TOUR OF SCALA

IMPLICIT CONVERSIONS

An implicit conversion from type `S` to type `T` is defined by an implicit value which has function type `S => T`, or by an implicit method convertible to a value of that type.

Implicit conversions are applied in two situations:

- If an expression `e` is of type `S`, and `S` does not conform to the expression's expected type `T`.
- In a selection `e.m` with `e` of type `S`, if the selector `m` does not denote a member of `S`.

In the first case, a conversion `c` is searched for which is applicable to `e` and whose result type conforms to `T`. In the second case, a conversion `c` is searched for which is applicable to `e` and whose result contains a member named `m`.

If an implicit method `List[A] => Ordered[List[A]]` is in scope, as well as an implicit method `Int => Ordered[Int]`, the following operation on the two lists of type `List[Int]` is legal:

```
List(1, 2, 3) <= List(4, 5)
```

An implicit method `Int => Ordered[Int]` is provided automatically through `scala.Predef.intWrapper`. An example of an implicit method `List[A] => Ordered[List[A]]` is provided below.

```
import scala.language.implicitConversions

implicit def list2ordered[A](x: List[A])
    (implicit elem2ordered: A => Ordered[A]): Ordered[List[A]] =
  new Ordered[List[A]] {
    //replace with a more useful implementation
    def compare(that: List[A]): Int = 1
  }
```

The implicitly imported object `scala.Predef` declares several aliases to frequently used types (e.g. `scala.collection.immutable.Map` is aliased to `Map`) and methods (e.g. `assert`) but also several implicit conversions.

For example, when calling a Java method that expects a `java.lang.Integer`, you are free to pass it a `scala.Int` instead. That's because Predef includes the following implicit conversions:

```
import scala.language.implicitConversions

implicit def int2Integer(x: Int) =
  java.lang.Integer.valueOf(x)
```

Because implicit conversions can have pitfalls if used indiscriminately the compiler warns when compiling the implicit conversion definition.

To turn off the warnings take either of these actions:

- Import `scala.language.implicitConversions` into the scope of the implicit conversion definition
- Invoke the compiler with `-language:implicitConversions`

No warning is emitted when the conversion is applied by the compiler

No warning is emitted when the conversion is applied by the compiler.

← previous next →

Contributors to this page:

 ckipp01  mlachkar  ashawley  Duhemm  martijnhoekstra  heathermiller