Master 2 LITL - Programmation pour le TAL
Java : La classe `Object`, l'API et les listes

Franck Sajous/CLLE-ERSS

`http://fsajous.free.fr/`

« *Everything is Object* »

En Java, toute classe qui n'étend pas explicitement une autre classe étend directement la classe `Object`

« *Everything is Object* »

En Java, toute classe qui n'étend pas explicitement une autre classe étend directement la classe `Object`

```
class Point extends ObjetGraphique
{
    ...
}
```

« *Everything is Object* »

En Java, toute classe qui n'étend pas explicitement une autre classe étend directement la classe `Object`

```
class Point extends ObjetGraphique
{
    ...
}
```

- `Point` et `ObjetGraphique` sont des classes définies par l'utilisateur ;

« *Everything is Object* »

En Java, toute classe qui n'étend pas explicitement une autre classe étend directement la classe `Object`

```
class Point extends ObjetGraphique
{
    ...
}
```

- `Point` et `ObjetGraphique` sont des classes définies par l'utilisateur ;
- la classe `String` fait partie de l'API Java. Elle étend directement la classe `Object` ;

« *Everything is Object* »

En Java, toute classe qui n'étend pas explicitement une autre classe étend directement la classe `Object`

```
class Point extends ObjetGraphique
{
    ...
}
```

- `Point` et `ObjetGraphique` sont des classes définies par l'utilisateur ;
- la classe `String` fait partie de l'API Java. Elle étend directement la classe `Object` ;
- `Point` étend `ObjetGraphique` ;

« *Everything is Object* »

En Java, toute classe qui n'étend pas explicitement une autre classe étend directement la classe `Object`

```
class Point extends ObjetGraphique
{
    ...
}
```

- `Point` et `ObjetGraphique` sont des classes définies par l'utilisateur ;
- la classe `String` fait partie de l'API Java. Elle étend directement la classe `Object` ;
- `Point` étend `ObjetGraphique` ;
- `ObjetGraphique` étend (directement) la classe `Object` ;

« *Everything is Object* »

En Java, toute classe qui n'étend pas explicitement une autre classe étend directement la classe `Object`

```
class Point extends ObjetGraphique
{
    ...
}
```

- `Point` et `ObjetGraphique` sont des classes définies par l'utilisateur ;
- la classe `String` fait partie de l'API Java. Elle étend directement la classe `Object` ;
- `Point` étend `ObjetGraphique` ;
- `ObjetGraphique` étend (directement) la classe `Object` ;
- donc `Point` étend (indirectement) la classe `Object`.

« *Everything is Object* »

En Java, toute classe qui n'étend pas explicitement une autre classe étend directement la classe `Object`

```
class Point extends ObjetGraphique
{
    ...
}
```

- `Point` et `ObjetGraphique` sont des classes définies par l'utilisateur ;
- la classe `String` fait partie de l'API Java. Elle étend directement la classe `Object` ;
- `Point` étend `ObjetGraphique` ;
- `ObjetGraphique` étend (directement) la classe `Object` ;
- donc `Point` étend (indirectement) la classe `Object`.

API : doc de la classe `Object` → rien de bien intéressant, mais. . .

API vs. classes utilisateurs

- Jusqu'ici, nous avons défini toutes les classes que nous utilisions
- Java dispose d'un nombre important de classes regroupées dans une « API » (Application programming interface)
- Il s'agit +/- de classes « prêtes à l'emploi » regroupées en packages
- E.g. la paquet `java.io` contient des classes liées aux entrées/sorties (lecture et écriture de fichiers, notamment)
- `javax.swing` contient des classes servant à construire des IHM
- `java.util` contient des classes « structures de données » telles que des listes ordonnées
- liste des classes : `https://docs.oracle.com/javase/8/docs/api/`
- pour utiliser une classe d'un paquet donné de l'API dans une classe utilisateur :
  `import` *bigpackage.subpackage. ... . packageName.ClassName;*
  ou
  `import` *bigpackage.subpackage. ... . packageName.*;*
  (permet d'utiliser toutes les classes de *packageName*)

Exemple: la classe *String*

Chaînes constantes, objets `String`

- Chaînes constantes déjà rencontrées :
  `System.out.println ("Bonjour");`
  "Bonjour" : chaîne constante, codée « en dur » dans le programme.
- Utilisation d'objets de type `String` :

```
String chaine = new String ("Bonjour");
System.out.println (chaine);
System.out.println ("Longueur = " + chaine.length());
chaine = chaine.concat (" le monde");
System.out.println (chaine);
String chaine2 = chaine.replace ("jour", "ne nuit");
System.out.println (chaine2);

if (chaine1.equals (chaine2)) {... }
```

- voir la doc de l'API !

Les tableaux

Liste ordonnée, de taille fixe, d'éléments du même type.

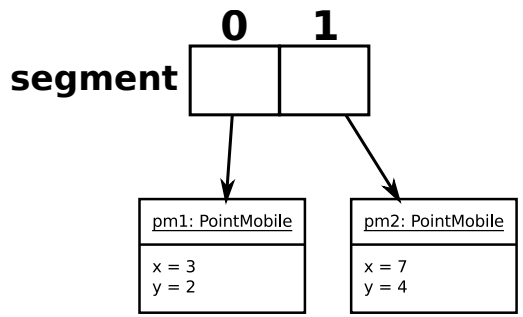
Syntaxe :

```
type nomtableau [];
type nomtableau [] = new type[taille];
```

- type : type des éléments que contiendra le tableau.
- taille : nombre d'éléments maximum que peut contenir le tableau.
- indice du premier élément du tableau : 0
- taille du tableau : `nom_tableau.length` (de type `int`)
- indice du dernier élément du tableau : `nom_tableau.length - 1`

Tableaux, exemples

```
int tableauEntiers[];
tableauEntiers = new int[5];

for (int i = 0; i < tableauEntiers.length; i++)
    tableauEntiers[i] = (i * i);

for (i = 0; i < tableauEntiers.length; i++)
    System.out.println ("Le carré de " + String.valueOf(i)
        + " est " + String.valueOf(tableauEntiers[i]));

double tableauFlottants[] = {3.2, 5.4, -6.458};
System.out.println (tableauFlottants.length); // affiche : 3

PointMobile segment = new PointMobile[2];
PointMobile pm1 = new PointMobile (3, 2);
tableauPoint[0] = pm1;
PointMobile pm2 = new PointMobile ();
tableauPoint[1] = pm2;
tableauPoint[1].setX (7);
tableauPoint[1].setY (4);
```



tableauEntier :

0	1	4	9	16

ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.size()); // affiche 1
System.out.println(tableau.contains(pm1)); //affiche false


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```

ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.size()); // affiche 1
System.out.println(tableau.contains(pm1)); //affiche false


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```

"on va se servir des classes de l'API contenues dans le paquet java.util"

ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

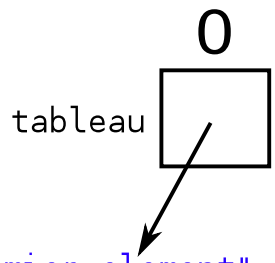
```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.size()); // affiche 1
System.out.println(tableau.contains(pm1)); //affiche false


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```

"on déclare et instancie tableau comme une variable de type ArrayList"

ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

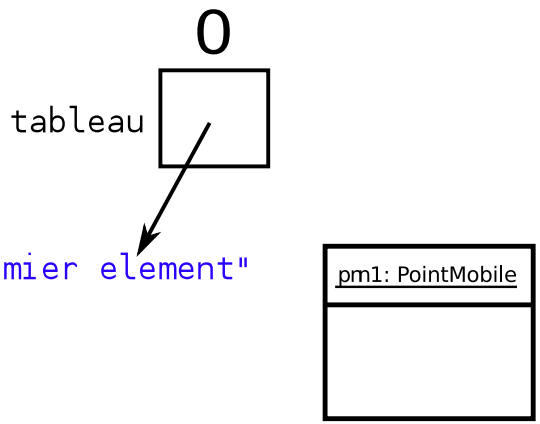
```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.size()); // affiche 1
System.out.println(tableau.contains(pm1)); //affiche false


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```



ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

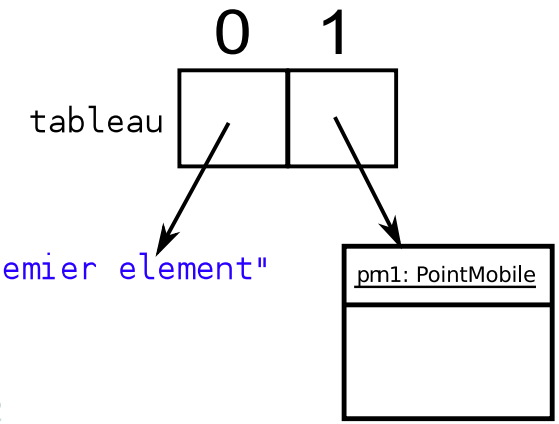
```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.size()); // affiche 1
System.out.println(tableau.contains(pm1)); //affiche false


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```



ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

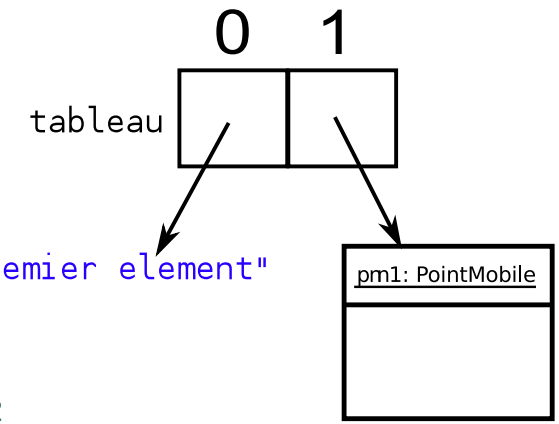
```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.size()); // affiche 1
System.out.println(tableau.contains(pm1)); //affiche false


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```



ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

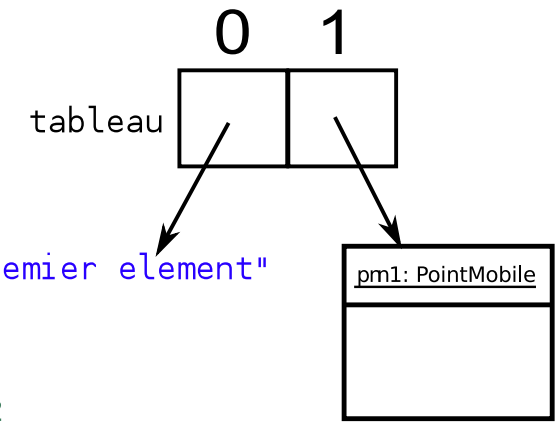
```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.size()); // affiche 1
System.out.println(tableau.contains(pm1)); //affiche false


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```



ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.size()); // affiche 1
System.out.println(tableau.contains(pm1)); //affiche false


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```



ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.
System.out.println(tableau.


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```

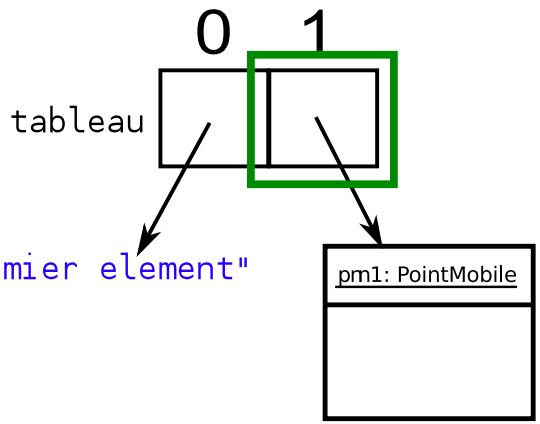


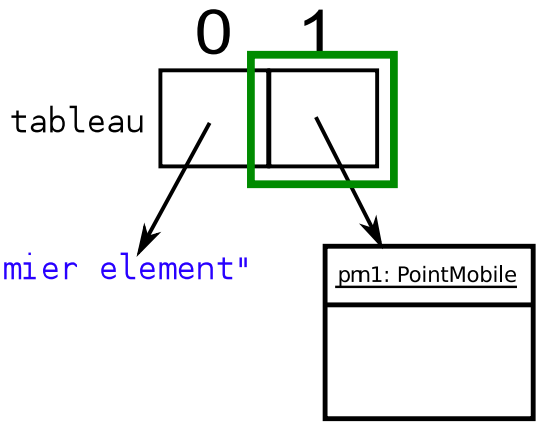
tableau.get(1)

deuxième élément du tableau

ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.
System.out.println(tableau.


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```



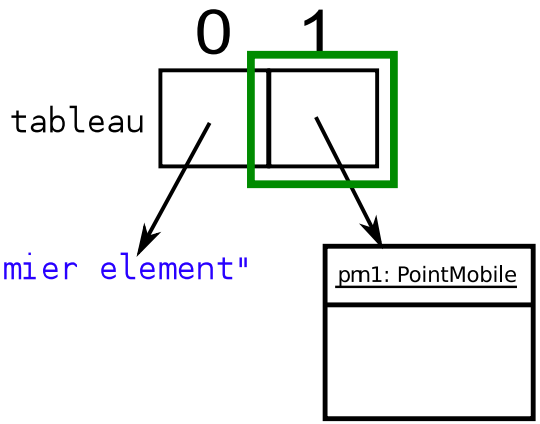
(PointMobile)tableau.get(1)

deuxième élément du tableau, dont on sait qu'il est de type PointMobile

ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);    (PointMobile)tableau.get(1).setNom(...)
System.out.println(tableau.
System.out.println(tableau.


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```

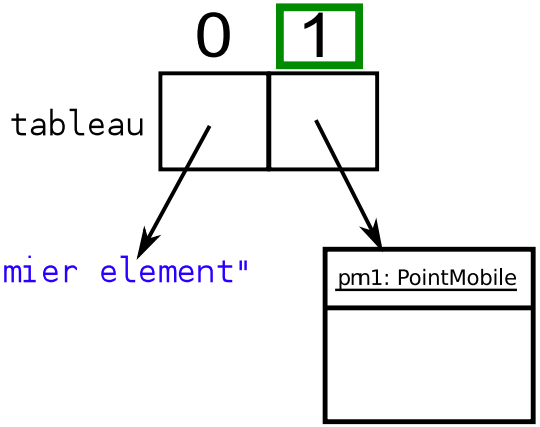


on peut donc lui appliquer les méthodes de cette classe

ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

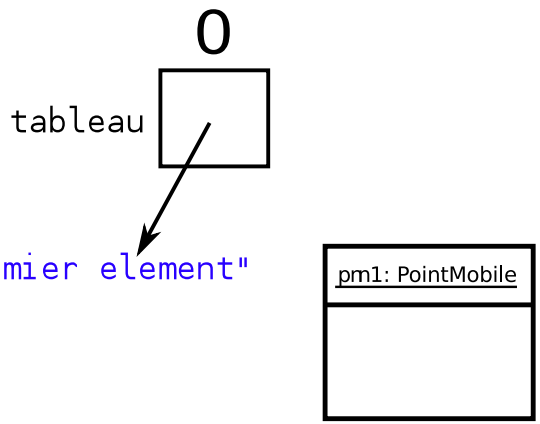
```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.size()); // affiche 1
System.out.println(tableau.contains(pm1)); //affiche false


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```



ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

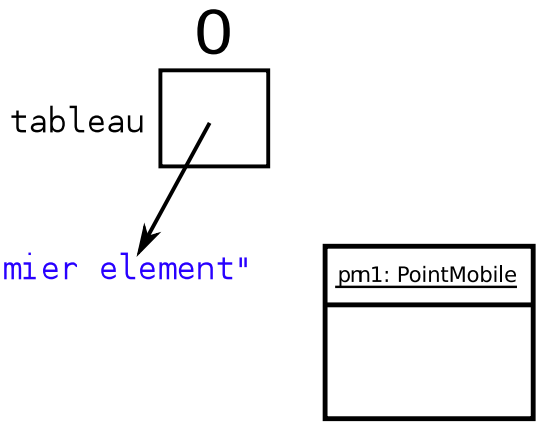
```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.size()); // affiche 1
System.out.println(tableau.contains(pm1)); //affiche false


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```



ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

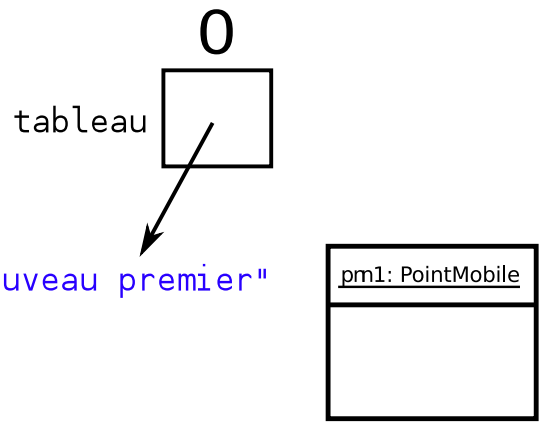
```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.size()); // affiche 1
System.out.println(tableau.contains(pm1)); //affiche false


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```



ArrayList

Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

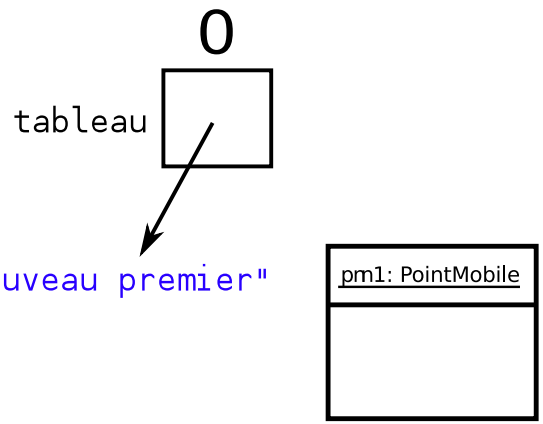
```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.size()); // affiche 1
System.out.println(tableau.contains(pm1)); //affiche false


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```



ArrayList

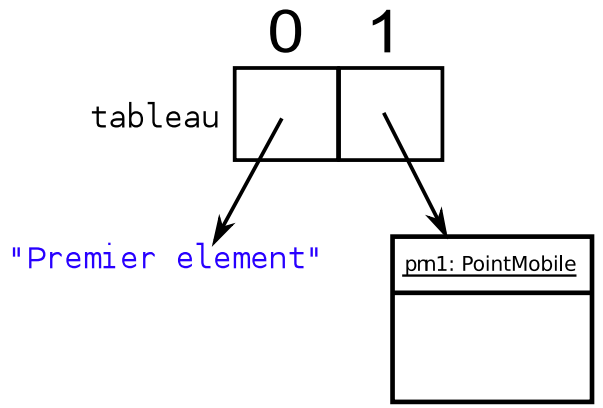
Tableaux (= liste ordonnée) de taille variable, d'objets de types potentiellement hétérogènes.

```
import java.util.*;

ArrayList tableau;
tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

System.out.println(tableau.size()); // affiche 2
System.out.println(tableau.get(0)); // affiche "Premier element"


((PointMobile) tableau.get(1)).setNom("PointM-1");
System.out.println(tableau.indexOf(pm1)); //affiche 1


tableau.remove(pm1);
System.out.println(tableau.size()); // affiche 1
System.out.println(tableau.contains(pm1)); //affiche false


tableau.set(0, "Nouveau premier");
System.out.println(tableau.get(0)); // affiche "Nouveau premier"
```



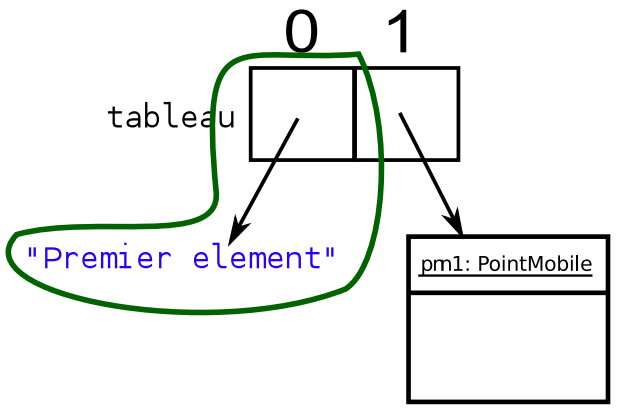
ArrayList, itérations

```
ArrayList tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

((PointMobile) tableau.get(1)).setNom("PointM-1");

for (int i = 0; i < tableau.size(); i++)
{
    Object obj = tableau.get(i);
    System.out.println("Indice " + i + " : "
            + obj.getClass().getName());
    System.out.println("Indice " + i + " : "
            + obj.toString());
}
```



ArrayList, itérations

```
ArrayList tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

((PointMobile) tableau.get(1)).setNom("PointM-1");

for (int i = 0; i < tableau.size(); i++)
{
    Object obj = tableau.get(i);
    System.out.println("Indice " + i + " : "
            + obj.getClass().getName());
    System.out.println("Indice " + i + " : "
            + obj.toString());
}
```



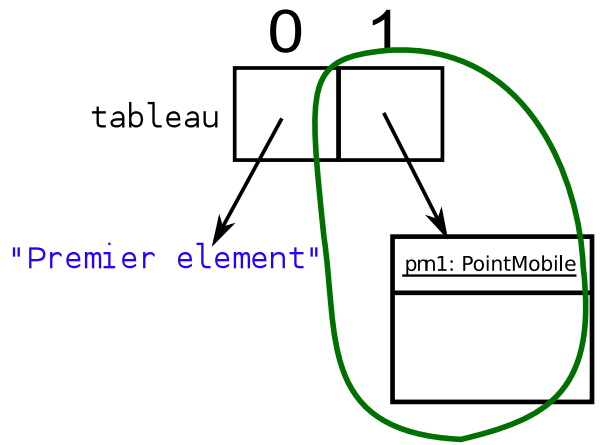
- première itération (i = 0) : affiche

  `Indice 0 : java.lang.String`

  `Indice 0 : Premier element`

ArrayList, itérations

```
ArrayList tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

((PointMobile) tableau.get(1)).setNom("PointM-1");

for (int i = 0; i < tableau.size(); i++)
{
    Object obj = tableau.get(i);
    System.out.println("Indice " + i + " : "
            + obj.getClass().getName());
    System.out.println("Indice " + i + " : "
            + obj.toString());
}
```



- première itération (i = 0) : affiche

  `Indice 0 : java.lang.String`

  `Indice 0 : Premier element`

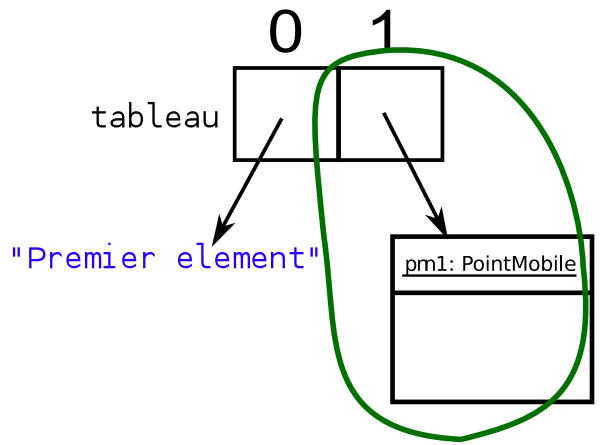
- deuième itération (i = 1) : affiche

  `Indice 1 : graph.PointMobile`

  `Indice 1 : [PointMobile-PointM-1] (0, 0)`

ArrayList, itérations

```
ArrayList tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

((PointMobile) tableau.get(1)).setNom("PointM-1");

for (int i = 0; i < tableau.size(); i++)
{
    Object obj = tableau.get(i);
    System.out.println("Indice " + i + " : "
            + obj.getClass().getName());
    System.out.println("Indice " + i + " : "
            + obj.toString());
}
```



- première itération (i = 0) : affiche

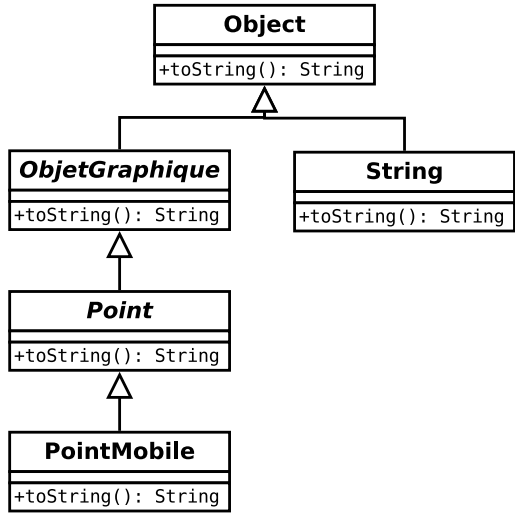
  `Indice 0 : java.lang.String`

  `Indice 0 : Premier element`

- deuième itération (i = 1) : affiche

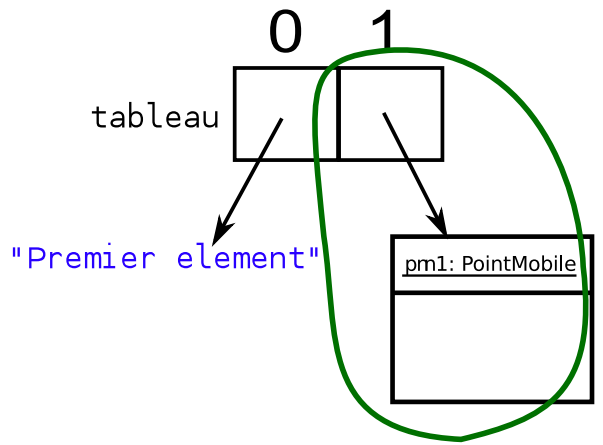
  `Indice 1 : graph.PointMobile`

  `Indice 1 : [PointMobile-PointM-1] (0, 0)`

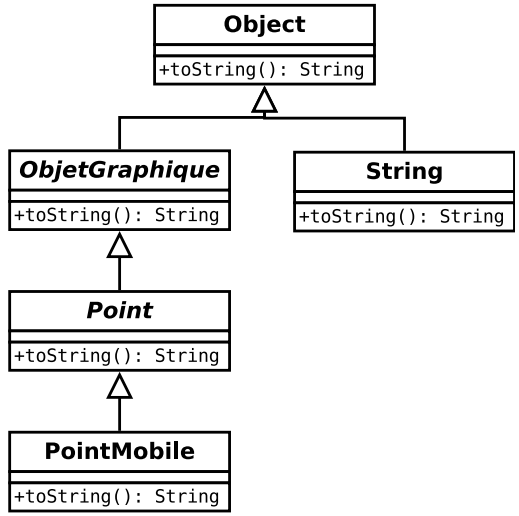


ArrayList, itérations

```
ArrayList tableau = new ArrayList();

tableau.add(new String ("Premier element"));
PointMobile pm1 = new PointMobile();
tableau.add(pm1);

((PointMobile) tableau.get(1)).setNom("PointM-1");

for (int i = 0; i < tableau.size(); i++)
{
    Object obj = tableau.get(i);
    System.out.println("Indice " + i + " : "
            + obj.getClass().getName());
    System.out.println("Indice " + i + " : "
            + obj.toString());
}
```



- première itération (i = 0) : affiche
  `Indice 0 : java.lang.String`
  `Indice 0 : Premier element`
- deuième itération (i = 1) : affiche
  `Indice 1 : graph.PointMobile`
  `Indice 1 : [PointMobile-PointM-1] (0, 0)`
- ce n'est pas la méthode `toString()` de la classe `Object` qui s'exécute, mais celle de la classe réelle de l'objet



ArrayList, itérateur

```
ArrayList tableau = new ArrayList();
...
```

ArrayList, itérateur

```
ArrayList tableau = new ArrayList();
```

...

Deux manières d'itérer sur une liste :

ArrayList, itérateur

```
ArrayList tableau = new ArrayList();
```

...

Deux manières d'itérer sur une liste :

```
for (int i = 0; i < tableau.size(); i++)
{
    Object obj = tableau.get(i);
    System.out.println(obj.toString());
}
```

ArrayList, itérateur

```
ArrayList tableau = new ArrayList();
```

...

Deux manières d'itérer sur une liste :

```
for (int i = 0; i < tableau.size(); i++)
{
    Object obj = tableau.get(i);
    System.out.println(obj.toString());
}

Iterator it = tableau.iterator();
while (it.hasNext())
{
    Object obj = it.next();
    System.out.println(obj.toString());
}
```

ArrayList « typés »

On peut spécifier le type des objets que l'on va mettre dans la liste :

```
ArrayList<String> tabChaines = new ArrayList<String>();

tabChaines.add("Premiere chaine");
tabChaines.add("Deuxieme chaine");

Iterator<String> it = tabChaines.iterator();
while (it.hasNext())
{
    String chaineCourante = it.next();
    System.out.println(chaineCourante.toUpperCase());
}
```

ArrayList « typés »

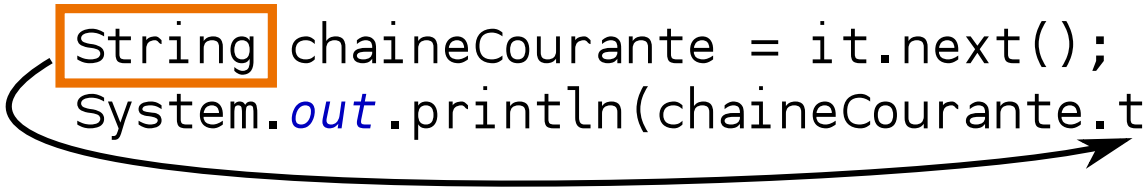
On peut spécifier le type des objets que l'on va mettre dans la liste :

```
ArrayList<String> tabChaines = new ArrayList<String>();

tabChaines.add("Premiere chaine");
tabChaines.add("Deuxieme chaine");

Iterator<String> it = tabChaines.iterator();
while (it.hasNext())
{
    String chaineCourante = it.next();
    System.out.println(chaineCourante.toUpperCase());
}
```

ArrayList « typés »

On peut spécifier le type des objets que l'on va mettre dans la liste :

```
ArrayList<String> tabChaines = new ArrayList<String>();

tabChaines.add("Premiere chaine");
tabChaines.add("Deuxieme chaine");

Iterator<String> it = tabChaines.iterator();
while (it.hasNext())
{
    String chaineCourante = it.next();
    System.out.println(chaineCourante.toUpperCase());
}
```



ArrayList d'objets

```
ArrayList tabChaines = new ArrayList();
```

un raccourci pour :

```
ArrayList<Object> tabChaines = new ArrayList<Object>();
```

ArrayList d'objets

```
ArrayList tabChaines = new ArrayList();
```

un raccourci pour :

```
ArrayList<Object> tabChaines = new ArrayList<Object>();
```

```
ArrayList tabChaines = new ArrayList();

tabChaines.add("Premiere chaine");
tabChaines.add("Deuxieme chaine");

Iterator it = tabChaines.iterator();
while (it.hasNext())
{
    String chaineCourante = (String) it.next();
    System.out.println(chaineCourante.toUpperCase());
}
```

ArrayList d'objets

```
ArrayList tabChaines = new ArrayList();
```

un raccourci pour :

```
ArrayList<Object> tabChaines = new ArrayList<Object>();
```

```
ArrayList tabChaines = new ArrayList();

tabChaines.add("Premiere chaine");
tabChaines.add("Deuxieme chaine");

Iterator it = tabChaines.iterator();
while (it.hasNext())
{
    String chaineCourante = (String) it.next();
    System.out.println(chaineCourante.toUpperCase());
}
```