

Les classes internes

contenu de la section

LES CLASSES INTERNES.....1

CONTENU DE LA SECTION.....2

LES PRINCIPES ET UN PREMIER EXEMPLE.....3

les objectifs et les principes.....3

un premier exemple.....3

LE LIEN ENTRE LES INSTANCES4

l'accès aux membres de l'instance du conteneur.....4

QUELQUES COMPLÉMENTS SYNTAXIQUES.....5

l'instanciation directe des classes internes.....5

les classes internes static.....5

LES CLASSES ANONYMES.....6

la syntaxe.....6

QUELQUES COMPLÉMENTS SYNTAXIQUES.....7

divers.....7

Le lien vers l'instance de la classe externe.....7

Les principes et un premier exemple

les objectifs et les principes

- définir des classes pour des objets locaux (des structures pour usage interne par ex.)
 - limiter la visibilité des classes incluses à la classe «conteneur»
- lier les instances des classes incluses aux instances de la classe «conteneur»

un premier exemple

- possibilité de définir des classes internes protected, private, ...

```
class Memo {  
  
    class Enreg {  
        private int state;  
        private boolean locked;  
  
        Enreg(int s, boolean l) { state = s; locked = l; }  
        int getState() { return state; }  
        boolean getLocked() { return locked; }  
    }  
  
    private Hashtable memo = new Hashtable();  
  
    public void addInfo(String key, int s, boolean l) { memo.put( key, new Enreg( s, l) ; }  
    public int getState(String key) { Enreg e = (Enreg) memo.get(key); return (e != null) ? e.getState() : -1; }  
    public boolean getLocked(String key) { Enreg e = (Enreg) memo.get(key); return (e != null) ? e.getLocked() : false; }  
    .....  
}
```

Le lien entre les instances

l'accès aux membres de l'instance du conteneur

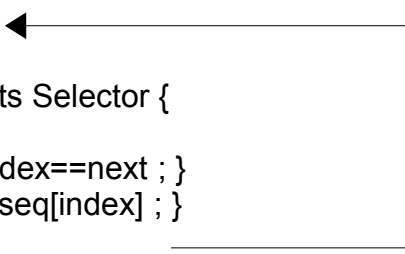
- les instances de classes internes sont liées à l'objet «conteneur» --> les classes incluses ont accès aux membres de l'objet «conteneur»

```
interface Selector {
    boolean end() ;
    Object current() ;
    void next() ;
}

class Sequence {
    private Object[] seq ;
    private int next =0 ;

    private class SSelector implements Selector {
        private int index=0 ;
        public boolean end() { return index==next ; }
        public Object current() { return seq[index] ; }
        public void next() { index++ ; }
    }

    public Sequence(int size) { seq=new Object[size] ; }
    public void add(Object o) { if (next<seq.length) { seq[next] = o ; next++ ; } }
    public Selector getSelector( ) { return new SSelector( ) ; }
}
```



```
class TestSequence {
    public static void main(String[] arg) {
        Sequence sRef=new Sequence(10) ;
        for (int i=0 ;i<5 ;i++) sRef.add(("string numero "+i)) ;
        Selector select=sRef.getSelector() ;
        while( !select.end()) {
            System.out.println((String) select.current()) ;
            select.next() ;
        }
    }
}
```

Quelques compléments syntaxiques

l'instanciation directe des classes internes

- le nom de la classe **InnerClass** interne à **OuterClass** est : **OuterClass.InnerClass**
- impossibilité de créer une instance d'une classe interne (non static) sans avoir l'objet de la classe externe qui la contient.
- la forme particulière de l'opérateur de création : **reference.new nomRelatif(...)**

```
class Bidon {  
    class BidonI1 {  
        private int val1=2 ;  
        public int getVal1( ) { return val1 ; }  
        public int setVal1(int v) { val1=v; }  
    }  
  
    private class BidonI2 {  
        private int val2;  
        public void setVal2(int v) { val2=v ; }  
        public int getVal2( ) { return val2; }  
    }  
}
```

```
public class TestBidon {  
    public static void main(String[ ] arg) {  
        Bidon bRef=new Bidon( ) ;  
        Bidon.BidonI1 bRefI1=bRef.new BidonI1( ) ;  
        System.out.println("valeur de val = "+ bRefI1.getVal1( ) ) ;  
        Bidon.BidonI2 bRefI2=bRef.new BidonI2( ) ;  
    }  
}
```

utiliser le nom «relatif»

interdit, BidonI2 est private !!

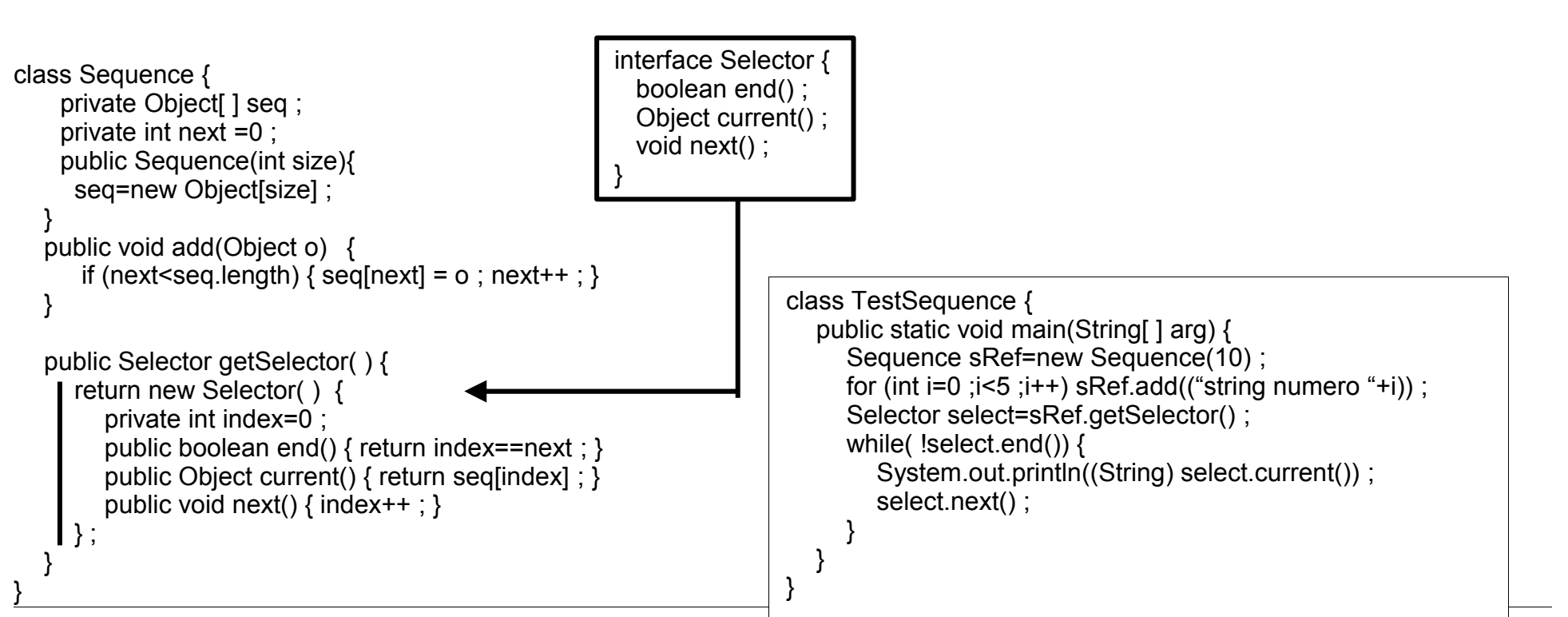
les classes internes static

- l'instance d'une classe interne static n'a pas de référence sur un objet conteneur
→ **possibilité d'instancier directement une classe interne static**

Les classes anonymes

la syntaxe

- il est possible de définir la classe incluse lors du new (sans lui associer de nom)



Quelques compléments syntaxiques

divers

- une classes interne peut être définie :
 - dans n'importe quel bloc
 - dans n'importe quelle expression (en particulier le passage d'argument de méthode)

```
.....  
button.addActionListener(  
    new ActionListener() {  
        public void actionPerformed(ActionEvent e) { System.out.println(e.getSource()) ; }  
    }  
);  
.....
```

Le lien vers l'instance de la classe externe

- désignation de la classe externe dans une instance de classe interne : this