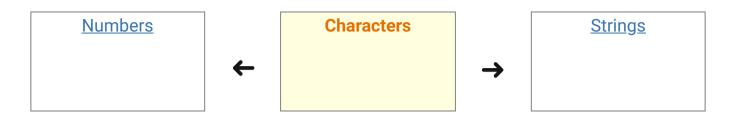
Tutorials Watch & Listen FAQ Oracle University

Home > Tutorials > Numbers and Strings > Characters



Characters

Characters

Most of the time, if you are using a single character value, you will use the primitive char type. For example:

```
char ch = 'a';
// Unicode for uppercase Greek omega character
char uniChar = '\u03A9';
// an array of chars
char[] charArray = { 'a', 'b', 'c', 'd', 'e' };
```

There are times, however, when you need to use a char as an object—for example, as a method argument where an object is expected. The Java programming language provides a wrapper class that "wraps" the char in a Character object for this purpose. An object of type Character contains a single field, whose type is char. This Character class also offers a number of useful class (that is, static) methods for manipulating characters.

You can create a Character object with the Character constructor:

```
1 | Character ch = new Character('a');
```

The Java compiler will also create a <u>Character</u> object for you under some circumstances. For example, if you pass a primitive <u>char</u> into a method that expects an object, the compiler automatically converts the <u>char</u> to a <u>Character</u> for you. This feature is called <u>autoboxing</u>—or <u>unboxing</u>, if the conversion goes the other way. For more information on autoboxing and unboxing, see the section Autoboxing and Unboxing.

Note: The <u>Character</u> class is immutable, so that once it is created, a <u>Character</u> object cannot be changed.

The following table lists some of the most useful methods in the <u>Character</u> class, but is not exhaustive. For a complete listing of all methods in this class (there are more than 50), refer to the <u>Character</u> API specification.

- <u>boolean isLetter(char ch)</u> and <u>boolean isDigit(char ch)</u>:
 Determines whether the specified <u>char</u> value is a letter or a digit, respectively.
- <u>boolean isWhitespace(char ch)</u>: Determines whether the specified <u>char</u> value is white space.
- <u>boolean isUpperCase(char ch)</u> and <u>boolean isLowerCase(char ch)</u>:
 Determines whether the specified char value is uppercase or lowercase, respectively.
- <a href="mailto:char chouperCase(char chouperCase(char
- <u>toString(char ch)</u>: Returns a <u>String</u> object representing the specified character value that is, a one-character string.

Characters and Code Points

The Java platform has supported Unicode Standard starting with JDK 1.0.2. Java SE 15 supports Unicode 13.0. The <a href="https://char.com/

class are based on the original Unicode specification, which defined characters as fixed-width 16-bit entities. The Unicode Standard has since been changed to allow for characters whose representation requires more than 16 bits. The range of legal code points is now U+0000 to U+10FFFF, known as Unicode scalar value.

A char value is encoded with 16 bits. It can thus represent numbers from 0x0000 to 0xFFFF. This set of characters is sometimes referred to as the *Basic Multilingual Plane (BMP)*. Characters whose code points are greater than 0xFFFF (noted U+FFFF) are called *supplementary characters*.

A char value, therefore, represents Basic Multilingual Plane (BMP) code points. An int value represents all Unicode code points, including supplementary code points. Unless otherwise specified, the behavior with respect to supplementary characters and surrogate char values is as follows:

- The methods that only accept a char value cannot support supplementary characters. They treat char values from the surrogate ranges as undefined characters.
- The methods that accept an int value support all Unicode characters, including supplementary characters.

You can refer to the documentation of the <u>Character</u> class for more information.

Escape Sequence

A character preceded by a backslash (\) is an escape sequence and has special meaning to the compiler. The following table shows the Java escape sequences:

Escape Sequence	Description
\t	Insert a tab in the text at this point.
\b	Insert a backspace in the text at this point.
\n	Insert a newline in the text at this point.
\r	Insert a carriage return in the text at this point.
\f	Insert a form feed in the text at this point.
\ 1	Insert a single quote character in the text at this point.
\"	Insert a double quote character in the text at this point.
\\	Insert a backslash character in the text at this point.

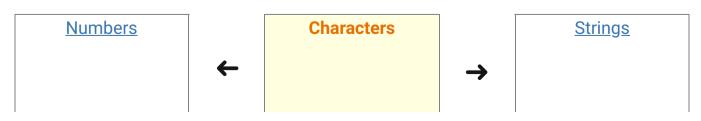
When an escape sequence is encountered in a print statement, the compiler interprets it accordingly. For example, if you want to put quotes within quotes you must use the escape sequence, ", on the interior quotes. To print the sentence

```
_{1}\mid She said "Hello!" to me.
```

you would write

```
1 | System.out.println("She said \"Hello!\" to me.");
```

Last update: September 14, 2021



<u>Home</u> > <u>Tutorials</u> > <u>Numbers and Strings</u> > Characters