Programozási nyelvek és paradigmák

Információ elrejtése

Kozsik Tamás (2020)

Szelektív láthatóság

- Megadhatjuk, mely osztály(ok) számára látható
- Altípusa(i) számára is látható

```
create {A,B,C} make
feature {A,B} make
```

Szelektív láthatóság

- Megadhatjuk, mely osztály(ok) számára látható
- Altípusa(i) számára is látható

```
create {A,B,C} make
feature {A,B} make
```

- Finoman szabályozható függőségek!

Hagyományos láthatósági kategóriák

- nyilvános: `{ANY}` vagy semmi

```
create {ANY} make
feature make
```

- protected-szerű:

```
class C
feature {C}
    f ...
```

- titkos: `{}` vagy `{NONE}`
  - nem ugyanaz, mint a `private`
  - nem modulra, hanem objektumra privát
  - Scala: `private[this]` hasonlít
  - előnyök és hátrányok?

Titkos tag általában

- Ada, C++, Java...
- Fordítási egységre vagy aktuális osztályra korlátozott hozzáférés
- Leszármazott sem fér hozzá

```
class Base {
    private int v;
    int query( Base that ){ return that.v; }  // ok
}

class Sub extends Base {
    int mine(){ return this.v; }  // fordítási hiba
}
```

Titkos feature / create az Eiffelben

```
class BASE
feature {}
    v: INTEGER
feature
    query( other: BASE ): INTEGER
    do
        Result := other.v    -- fordítási hiba
    end
end

class SUB inherit BASE
feature
    mine: INTEGER
    do
        Result := v    -- ok
    end
end
```

Privát öröklődés

- Non-conforming
- Nem hoz létre altípus kapcsolatot
- Lehet `frozen` osztályból is

```
frozen class SEQUENCE[T]
  -- ommitted
end
```

```
class QUEUE[T]
inherit {NONE} SEQUENCE[T]
  -- ommitted
end
```

Láthatóság megváltoztatása öröklődés során

```
class QUEUE[T]
inherit {NONE} SEQUENCE[T]
    export   {ANY} hiext, lov, lorem, size;
           {QUEUE} all -- e.g. hirem, hiv, loext
    end
end
```

```
class DEQUE[T]
inherit QUEUE[T] export {ANY} hirem, hiv, loext end
end
```

LSP: kontravariáns láthatóság

- Altípusban bővebb hozzáférés

```
package java.lang;
public class Object {
    protected Object clone() throws CloneNotSupportedException
    ...
}

public class Point implements Cloneable {
    ...
    @Override public Point clone() { ... }
}
```

LSP: kontravariáns láthatóság

- Altípusban bővebb hozzáférés

```
package java.lang;
public class Object {
    protected Object clone() throws CloneNotSupportedException
    ...
}

public class Point implements Cloneable {
    ...
    @Override public Point clone() { ... }
}
```

- Paraméter típusa kontravariáns
- Visszatérési érték típusa kovariáns
- Kiváltható kivételek típusa kovariáns

Eiffel: kovariáns láthatóság is megengedett

- Lehet csökkenteni a láthatóságot altípusban
- A kovariáns láthatóság sérti az LSP-t
- CAT-probléma (Changed **Availability** or Type)

```
class SEQUENCE[T] ... end

class QUEUE[T]
inherit SEQUENCE[T]     -- altípus
    export   {ANY} hiext, lov, lorem, size;
           {QUEUE} all -- e.g. hirem, hiv, loext
    end
end
```

Eiffel: kovariáns láthatóság is megengedett

- Lehet csökkenteni a láthatóságot altípusban
- A kovariáns láthatóság sérti az LSP-t
- CAT-probléma (Changed **Availability** or Type)

```
class SEQUENCE[T] ... end

class QUEUE[T]
inherit SEQUENCE[T]     -- altípus
    export   {ANY} hiext, lov, lorem, size;
           {QUEUE} all -- e.g. hirem, hiv, loext
    end
end

s: SEQUENCE[INTEGER]
...
create {QUEUE[INTEGER]} s
s.loext(3)     -- polymorphic CAT-call
```

Attribútumok kívülről nem változtathatók

```
class CELL [T]
feature
    item: detachable T
end


c: CELL[INTEGER]
...
c.item := 1     -- fordítási hiba
```

Attribútumok kívülről nem változtathatók

```
class CELL [T]
feature
    item: detachable T
end


c: CELL[INTEGER]
...
c.item := 1     -- fordítási hiba
```

- Az objektum jogosult a saját attribútumait módosítani
- A titkos örökölteket is
- Más objektum csak kérheti (rutin meghívásával)
- Az *assign procedure* csak nyelvi fűszer

Módosítható (mutable) objektum

```
class CELL [T]
create
    put
feature
    item: T assign put
feature
    put( v: like item ) do item := v ensure item = v end
end


c: CELL[INTEGER]
...
create c.put(3);
c.put(5)
c.item := 1          -- ez is put-hívás
```

Módosíthatatlan (immutable) objektum

```
class CELL [T]
create
    put
feature
    item: T
feature {NONE}
    put( v: like item ) do item := v ensure item = v end
end


c: CELL[INTEGER]
...
create c.put(3);
c.put(5)                -- fordítási hiba
```

Nagyon nem Eiffel-stílusú

```
class CELL [T]
create
    put
feature
    get: T
        do
            Result := item
        end
    put( v: T )
        do
            item := v
        ensure
            item = v
        end
feature {NONE}
    item: T
end
```

Mutable STRING

```
name, nom: STRING
...
name := "apple"
nom = name
nom[1] := 'A'
print(name)  print("%N")
```

Asszociáció, de nem kompozíció

```
class PERSON
create
    make
feature
    name: STRING
feature {NONE}
    make( str: STRING )
        do
            name := str
        ensure
            name = str
        end
end
```

Asszociáció, de nem kompozíció

```
class PERSON
create
    make
feature
    name: STRING
feature {NONE}
    make( str: STRING )
        do
            name := str
        ensure
            name = str
        end
end

str: STRING
p: PERSON
...
str := "Buffalo Bill"
create p.make(str)
str[10] := 'u'          -- belső állapot kiszökött
```

Mutable típusú adattag is kinyírja az információ elrejtését

```
class PERSON
create
    make
feature
    name: STRING
feature {NONE}
    make( str: STRING )
        do
            name := str.twin
        ensure
            name ~ str
        end
end
```

Mutable típusú adattag is kinyírja az információ elrejtését

```
class PERSON
create
    make
feature
    name: STRING
feature {NONE}
    make( str: STRING )
        do
            name := str.twin
        ensure
            name ~ str
        end
end

p: PERSON
...
create p.make("Buffalo Bill")
p.name[10] := 'u'          -- belső állapot kiszökött
```