Title

stata.com

label language — Labels for variables and values in multiple languages

Syntax

List defined languages

```
label language
```

Change labels to specified language name

```
label language languagename
```

Create new set of labels with specified language name

```
label language languagename, new [copy]
```

Rename current label set

```
label language languagename, rename
```

Delete specified label set

```
label language languagename, delete
```

Menu

Data > Data utilities > Label utilities > Set label language

Description

`label language` lets you create and use datasets that contain different sets of data, variable, and value labels. A dataset might contain one set in English, another in German, and a third in Spanish. A dataset may contain up to 100 sets of labels.

We will write about the different sets as if they reflect different spoken languages, but you need not use the multiple sets in this way. You could create a dataset with one set of long labels and another set of shorter ones.

One set of labels is in use at any instant, but a dataset may contain multiple sets. You can choose among the sets by typing

```
. label language languagename
```

When other Stata commands produce output (such as `describe` and `tabulate`), they use the currently set language. When you define or modify the labels by using the other `label` commands (see [D] **label**), you modify the current set.

`label language` (without arguments)
lists the available languages and the name of the current one. The current language refers to the labels you will see if you used, say, `describe` or `tabulate`. The available languages refer to the names of the other sets of previously created labels. For instance, you might currently be using the labels in `en` (English), but labels in `de` (German) and `es` (Spanish) may also be available.

`label language` *languagename*
changes the labels to those of the specified language. For instance, if `label language` revealed that `en`, `de`, and `es` were available, typing `label language de` would change the current language to German.

`label language` *languagename*`, new`
allows you to create a new set of labels and collectively name them *languagename*. You may name the set as you please, as long as the name does not exceed 24 characters. If the labels correspond to spoken languages, we recommend that you use the language's ISO 639-1 two-letter code, such as `en` for English, `de` for German, and `es` for Spanish. A list of codes for popular languages is listed in the appendix below. For a complete list, see http://lcweb.loc.gov/standards/iso639-2/iso639jac.html.

`label language` *languagename*`, rename`
changes the name of the label set currently in use. If the label set in use were named `default` and you now wanted to change that to `en`, you could type `label language en, rename`.

Our choice of the name `default` in the example was not accidental. If you have not yet used `label language` to create a new language, the dataset will have one language, named `default`.

`label language` *languagename*`, delete`
deletes the specified label set. If *languagename* is also the current language, one of the other available languages becomes the current language.

Option

`copy` is used with `label language, new` and copies the labels from the current language to the new language.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Creating labels in the first language

You can begin by ignoring the `label language` command. You create the data, variable, and value labels just as you would ordinarily; see [D] **label**.

```
. label data "1978 Automobile Data"
. label variable foreign "Car type"
. label values foreign origin
. label define origin  0 "Domestic"  1 "Foreign"
```

At some point—at the beginning, the middle, or the end—rename the language appropriately. For instance, if the labels you defined were in English, type

```
. label language en, rename
```

`label language, rename` simply changes the name of the currently set language. You may change the name as often as you wish.

Creating labels in the second and subsequent languages

After creating the first language, you can create a new language by typing

```
. label language newlanguagename, new
```

or by typing the two commands

```
. label language existinglanguagename
. label language newlanguagename, new copy
```

In the first case, you start with a clean slate: no data, variable, or value labels are defined. In the second case, you start with the labels from *existinglanguagename*, and you can make the changes from there.

Creating labels from a clean slate

To create new labels in the language named `de`, type

```
. label language de, new
```

If you were now to type `describe`, you would find that there are no data, variable, or value labels. You can define new labels in the usual way:

```
. label data "1978 Automobil Daten"
. label variable foreign "Art Auto"
. label values foreign origin_de
. label define origin_de  0 "Innen"  1 "Ausländisch"
```

Creating labels from a previously existing language

It is sometimes easier to start with the labels from a previously existing language, which you can then translate:

```
. label language en
. label language de, new copy
```

If you were now to type `describe`, you would see the English-language labels, even though the new language is named `de`. You can then work to translate the labels:

```
. label data "1978 Automobil Daten"
. label variable foreign "Art Auto"
```

Typing `describe`, you might also discover that the variable `foreign` has the value label `origin`. Do not change the contents of the value label. Instead, create a new value label:

```
. label define origin_de  0 "Innen"  1 "Ausländisch"
. label values foreign origin_de
```

Creating value labels with the `copy` option is no different from creating them from a clean slate, except that you start with an existing set of labels from another language. Using `describe` can make it easier to translate them.

Switching languages

You can discover the names of the previously defined languages by typing

```
. label language
```

You can switch to a previously defined language—say, to `en`—by typing

```
. label language en
```

Changing the name of a language

To change the name of a previously defined language make it the current language and then specify the `rename` option:

```
. label language de
. label language German, rename
```

You may rename a language as often as you wish:

```
. label language de, rename
```

Deleting a language

To delete a previously defined language, such as `de`, type

```
. label language de, delete
```

The `delete` option deletes the specified language and, if the language was also the currently set language, resets the current language to one of the other languages or to `default` if there are none.

Appendix: Selected ISO 639-1 two-letter codes

You may name languages as you please. You may name German labels `Deutsch`, `German`, `Aleman`, or whatever else appeals to you. For consistency across datasets, if the language you are creating is a spoken language, we suggest that you use the ISO 639-1 two-letter codes. Some of them are listed below, and the full list can be found at http://lcweb.loc.gov/standards/iso639-2/iso639jac.html.

Two-letter code	English name of language
ar	Arabic
cs	Czech
cy	Welsh
de	German
el	Greek
en	English
es	Spanish; Castillian
fa	Persian
fi	Finnish
fr	French
ga	Irish
he	Hebrew
hi	Hindi
is	Icelandic
it	Italian
ja	Japanese
kl	Kalaallisut; Greenlandic
lt	Lithuanian
lv	Latvian
nl	Dutch; Flemish
no	Norwegian
pl	Polish
pt	Portuguese
ro	Romanian; Moldavian
ru	Russian
sk	Slovak
sr	Serbian
sv	Swedish
tr	Turkish
uk	Ukrainian
uz	Uzbek
zh	Chinese

Stored results

`label language` without arguments stores the following in `r()`:

Scalars
- `r(k)` number of languages defined

Macros
- `r(languages)` list of languages, listed one after the other
- `r(language)` name of current language

Methods and formulas

This section is included for programmers who wish to access or extend the services `label language` provides.

Language sets are implemented using [P] **char**. The names of the languages and the name of the current language are stored in

`_dta[_lang_list]`	list of defined languages
`_dta[_lang_c]`	currently set language

If these characteristics are undefined, results are as if each contained the word "`default`". Do not change the contents of the above two macros except by using `label language`.

For each language *languagename* except the current language, data, variable, and value labels are stored in

`_dta[_lang_v_`*languagename*`]`	data label
varname`[_lang_v_`*languagename*`]`	variable label
varname`[_lang_l_`*languagename*`]`	value-label name

References

Golbe, D. L. 2010. Stata tip 83: Merging multilingual datasets. *Stata Journal* 10: 152–156.

Weesie, J. 2005. Multilingual datasets. *Stata Journal* 5: 162–187.

Also see

[D] **label** — Manipulate labels

[D] **labelbook** — Label utilities

[D] **codebook** — Describe data contents