The String class in Java is a powerful class that provides a variety of methods for string manipulation. Below is a practical example for each method in the String class, along with detailed explanations.

**1. char charAt(int index)**

* Returns the character at the specified index.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = "Hello, World!";  char result = str.charAt(7);  System.out.println("Character at index 7: " + result); // Output: W  }  } |

Explanation: The charAt method retrieves the character at the specified index. Index starts from 0.

**2. boolean equals(Object anObject)**

* Compares the content of the string with another object.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str1 = "Java";  String str2 = "Java";  boolean isEqual = str1.equals(str2);  System.out.println("Are the strings equal? " + isEqual); // Output: true  }  } |

Explanation: This method compares the content of str1 and str2. It is case-sensitive.

**3. boolean equalsIgnoreCase(String anotherString)**

* Compares two strings, ignoring case considerations.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str1 = "Java";  String str2 = "java";  boolean isEqual = str1.equalsIgnoreCase(str2);  System.out.println("Are the strings equal (ignoring case)? " + isEqual); // Output: true  }  } |

Explanation: It ignores case differences when comparing strings.

**4. int compareTo(String anotherString)**

* Compares two strings lexicographically.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str1 = "Apple";  String str2 = "Banana";  int result = str1.compareTo(str2);  System.out.println("Comparison result: " + result); // Output: -1  }  } |

Explanation: Returns a negative, zero, or positive number depending on the comparison.

**5. String concat(String str)**

* Concatenates the specified string to the end of the current string.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str1 = "Hello";  String str2 = " World";  String result = str1.concat(str2);  System.out.println("Concatenated String: " + result); // Output: Hello World  }  } |

Explanation: Combines two strings.

**6. boolean contains(CharSequence s)**

* Checks if the string contains a specific sequence of characters.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = "Hello, World!";  boolean contains = str.contains("World");  System.out.println("Does the string contain 'World'? " + contains); // Output: true  }  } |

Explanation: Checks for the existence of a substring.

**7. boolean startsWith(String prefix)**

* Checks if the string starts with a specific prefix.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = "Java Programming";  boolean starts = str.startsWith("Java");  System.out.println("Does the string start with 'Java'? " + starts); // Output: true  }  } |

**8. boolean endsWith(String suffix)**

* Checks if the string ends with a specific suffix.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = "Hello.java";  boolean ends = str.endsWith(".java");  System.out.println("Does the string end with '.java'? " + ends); // Output: true  }  } |

**9. int indexOf(String str)**

* Returns the index of the first occurrence of a substring.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = "Hello, World!";  int index = str.indexOf("World");  System.out.println("Index of 'World': " + index); // Output: 7  }  } |

**10. int lastIndexOf(String str)**

* Returns the index of the last occurrence of a substring.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = "Hello, Hello, World!";  int lastIndex = str.lastIndexOf("Hello");  System.out.println("Last index of 'Hello': " + lastIndex); // Output: 7  }  } |

**11. int length()**

* Returns the length of the string.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = "Hello, World!";  int length = str.length();  System.out.println("Length of the string: " + length); // Output: 13  }  } |

**12. String replace(CharSequence target, CharSequence replacement)**

* Replaces all occurrences of a target sequence with a replacement sequence.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = "Java is fun!";  String replaced = str.replace("fun", "powerful");  System.out.println("Replaced String: " + replaced); // Output: Java is powerful!  }  } |

**13. String[] split(String regex)**

* Splits the string into an array based on the given regex.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = "Java,Python,C++";  String[] languages = str.split(",");  for (String language : languages)  {  System.out.println(language);  }  }  } |

**14. String substring(int beginIndex, int endIndex)**

* Extracts a substring.

|  |
| --- |
| public class StringExample {  public static void main(String[] args) {  String str = "Hello, World!";  String substr = str.substring(7, 12);  System.out.println("Substring: " + substr); // Output: World  }  } |

**15. String toLowerCase() and String toUpperCase()**

* Converts the string to lower or upper case.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = "Java Programming";  System.out.println("Lowercase: " + str.toLowerCase());  System.out.println("Uppercase: " + str.toUpperCase());  }  } |

**16. String trim()**

* Removes leading and trailing spaces.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = " Hello, World! ";  String trimmed = str.trim();  System.out.println("Trimmed String: '" + trimmed + "'"); // Output: 'Hello, World!'  }  } |

**17. boolean isEmpty()**

* Checks if the string is empty.

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = "";  System.out.println("Is string empty? " + str.isEmpty()); // Output: true  }  } |

**18. boolean isBlank()**

* Checks if the string is empty or contains only white spaces (Java 11+).

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = " ";  System.out.println("Is the string blank? " + str.isBlank()); // Output: true  }  } |

**19. String strip()**

* Removes leading and trailing spaces (Java 11+).

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = " Hello ";  System.out.println("Stripped String: '" + str.strip() + "'"); // Output: 'Hello'  }  } |

Explanation: Similar to trim, but more Unicode-aware.

**20. String repeat(int count)**

* Repeats the string count times (Java 11+).

|  |
| --- |
| public class StringExample  {  public static void main(String[] args)  {  String str = "Hello";  String repeated = str.repeat(3);  System.out.println("Repeated String: " + repeated); // Output: HelloHelloHello  }  } |

**String Practices:**

<https://www.tpointtech.com/java-string-methods>

<https://javaconceptoftheday.com/java-interview-programs-on-strings/>