Java ArrayList and String Cheat Sheet

ArrayList Reference

Import Statement

java

```java
import java.util.ArrayList;
import java.util.Collections;
```

Declaration and Initialization

java

```java
// Generic ArrayList (recommended)
ArrayList<String> list = new ArrayList<>();
ArrayList<Integer> numbers = new ArrayList<>();

// With initial capacity
ArrayList<String> list = new ArrayList<>(10);

// Initialize with values
ArrayList<String> list = new ArrayList<>(Arrays.asList("a", "b", "c"));
```

Adding Elements

java

```java
list.add("element");        // Add to end
list.add(0, "first");       // Add at specific index
list.addAll(otherList);     // Add all elements from another list
list.addAll(1, otherList);  // Add all at specific index
```

Accessing Elements

java

```java
String item = list.get(0);   // Get element at index
String first = list.get(0);  // First element
String last = list.get(list.size() - 1); // Last element
```

Modifying Elements

java

```java
list.set(0, "newValue");     // Replace element at index
```

Removing Elements

java

```java
list.remove(0);            // Remove by index
list.remove("element");     // Remove by value (first occurrence)
list.removeAll(otherList);   // Remove all occurrences
list.clear();              // Remove all elements
```

Size and Capacity

java

```java
int size = list.size();      // Number of elements
boolean empty = list.isEmpty(); // Check if empty
list.trimToSize();          // Reduce capacity to current size
```

Searching and Checking

java

```java
boolean contains = list.contains("item");    // Check if contains
int index = list.indexOf("item");          // First occurrence (-1 if not found)
int lastIndex = list.lastIndexOf("item");    // Last occurrence
```

Iteration

java

```java
// Enhanced for loop (recommended)
for (String item : list) {
    System.out.println(item);
}

// Traditional for loop
for (int i = 0; i < list.size(); i++) {
    System.out.println(list.get(i));
}

// Iterator
Iterator<String> it = list.iterator();
while (it.hasNext()) {
    System.out.println(it.next());
}

// forEach with lambda (Java 8+)
list.forEach(System.out::println);
```

Conversion

java

```java
// To Array
String[] array = list.toArray(new String[0]);

// From Array
List<String> list = Arrays.asList(array);
ArrayList<String> arrayList = new ArrayList<>(Arrays.asList(array));
```

Sorting and Manipulating

java

```java
Collections.sort(list);              // Natural order
Collections.sort(list, Collections.reverseOrder()); // Reverse order
Collections.shuffle(list);           // Randomize order
Collections.reverse(list);           // Reverse current order
```

Sublist Operations

java

```java
List<String> sublist = list.subList(1, 4); // Elements from index 1 to 3
ArrayList<String> copy = new ArrayList<>(list); // Create copy
```

String Reference

Declaration and Initialization

java

```java
String str = "Hello World";
String str2 = new String("Hello");
String empty = "";
String nullStr = null;
```

Basic Properties

java

```java
int length = str.length();      // Get string length
boolean empty = str.isEmpty();  // Check if empty
boolean blank = str.isBlank();  // Check if blank (Java 11+)
```

Character Access

java

```java
char ch = str.charAt(0);        // Get character at index
char[] chars = str.toCharArray(); // Convert to char array
```

Case Operations

java

```java
String upper = str.toUpperCase(); // Convert to uppercase
String lower = str.toLowerCase(); // Convert to lowercase
```

Trimming and Stripping

java

```java
String trimmed = str.trim();      // Remove leading/trailing whitespace
String stripped = str.strip();    // Remove leading/trailing whitespace (Java 11+)
String stripLeading = str.stripLeading();  // Remove leading whitespace (Java 11+)
String stripTrailing = str.stripTrailing(); // Remove trailing whitespace (Java 11+)
```

Substring Operations

java

```java
String sub = str.substring(5);    // From index 5 to end
String sub2 = str.substring(0, 5); // From index 0 to 4 (exclusive)
```

Search Operations

java

```java
boolean contains = str.contains("Hello");      // Check if contains substring
boolean startsWith = str.startsWith("Hello");  // Check if starts with
boolean endsWith = str.endsWith("World");      // Check if ends with
int index = str.indexOf("o");                  // First occurrence of character
int index2 = str.indexOf("llo");               // First occurrence of substring
int lastIndex = str.lastIndexOf("o");          // Last occurrence
```

Comparison

java

```java
boolean equal = str.equals("Hello");          // Case-sensitive comparison
boolean equalIgnoreCase = str.equalsIgnoreCase("hello"); // Case-insensitive
int comparison = str.compareTo("Hello");      // Lexicographic comparison
int comparisonIgnoreCase = str.compareToIgnoreCase("hello");
```

Replacement

java

```java
String replaced = str.replace("o", "0");      // Replace all occurrences
String replacedFirst = str.replaceFirst("o", "0"); // Replace first occurrence
String replacedRegex = str.replaceAll("\\d", "X"); // Replace using regex
```

Splitting and Joining

java

```java
String[] parts = str.split(" ");           // Split by space
String[] parts2 = str.split("\\s+");       // Split by whitespace (regex)
String[] limited = str.split(" ", 2);      // Limit splits to 2

// Joining (Java 8+)
String joined = String.join(", ", "a", "b", "c");
String joinedList = String.join("-", Arrays.asList("x", "y", "z"));
```

String Building (for multiple concatenations)

java

```java
StringBuilder sb = new StringBuilder();
sb.append("Hello");
sb.append(" ");
sb.append("World");
String result = sb.toString();

// Method chaining
String result2 = new StringBuilder()
    .append("Hello")
    .append(" ")
    .append("World")
    .toString();
```

Formatting

java

```java
String formatted = String.format("Hello %s, you are %d years old", "John", 25);
String formatted2 = String.format("%.2f", 3.14159); // 3.14

// Text blocks (Java 15+)
String textBlock = """
    This is a
    multi-line
    string
    """;
```

Conversion

java

```java
// To other types
int num = Integer.parseInt("123");
double d = Double.parseDouble("3.14");
boolean b = Boolean.parseBoolean("true");

// From other types
String fromInt = String.valueOf(123);
String fromDouble = String.valueOf(3.14);
String fromBoolean = String.valueOf(true);
String fromChar = String.valueOf('A');
```

Utility Methods

java

```java
String repeated = str.repeat(3);          // Repeat string n times (Java 11+)
Stream<String> lines = str.lines();       // Get stream of lines (Java 11+)
String indented = str.indent(4);          // Add indentation (Java 12+)
```

Common Patterns

Convert ArrayList to String

java

```java
// Using String.join()
String result = String.join(", ", arrayList);

// Using StringBuilder
StringBuilder sb = new StringBuilder();
for (String item : arrayList) {
    if (sb.length() > 0) sb.append(", ");
    sb.append(item);
}
String result = sb.toString();
```

Convert String to ArrayList

java

```java
String str = "apple,banana,cherry";
ArrayList<String> list = new ArrayList<>(Arrays.asList(str.split(",")));
```

Filter ArrayList with String methods

java

```java
// Remove empty strings
list.removeIf(String::isEmpty);

// Filter strings starting with "A"
list.removeIf(s -> !s.startsWith("A"));

// Using streams (Java 8+)
List<String> filtered = list.stream()
    .filter(s -> s.length() > 3)
    .collect(Collectors.toList());
```

Performance Tips

1. **Use StringBuilder** for multiple string concatenations instead of + operator
2. **Use ArrayList** over Vector (Vector is synchronized and slower)
3. **Specify initial capacity** for ArrayList if you know approximate size
4. **Use enhanced for loop** for iteration when you don't need index
5. **Use String.equals()** for comparison, not == operator
6. **Consider using StringBuffer** instead of StringBuilder in multi-threaded environments