Les Collections

- Nous avons déjà vu les structures de type tableau en JAVA. Ces structures efficaces pour stocker des éléments exigent de connaître à l'avance la taille du tableau et les insertions ou suppressions d'éléments ne sont pas possibles.

Pour répondre à cette problématique, Une solution performante consiste à utiliser des objets de type **ArrayList** ou **Vector**.

- Tableau `type[]` et `Array`
  - accès par index
  - recherche efficace si le tableau est trié (dichotomie)
  - insertions et suppressions peu efficaces
  - défaut majeur : nombre d'éléments borné
- Liste `interface List`
  - accès séquentiel : premier, suivant
  - insertions et suppressions efficaces
  - recherche lente, non efficace

- Tableau dynamique = tableau + liste `class ArrayList`

Déclaration d'un ArrayList

- La déclaration d'un ArrayList s'effectue de la façon suivante :

```
Vector <type variable> nom_vecteur = new Vector<type variable>();
ArrayList <type variable> nom_liste = new ArrayList<type variable>();
```

Exemple : on souhaite stocker dans une liste les noms des élèves et dans un vecteur les notes obtenues au BTS blanc.

```
/*  On commence par créer un ArrayList qu'on appellera par exemple eleves et qui contiendra des String */

List <String> eleves = new ArrayList<String>();

/*  On crée un Vector qu'on appellera par exemple notes et qui contiendra des integer */
```

```
Vector <Integer> population = new Vector<Integer>() ;
```

A ce stade, **eleves** est une liste vide, qui ne contient aucun élément.

La commande : `eleves.size(); produit 0`

Ajout d'un nouvel élément :

```
eleves.add("Dupont");
eleves.add("Wissen");
int n = eleves.size();// n= 2
```

Accéder à un élément de la liste :

```
String candidat = eleves.get(1);//candidat vaut Wissen
```

Insérer un élément :

Supposons qu'on veuille insérer un nouvel élève « Oxo » entre Dupont et Wissen :

Il suffit décrire :

```
eleves.add(1, "Oxo");
```

Supprimer un élément qui se trouve à l'indice i :

```
eleves.remove(i);
```

Pour parcourir une liste, on peut utiliser la boucle `for` classique ou le `for each` qui est simplifiée

```
for(String candidat : eleves) {   System.out.println(candidat);
}
```

Class ArrayList<E> : principales méthodes

- **boolean add(E)** ajoute l'élément spécifié à la fin de la liste
- **Void add(int index,element) insert un élément à l'indice index**
- **void clear()** : tout supprimer
- **.get(int index)** : retourne l'élément situé la position spécifiée
- **boolean contains(Object)** : test d'appartenance
- **boolean isEmpty()** : test de l'absence d'éléments
- **boolean remove(Object)** : retrait d'un élément
- **int size()** : nombre d'éléments