Scala data types

A data type indicates to the compiler the type of value that will be stored at a certain location. In contrast to Java, Scala has no primitive data types because everything is an object.
Scala treats the following data types as objects and allows them to invoke methods on int, String, and long.

- **Byte:** Byte is a signed 8-bit value. The value is between -128 and 127.
- **Short:** is a 16-bit signed value in the range -32768 to 32767.
- **Int:** is a 32-bit signed value between -2147483648 and 2147483647.
- **Long:** is a signed 64-bit value between -9223372036854775808 and 9223372036854775807
- **Float:** is a 32-bit single-precision floating-point number.
- **Double:** is a 64-bit double precision floating-point number.
- **Char:** is a 16-bit unsigned unicode character that spans the range U+0000 to U+FFFF.
- **String:** A string is a collection of characters encapsulated in double quotations.
- **Boolean:** a literal that contains both true and false values.
- **AnyRef:** Supertype of any reference
- **Nothing:** Subtype of every other type includes no values
- **Null:** Empty reference
- **Unit:** Corresponds to no value
- **Any:** Super type of any type or object of any type

Scala Basic Literals

The fundamental literals are as follows:

- **Integer literals:** They are of type int,long, and are prefixed with the L or l suffix. Several examples of integer literals include 42,42FFF, and so on.

- **Floating point Literals:** It includes float followed by the suffix F or f. 3.12, 4.67, and so on are examples.

- **Strings Literals:** A string is a collection of characters enclosed in double quotes. "Hello World," "Set of Strings," and other such phrases are examples of strings.

- **A character literals:** It is a single character wrapped in quotation marks. Examples include ‘\n’, ‘\t’ etc.
- **Multiline Strings:** A multiline string is a collection of characters surrounded in triple quotes """"... "". The sequence may contain newlines and other controlled characters. A multiline string is an illustration of this.

  “””String is a
  sequence of
  characters.”””

- **NULL value:** A null value is a reference to a null object.
- **Escape Sequence:** The following escape sequences are supported in Scala
  - ➔ \b: backspace
  - ➔ \t: Horizontal tab
  - ➔ \f: form feed
  - ➔ \n: newline character
  - ➔ \r: carriage return
  - ➔ \”: double quote
  - ➔ \’: single quote
  - ➔ \\: backslash

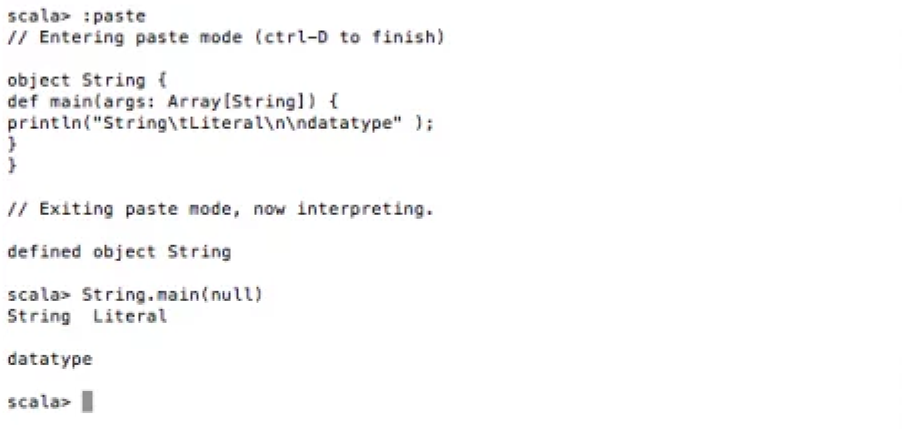
Consider the following example of an escape sequence;

```
object String {
def main(args: Array[String]) {
println("String\tLiteral\n\ndatatype" );
}
}
```

Execute the preceding code in the shell as demonstrated below.

```
scala>String.main(null)
String   Literal
datatype
scala>
```

This output indicates that a tab space has been added between String and Literal, as well as the addition of a new line stating the data type. The following graphic illustrates its execution in the Scala shell.



```
scala> :paste
// Entering paste mode (ctrl-D to finish)

object String {
def main(args: Array[String]) {
println("String\tLiteral\n\ndatatype" );
}
}

// Exiting paste mode, now interpreting.

defined object String

scala> String.main(null)
String  Literal

datatype

scala> 
```