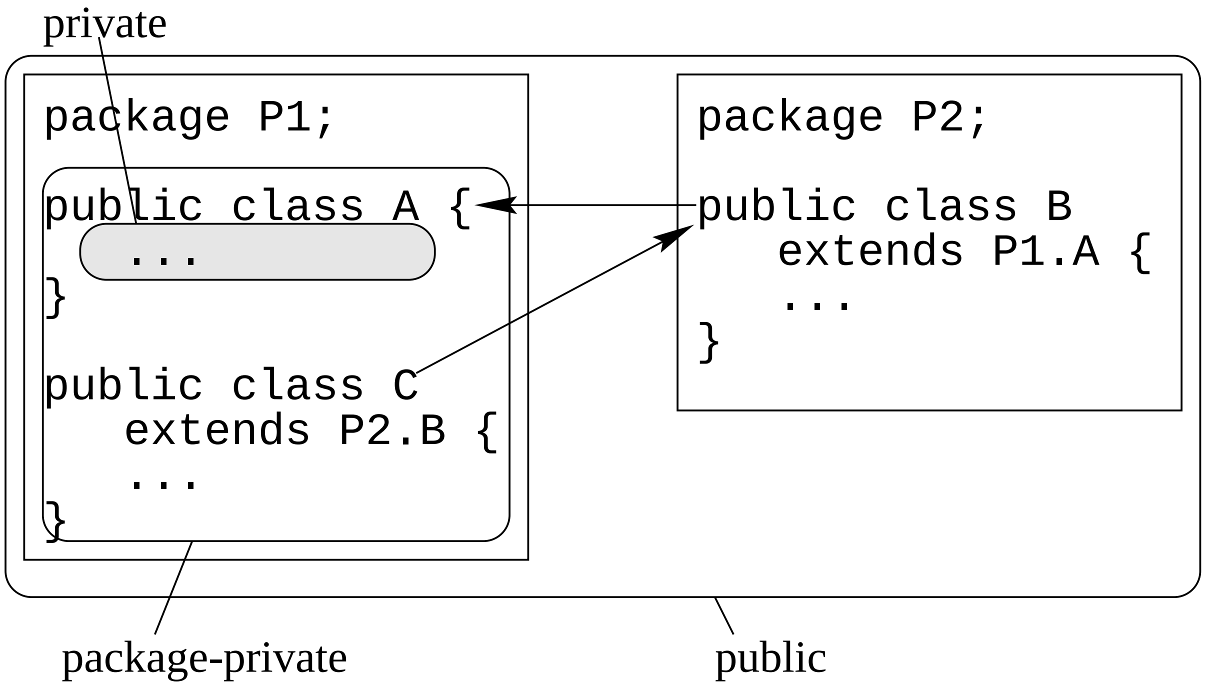
```
package P2;

public class B
   extends P1.A {
   ...
}
```

Les attributs de visibilité

- `public`: visible partout;
- `protected`: visible dans le package de définition et dans les sous-classes, mais pas ailleurs;
- par défaut, “package-private”: visible uniquement dans le package de définition;
- `private`: visible uniquement dans la classe de définition.

Les attributs de visibilité (2)



L'exemple de T. Thorn

```
package P1;

public class A {
  String foo () {
    return "A"; }
}
```

```
package P2;
import P1.A;
public class B extends P1.A {
  public String foo () {
    return "B"; }
}
```

```
package P1;
import P2.B;
public class C extends P2.B {
  public String foo () { return "C"; }
  public static void main (String [] args) {
    A a = new C ();
    B b = new C ();
    System.out.println (a.foo());
    System.out.println (b.foo());
  }
}
```

Quelques résultats

- Sun JVM 1.0: `C C`;
- Sun JVM 1.1: `C B`;
- Microsoft VJ++: `A C`.

Pourquoi ?