How do Scala classes extend AnyRef?

What does this provide?

With `ScalaObject`.

This only provides `$tag`, for pattern matching speed.

What does `AnyRef` add to `java.lang.Object`?

`final eq

final ne

final ==

final !=`

What does `Any` provide?

`final ==` (forwards to equals)
`final !=` (opposite of ==, guaranteed)

`equals(Any)
hashCode()
toString()
final isInstanceOf[T]
final asInstanceOf[t] : T` (cast)

What does `AnyVal` add?

nothing

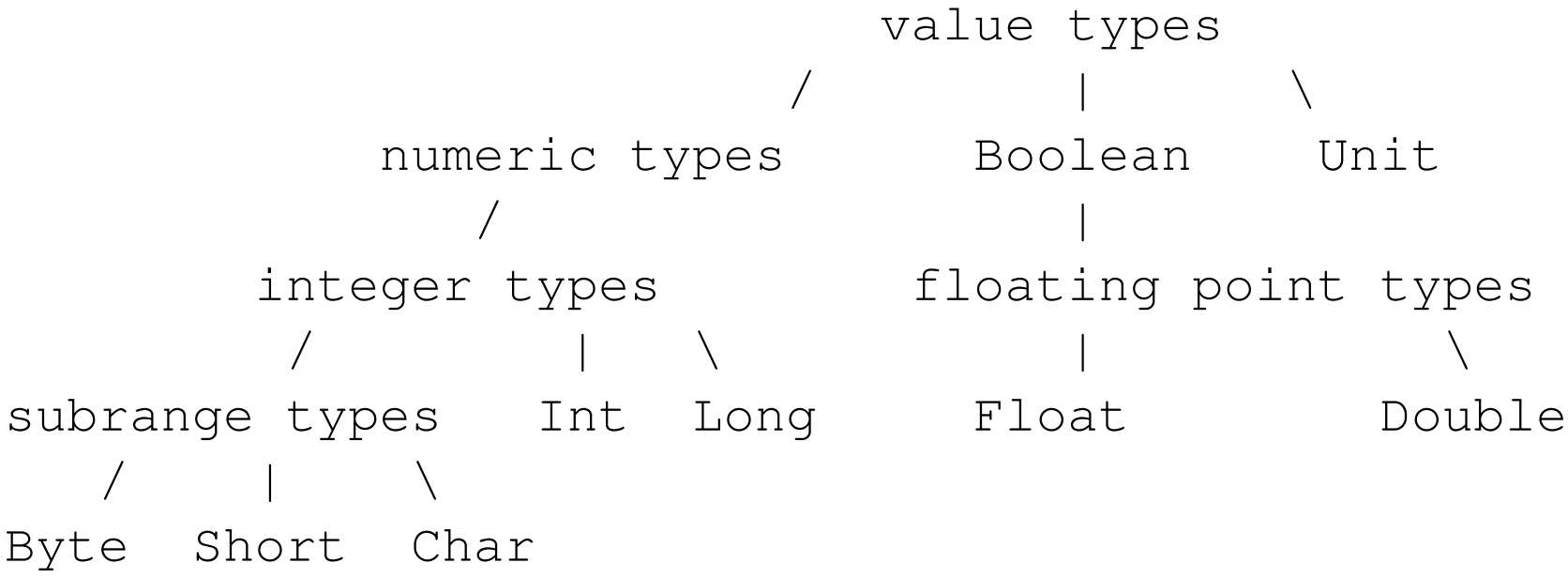
How are value types special?

They are declared abstract and final to prevent instantiation with the new keyword.

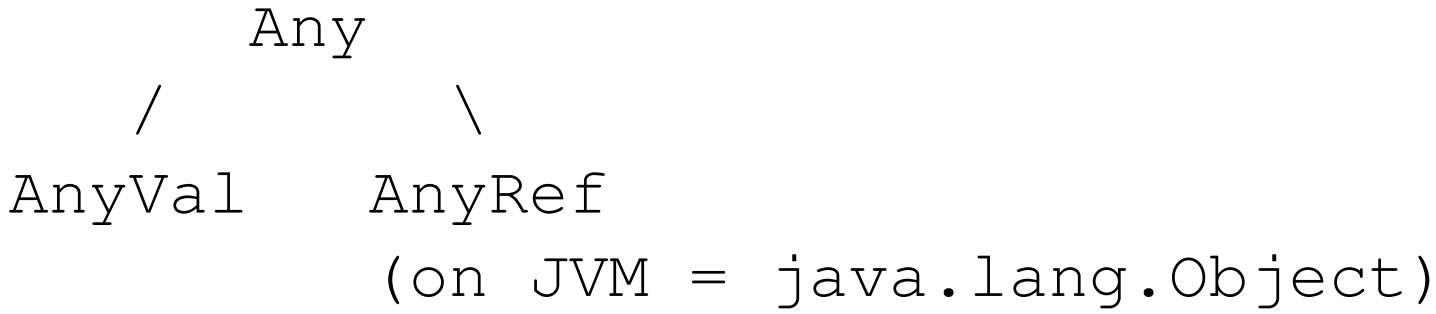
To what does every value type have implicit conversion?

It has conversion to a rich type in `scala.runtime` that provides more methods.

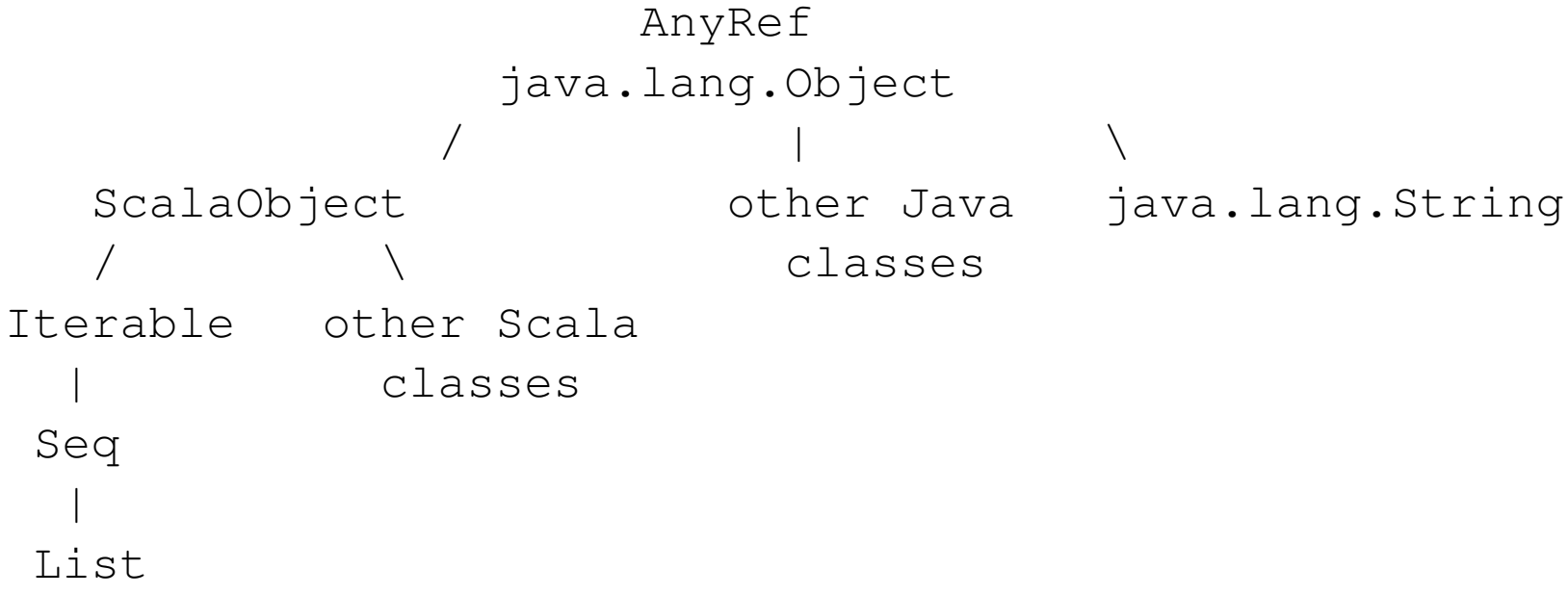
What is the terminology of the value types hierarchy?


value types
numeric types
Boolean
Unit
integer types
floating point types
subrange types
Int
Long
Float
Double
Byte
Short
Char

What is the top of the Scala type hierarchy?


Any
AnyVal
AnyRef
(on JVM = java.lang.Object)

What is the `AnyRef` hierarchy?


AnyRef
java.lang.Object
ScalaObject
other Java classes
java.lang.String
Iterable
other Scala classes
Seq
List

What is a literal identifier?

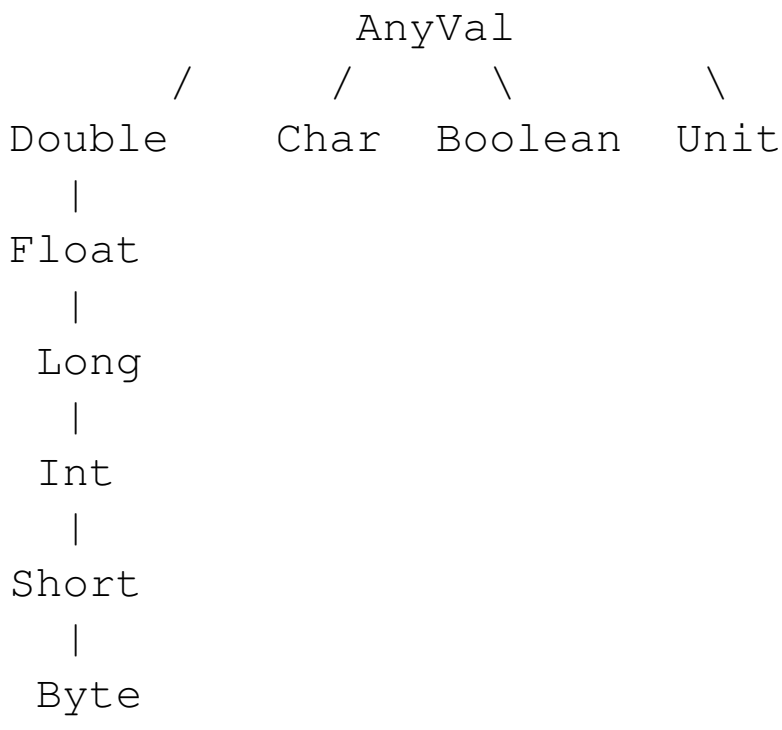
What does it do?

Any sequence of acceptable characters enclosed by back ticks.

It forces identifier interpretation over keywords or other restrictions.

E.g., `Thread.\yield\()`

What is the `AnyVal` hierarchy?



Implicit conversions:

```
Char -> Int
Byte -> Short -> Int -> Long -> Float -> Double
```