Why the package mixed

The problem

The R ecosystem already has some very good packages that deal with labelled objects. In particular, the inter-connected packages **haven** and **labelled** already provide the functionality most users would ever need.

As nice and useful these packages are, it has become apparent they have some fundamental design features that run, in some situations, against users' expectations. This has a lot to do with the treatment of declared missing values, that are fundamental for the social sciences.

The following minimal example (adapted from the vignette in package **haven**) illustrates the situation:

```
library(haven)

x1 <- labelled_spss(c(1:5, 99), c(Missing = 99), na_value = 99)
```

The printed objects from this package nicely display some properties:

```
> x1
<labelled_spss<double>[6]>
[1]  1  2  3  4  5 99
Missing values: 99

Labels:
 value   label
    99 Missing
```

There are 5 normal values (suppose they represent number of children), and one declared missing value coded 99. This value *acts* as a missing value, but it is different from a regular missing value in R, coded NA. The later stands for any missing information (something like an empty cell) regardless of the reason.

Here, on the other hand, the cell is *not* empty, but the value 99 is not a valid value either. It cannot possibly represent 99 children in the household, but for instance it could mean the respondent did not want to respond. It is properly identified as missing, with:

```
is.na(x1)
[1] FALSE FALSE FALSE FALSE FALSE  TRUE
```

But if performing a mean, for instance, the normal expectation is that value 99 would not play any role in the calculations (since it is *missing*). However:

```
mean(x1)
[1] 19
```

This means the value 99 did play an active role despite being identified as "missing". In an ideal world, the expected mean would be 3, or at best employ the argument `na.rm = TRUE` if the result is NA because of the declared missing value.

A solution to this problem is offered by package **labelled**, which has a function called `user_na_to_na()`:

```
mean(user_na_to_na(x1), na.rm = TRUE)
[1] 3
```

The mixed solution

While solving the problem, this above solution forces two additional operations:

- converting the declared missing values, and
- employing the `na.rm` argument.

This should not be necessary, especially if (and it is extremely likely that) users may forget the declared missing values are not actually missing values. To completely solve this situation, package **mixed** creates a very similar object, where declared missing values are actually stored (hence interpreted as) regular NA missing values in R.

```
library(mixed)

x2 <- mixed_labelled(c(1:5, 99), c(Missing = 99), na_value = 99)

x2
<mixed_labelled<integer>[6]>
[1]      1      2      3      4      5 NA(99)
Missing values: 99

Labels:
 value   label
    99 Missing
```

It is now obvious that value 99 is not anymore a regular number, but an actual missing value. More importantly, it circumvents the need to convert declared missing values to regular NAs, since they are already stored as NA values. The average value is calculated simply as:

```
mean(x2)
[1] 3
```

Notice that neither `user_na_to_na()`, nor employing `na.rm = TRUE` are necessary. Despite being stored as an NA value, the value 99 is not equivalent to an *empty cell*. The information still exists, and it is simply ignored in the calculations.

The `na.rm = TRUE` is only necessary if there are other unexplained missing values in the data:

```
mean(c(x2, NA))
[1] NA

mean(c(x2, NA), na.rm = TRUE)
[1] 3
```

As it can be seen, concatenating on this vector creates a similar one of the same class:

```
x2 <- c(x2, -1, 99)

x2
<mixed_labelled<integer>[8]>
[1]      1      2      3      4      5 NA(99)     -1 NA(99)
Missing values: 99

Labels:
 value   label
    99 Missing
```

It should be made obvious that packages **haven** and **labelled** are excellent packages which are not inherently

doing a bad thing: the very same result is obtained, just via a different route. Package **mixed** should not even be necessary, if the design philosophy of these packages would be different.

The functions in this package offer an alternative to packages **haven** and **labelled**, with only one but fundamental difference: instead of treating existing values as missing, package **mixed** interprets missing values as existing.

The declared missing values are (just like in package **haven**) identified as NAs, but they can also be compared against the original values:

```
is.na(x2)
[1] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE

x2 == 99
[1] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE
```

Most functions are designed to be as similar as possible, for instance `value_labels()` to add / change value labels:

```
value_labels(x2) <- c(DK = -1, NR = 99)

x2
<mixed_labelled<integer>[8]>
[1]      1      2      3      4      5 NA(99)     -1 NA(99)
Missing values: 99

Labels:
 value label
    -1    DK
    99    NR
```

The value -1 is now properly labelled, and it can further be declared as missing. Such declarations do not necessarily have to use the main function `mixed_labelled()`, because there is the separate function `missing_values()`:

```
missing_values(x2) <- c(-1, 99)

x2
<mixed_labelled<integer>[8]>
[1]      1      2      3      4      5 NA(99) NA(-1) NA(99)
Missing values: -1, 99

Labels:
 value label
    -1    DK
    99    NR
```