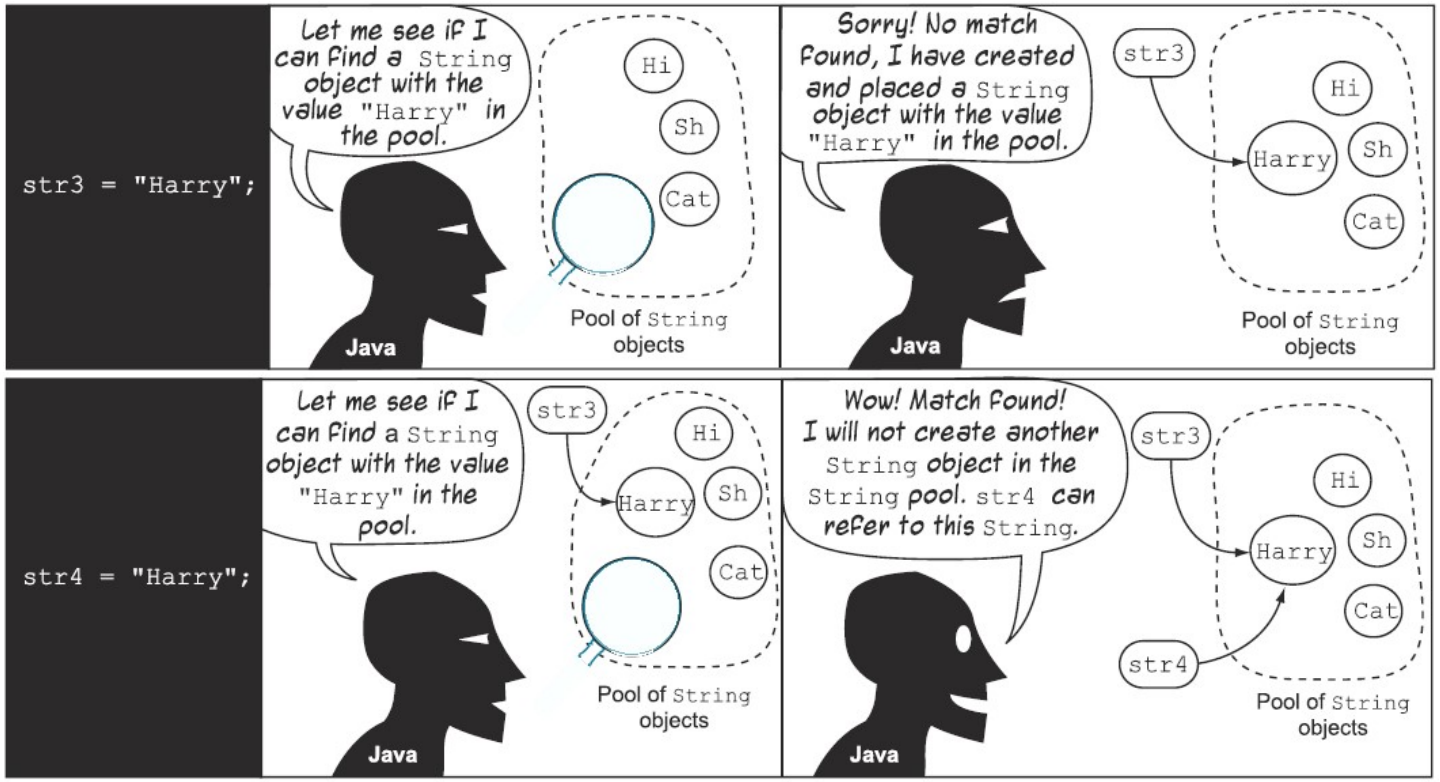
Java API

Java Strings

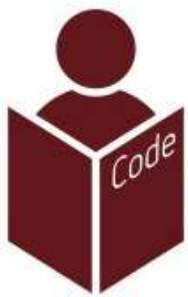
- Αλληλουχία χαρακτήρων
- Δήλωση μέσα σε " "
- Κάθε String είναι ένα αντικείμενο
- Εκτός από τον συνηθισμένο τρόπο δήλωσης ενός αλφαριθμητικού (String s = "Hello";) , υπάρχουν πάρα πολλοί ακόμα
- Ο σωστός χειρισμός των Strings μπορεί να βοηθήσει:
  - Σε αποφυγή λαθών (π.χ. σύγκρισης)
  - Στην βελτιστοποίηση του κώδικα!
  - Σε περιβάλλοντα multithreading

String pool

- Για λόγους βελτιστοποίησης η Java δημιουργεί μια «δεξαμενή» από Strings και επανα-χρησιμοποιεί όσα βρίσκονται εκεί μέσα



Διαφορετικοί τρόποι δημιουργίας String




```
String girl = new String("Shreya");
char[] name = new char[]{'P','a','u','l'};
String boy = new String(name);
```

String constructor that accepts a String

String constructor that accepts a char array

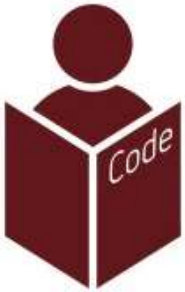
```
StringBuilder sd1 = new StringBuilder("String Builder");
String str5 = new String(sd1);
StringBuffer sb2 = new StringBuffer("String Buffer");
String str6 = new String(sb2);
```

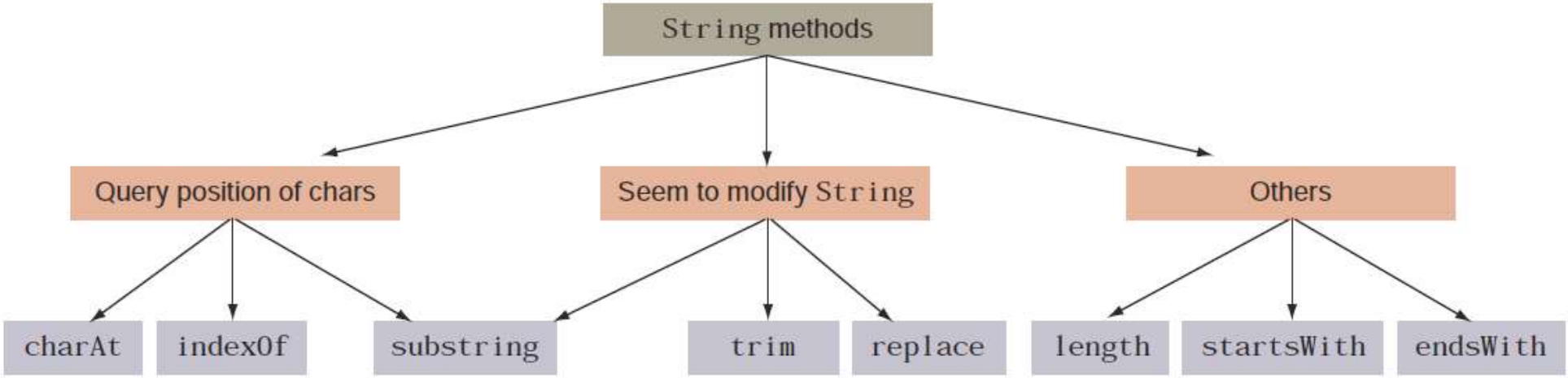
String constructor that accepts object of StringBuilder

String constructor that accepts object of StringBuffer

- Ανεξαρτήτως της ύπαρξης του String Pool, εάν δημιουργήσετε String με το keyword new, θα δημιουργηθεί νέο String

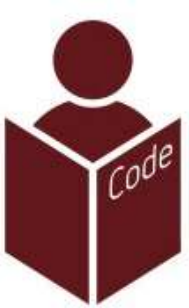
String handling methods





Σημαντικά!

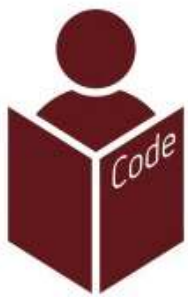
- String is Immutable
- Οι μέθοδοι της τάξης String δεν μεταβάλουν το περιεχόμενο του String



Σύγκριση Αλφαριθμητικών

- Χρειάζεται πολύ μεγάλη προσοχή. Γίνονται πολλά λάθη και από φοιτητές και από προγραμματιστές
- Ο τελεστής == χρησιμοποιείται για τη σύγκριση των διευθύνσεων σε reference variables (όπως είναι αναμενόμενο)
- Αρκετές φορές ο τελεστής == μπορεί να μας δώσει ισότητα αλφαριθμητικών, αλλά αυτό συμβαίνει λόγω της χρήσης του String Pool. Πρέπει να αποφεύγεται!
- Για τη σύγκριση της πραγματικής «τιμής» του αλφαριθμητικού ενδείκνυται η χρήση της μεθόδου equals()

Προσοχή!




```
String var3 = "code";
String var4 = "code";
System.out.println(var3.equals(var4));
System.out.println(var3 == var4);

String var1 = new String("Java");
String var2 = new String("Java");
System.out.println(var1.equals(var2));
System.out.println(var1 == var2);
```

Prints true

Prints true

Prints true

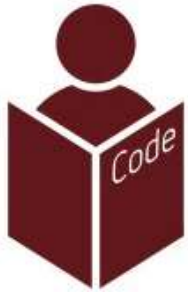
Prints false

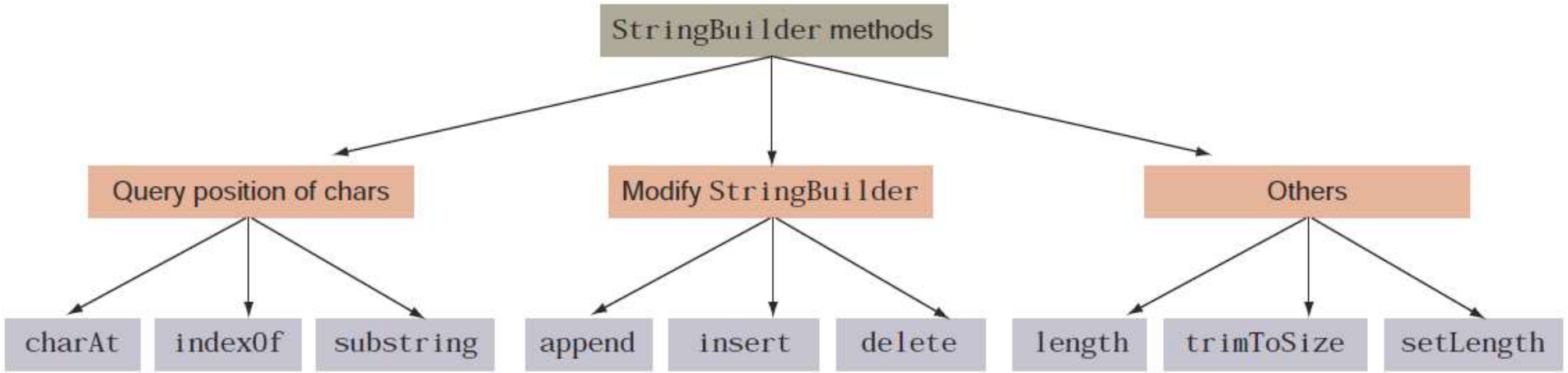
- (Άσκηση) Ελέγξτε αν τα αλφαριθμητικά που επιστρέφουν οι μέθοδοι της τάξης String αποθηκεύονται στο String Pool

Mutable Strings

- Μια από τις πιο βασικές τάξεις: StringBuilder
- Ανήκει στο package: java.lang
- Χρησιμοποιείται συχνά όταν έχουμε μεγάλα Strings ή συχνές αλλαγές πάνω σε ένα String
- Στην προηγούμενη περίπτωση μπορούμε να έχουμε σημαντικές βελτιώσεις στην «επίδοση» του κώδικά μας
- Ο constructor της τάξης δέχεται ένα μεγάλο αριθμό από overloads, οπότε μπορεί να χρησιμοποιηθεί με πολλούς και διαφορετικούς τρόπους!

StringBuilder Methods





StringBuilder Vs StringBuffer

- Η τάξη StringBuffer προσφέρει περίπου την ίδια λειτουργικότητα με την τάξη StringBuilder
- Η βασική διαφορά έγκειται στο γεγονός ότι οι μέθοδοι της τάξης StringBuffer είναι synchronized και ως αποτέλεσμα «thread safe»
- Ωστόσο, η χρήση synchronized μεθόδων συνοδεύεται από ένα αναμενόμενο πρόσθετο προγραμματιστικό κόστος

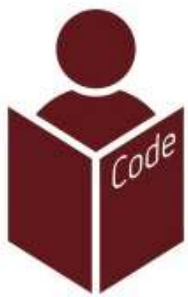
Java ArrayList

- Η τάξη ArrayList είναι μια από τις πιο γνωστές και ευρέως χρησιμοποιούμενες τάξεις της Java (και άλλως γλωσσών βέβαια)
- Προσπαθεί να συνδυάσει τα καλύτερα χαρακτηριστικά από τον κόσμο των πινάκων (Arrays) και των λιστών (Lists)
- Έχει μεταβλητό μέγεθος
- Γνωστές/Βασικές λειτουργίες:
  - Προσθήκη στοιχείων στη λίστα
  - Διαγραφή στοιχείων από τη λίστα
  - Αλλαγή στοιχείων της λίστας
  - Προσπέλαση όλων των στοιχείων της λίστας

ArrayList Properties

- It implements the interface `List`.
- It allows `null` values to be added to it.
- It implements all list operations (add, modify, and delete values).
- It allows duplicate values to be added to it.
- It maintains its insertion order.
- You can use either `Iterator` or `ListIterator` to iterate over the items of an `ArrayList`.
- It supports generics, making it type safe. (You have to declare the type of the elements that should be added to an `ArrayList` with its declaration.)

ArrayList Creation

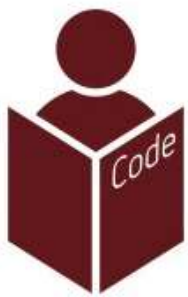



```
import java.util.ArrayList;
public class CreateArrayList {
    public static void main(String args[]) {
        ArrayList<String> myArrList = new ArrayList<String>();
    }
}
```

1 Import java.util.ArrayList

2 Declare an ArrayList object

ArrayList – Add Items

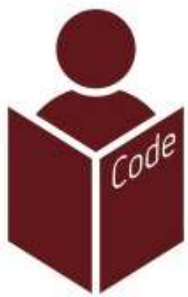



```
import java.util.ArrayList;
public class AddToArrayList {
    public static void main(String args[]) {
        ArrayList<String> list = new ArrayList<>();
         list.add("one");
         list.add("two");
         list.add("four");
         list.add(2, "three");
    }
}
```

1 Add element at the end

2 Add element at specified position

ArrayList – Item Iteration




```
import java.util.ArrayList;
public class AccessArrayList {
    public static void main(String args[]) {
        ArrayList<String> myArrList = new ArrayList<>();
        myArrList.add("One");
        myArrList.add("Two");
        myArrList.add("Four");
        myArrList.add(2, "Three");
        for (String element : myArrList) {
            System.out.println(element);
        }
    }
}
```

1 **Code to access ArrayList elements**

ArrayList – Item Iteration v2 using ListIterator


```
import java.util.ArrayList;
import java.util.ListIterator;
public class AccessArrayListUsingListIterator {
    public static void main(String args[]) {
        ArrayList<String> myArrList = new ArrayList<String>();
        myArrList.add("One");
        myArrList.add("Two");
        myArrList.add("Four");
        myArrList.add(2, "Three");
        ListIterator<String> iterator = myArrList.listIterator();
        while (iterator.hasNext()) {
            System.out.println(iterator.next());
        }
    }
}
```

1 Get the iterator

2 Use hasNext() to check whether more elements exist

3 Call next() to get the next item from iterator

Άσκηση

- Με τους παραπάνω 2 τρόπους και ίσως δοκιμάζοντας και κάποιον ακόμα (π.χ. με απλό loop?) δείτε αν μπορείτε να τροποποιήσετε (αλλαγή – προσθήκη – διαγραφή) τα στοιχεία που βρίσκονται εντός ενός ArrayList κατά τη διάρκεια των επαναλήψεων!
- Καταγράψτε τι παρατηρείτε!

ArrayList - addAll() method

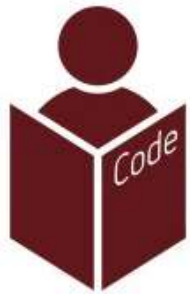

```
ArrayList<String> myArrList = new ArrayList<String>();
myArrList.add("One");
myArrList.add("Two");
ArrayList<String> yourArrList = new ArrayList<String>();
yourArrList.add("Three");
yourArrList.add("Four");
myArrList.addAll(1, yourArrList);
for (String val : myArrList)
    System.out.println(val);
```

Add elements of yourArrList to myArrList

ArrayList – Access elements

- `get(int index)`—This method returns the element at the specified position in this list.
- `size()`—This method returns the number of elements in this list.
- `contains(Object o)`—This method returns `true` if this list contains the specified element.
- `indexOf(Object o)`—This method returns the index of the first occurrence of the specified element in this list, or `-1` if this list doesn't contain the element.
- `lastIndexOf(Object o)`—This method returns the index of the last occurrence of the specified element in this list, or `-1` if this list doesn't contain the element.

Παραδείγματα




```
ArrayList<StringBuilder> myArrList =
                                    new ArrayList<StringBuilder>();
StringBuilder sb1 = new StringBuilder("Jan");
StringBuilder sb2 = new StringBuilder("Feb");
myArrList.add(sb1);
myArrList.add(sb2);
myArrList.add(sb2);
System.out.println(myArrList.contains(new StringBuilder("Jan")));
System.out.println(myArrList.contains(sb1));
System.out.println(myArrList.indexOf(new StringBuilder("Feb")));
System.out.println(myArrList.indexOf(sb2));
System.out.println(myArrList.lastIndexOf(
                                          new StringBuilder("Feb")));
System.out.println(myArrList.lastIndexOf(sb2));
}
```

Adds sb1 to the ArrayList

Adds sb2 to the ArrayList

Adds sb2 to the ArrayList again

Prints false

Prints true

Prints –1

Prints 1

Prints –1

Prints 2

Άσκηση

- Ελέγξτε τη χρήση των 2 παρακάτω μεθόδων της τάξης ArrayList:
  - toArray()
  - clone()