# Les tableaux d'objets

# Les tableaux unidimensionnels d'objets

- Un tableau peut contenir n'importe quel type de données. On peut donc créer des tableaux dont le type d'éléments est une classe. Ce sont des tableaux d'objets, par exemple un tableau de type Rectangle, un tableau de type Client et un tableau de type Produit.

- Déclaration et initialisation d'un tableau de 5 objets de type `Rectangle`

```
Rectangle[] tabRectangle1 = new Rectangle[5];
```

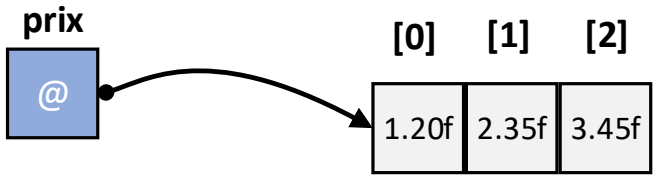
- Initialisation d'un tableau de type `Rectangle` en donnant ses éléments lors de sa déclaration

```
Rectangle[] tabRectangle2 = { new Rectangle(7, 3),
new Rectangle(4, 2), new Rectangle(10, 8) };
```

- Comme pour un tableau de type simple, une fois que l'opérateur `new` est exécuté, la taille d'un tableau d'objets est fixée et on ne peut plus la modifier.

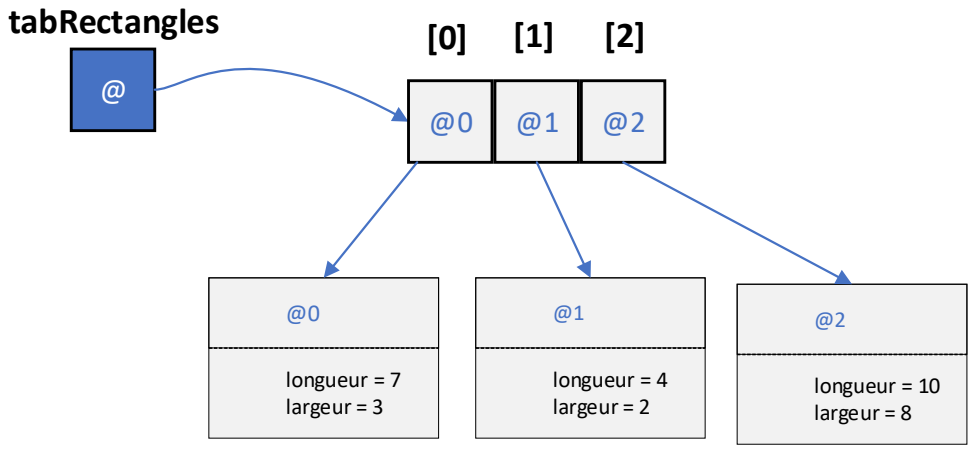
# Représentation en mémoire d'un tableau unidimensionnel d'objets

- Dans un tableau de type simple, chaque case contient une valeur de ce type.
- Exemple : `float[] prix = { 1.20f, 2.35f, 3.45f };`




- Par contre, dans un tableau d'objets, chaque case du tableau contient la référence vers un objet.

```
Rectangle[] tabRectangles = { new Rectangle(7, 3), new Rectangle(4, 2),
new Rectangle(10, 8) };
```

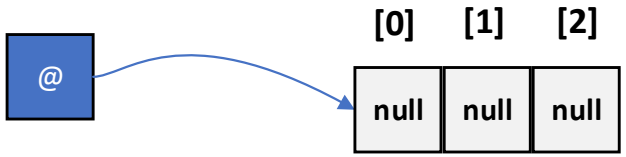


# Représentation en mémoire d'un tableau unidimensionnel d'objets

- L'opérateur `new` réserve le nombre de cases mémoires consécutives indiqué entre crochets. De plus, il initialise le tableau selon le type:

```
byte, short, int, long -> 0
float, double -> 0.0
boolean -> false
char      -> '\u0000'
objet quelconque (incluant String)-> null
```

- Pour un tableau d'objets, les éléments sont initialisés à null.
  - Exemple:
  - `Rectangle[] tabRectangle1 = new Rectangle[3];`



# Accès aux éléments d'un tableau unidimensionnel d'objets

- Pour faire appel à une méthode d'un objet se trouvant à la case d'indice i d'un tableau d'objets, on écrit `nomTableau[i].nomMethode()`.
- Exemple: pour avoir la largeur de l'objet qui se trouve à l'indice 2 de `tabRectangle`, on écrit:

```
tabRectangle[i].getLargeur();
```

# Exercice1

- Complétez la méthode main de la classe `Exercice1` fournie dans Moodle

```
public static void main(String[] args) {
// déclarer et initialiser un tableau de 3 rectangles nommé tabRectangle
//à compléter

// afficher la largeur du dernier rectangle
System.out.println("La largeur du dernier rectangle: " + "à compléter");

// modifier la longueur du  premier rectangle , la nouvelle longueur est 50
//à compléter

//  afficher le périmètre du premier rectangle
System.out.println("perimetre du premier rectangle: " +"à compléter");

// afficher les caractéristiques du premier rectangle
//à compléter

//Quel Rectangle a la plus grande superficie entre le premier et le deuxième
//à compléter
```

# Parcourir les éléments d'un tableau unidimensionnel d'objets

- Exemple: Parcourir le tableau `tabRecangle` Pour afficher ses éléments.

```
Rectangle[] tabRectangle = { new Rectangle(7, 3), new Rectangle(4, 2),
           new Rectangle(10, 8) };

for (int i = 0; i < tabRectangle.length; i++) {
      tabRectangle[i].afficher();

}

Ou

for (Rectangle unRectangle : tabRectangle) {
      unRectangle.afficher();

}
```

# Parcourir les éléments d'un tableau unidimensionnel d'objets

- Note: L"instruction d'affichage `System.out.println(tabRectangle[i])` affiche la référence de l'objet contenu dans la case i.

```
Rectangle[] tabRectangle = { new Rectangle(7, 3), new Rectangle(4, 2),
            new Rectangle(10, 8) };
for (int i = 0; i < tabRectangle.length; i++) {
   System.out.println(tabRectangle[i]);

}
```

```
<terminated> test [Java Application] C:\Program Files\Java
tabobjets.Rectangle@4617c264
tabobjets.Rectangle@36baf30c
tabobjets.Rectangle@7a81197d
```

# Exercice2

**Exercice2:** complétez la classe `Exercice2`

```
public class Exercice2 {
public static Scanner clavier = new Scanner(System.in);
public static void main(String[] args) {
   Rectangle[] tabRectangle = new Rectangle[3];
   // Appelez initialiserTabRectangle pour initialiser (remplir) tabRectangle
   // Appelez afficherTabRectangle pour afficher les elements de tabRectangle
   // Affichez l'aire totale des rectangles: appelez aireTotale(tabRectangle)
}
// Initialise les elements du tableau fourni en parametre avec des données entrees au clavier.
public static void initialiserTabRectangle(Rectangle[] tabRectangle) {
   int larg, longu;
   for (int i = 0; i < tabRectangle.length; i++) {
      System.out.print("Entrez la longueur du rectangle #" + (i+1) + ": ");
      longu = clavier.nextInt();
      System.out.print("Entrez la largeur du rectangle #" + (i+1) + ": ");
      larg = clavier.nextInt();
      //à completer
   }
}
//Affiche les éléments du tableau
public static void afficherTabRectangle(Rectangle[] tabRectangle) {
    //à completer
}
//Retourne l'aire totale des rectangles du tableau fourni en paramètre
public static int aireTotale(Rectangle[] tabRectangle) {
     //à completer
}
```

# Tableau à 2 dimensions d'objets

- Syntaxe de déclaration et d'initialisation d'un tableau à deux dimensions d'objets.

```
NomType[][] nomTableau = new NomType[NB_LIGNES][NB_COLONNES]
```

Exemple: un tableau de type Rectangle de 3 lignes et 2 colonnes.

```
Rectangle[][] tab2DRectangle1 = new Rectangle[3][2];
```

- Initialisation d'un tableau 2D de type `Rectangle` en donnant ses éléments lors de la déclaration.

```
Rectangle[][] tab2DRectangle2 = {
              { new Rectangle(8, 2), new Rectangle(7, 5) },
              { new Rectangle(10, 3), new Rectangle(3, 1) },
              { new Rectangle(5, 3), new Rectangle(5, 2) } };
}
```

# Parcourir un tableau à 2 dimensions d'objets

- Pour parcourir un tableau à deux dimensions, on utilise 2 boucles imbriquées.

```
NomType[][] nomTableau = new NomType[NB_LIGNES][NB_COLONNES];
//parcourir les lignes
for (int ligne = 0; ligne < nomTableau.length; ligne++) {
    //parcourir les colonnes
    for (int colonne = 0; colonne < nomTableau[ligne].length; colonne++) {
         tabRectangle[ligne][colonne].afficher();
     }
}
```

- Note: généralement, on utilise le nom de la variable `i` pour une `ligne` et le nom `j` pour une `colonne`.

# Parcourir un tableau à 2 dimensions d'objets

```
public class Tableau2DRectangle {
public static void main(String[] args) {
   Rectangle[][] tab2DRectangle1 = new Rectangle[3][2];
   Rectangle[][] tab2DRectangle2 = {
           { new Rectangle(8, 2), new Rectangle(7, 5) },
           { new Rectangle(10, 3), new Rectangle(3, 1) },
           { new Rectangle(5, 3), new Rectangle(5, 2) }
   };
   affichertabRectangle(tab2DRectangle2);
}

public static void affichertabRectangle(Rectangle[][] tabRectangle) {
  for (int i = 0; i < tabRectangle.length; i++) {
     for (int j = 0; j < tabRectangle[i].length; j++) {
        tabRectangle[i][j].afficher();
     }
  }
}
}
```