**fvrevar — Factor-variables operator programming command**

**Description**

`fvrevar` creates a variable list that includes equivalent, temporary variables in place of the factor variables, interactions, or time-series–operated variables in `varlist`. The resulting variable list can be used by commands that do not otherwise support factor variables or time-series–operated variables. The resulting list also could be used in a program to speed execution at the cost of using more memory.

**Quick start**

Create temporary indicator variables for the levels of categorical variable `a` and store names in `r(varlist)`

```
fvrevar i.a
```

Create temporary variables corresponding to the levels of `a`, `b`, and their interaction

```
fvrevar i.a##i.b
```

As above, and create a temporary variable for the lag of `x` using `tsset` data

```
fvrevar i.a##i.b L.x
```

Return the list of unoperated variables (`a`, `b`, and `x`) in `r(varlist)`

```
fvrevar i.a##i.b L.x, list
```

Create new variables `a_1`, `a_2`, …, corresponding to the levels of `a`

```
fvrevar i.a, stub(a_)
```

Create new variables `ab_1`, `ab_2`, …, corresponding to the levels of the interaction between `a` and `b`

```
fvrevar i.a##i.b, stub(ab_)
```
Syntax

```plaintext
fvrevar [varlist] [if] [in], substitute tsonly list stub(stub)
```

You must `tsset` your data before using `fvrevar` if `varlist` contains time-series operators; see `[TS] tsset`. `collect` is allowed; see [U] 11.1.10 Prefix commands.

Options

- `substitute` specifies that equivalent, temporary variables be substituted for any factor variables, interactions, or time-series–operated variables in `varlist`. `substitute` is the default action taken by `fvrevar`; you do not need to specify the option.
- `tsonly` specifies that equivalent, temporary variables be substituted for only the time-series–operated variables in `varlist`.
- `list` specifies that all factor-variable operators and time-series operators be removed from `varlist` and the resulting list of base variables be returned in `r(varlist)`. No new variables are created with this option.
- `stub(stub)` specifies that `fvrevar` generate named variables instead of temporary variables. The new variables will be named `stub#`.

Remarks and examples

`fvrevar` might create no new variables, one new variable, or many new variables, depending on the number of factor variables, interactions, and time-series operators appearing in `varlist`. Any new variables created are temporary. The new, equivalent `varlist` is returned in `r(varlist)`. The new `varlist` corresponds one to one with the original `varlist`.

Example 1

Typing

```plaintext
use https://www.stata-press.com/data/r17/auto2
fvrevar i.rep78 mpg turn
```

creates five temporary variables corresponding to the levels of `rep78`. No new variables are created for variables `mpg` and `turn` because they do not contain factor-variable or time-series operators.

The resulting variable list is

```plaintext
display "r(varlist)"
"__000000 __000001 __000002 __000003 __000004" mpg turn
```

(Your temporary variable names may be different, but that is of no consequence.) Temporary variables automatically vanish when the program concludes.
Example 2

Suppose we want to create temporary variables for specific levels of a factor variable. To do this, we can use the parenthesis notation of factor-variable syntax.

```
. fvrevar i(2,3)bn.rep78 mpg
```

creates two temporary variables corresponding to levels 2 and 3 of rep78. Notice that we specified that neither level 2 nor 3 be set as the base level by using the bn notation. If we did not specify bn, level 2 would have been treated as the base level.

The resulting variable list is

```
. display "'r(varlist)'"
  __000005 __000002 mpg
```

We can see the results by listing the new variables alongside the original value of rep78.

```
. list rep78 'r(varlist)' in 1/5
```

<table>
<thead>
<tr>
<th>rep78</th>
<th>__000005</th>
<th>__000002</th>
<th>mpg</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Average</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2.</td>
<td>Average</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>4.</td>
<td>Average</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5.</td>
<td>Good</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

If we had needed only the base-variable names, we could have specified

```
. fvrevar i(2,3)bn.rep78 mpg, list
. display "'r(varlist)'"
  mpg rep78
```

The order of the list will probably differ from that of the original list; base variables are listed only once.

Example 3

Now let’s assume we have a varlist containing both an interaction and time-series–operated variables. If we want to create temporary variables for the entire equivalent varlist, we can specify fvrevar with no options.

```
. generate t = _n
. tsset t
time variable: t, 1 to 74
delta: 1 unit
. fvrevar c.turn#i(2,3).rep78 L.mpg
```

The resulting variable list is

```
. display "'r(varlist)'"
  __000005 __000007 __000008
```

If we want to create temporary variables only for the time-series–operated variables, we can specify the tsonly option.

```
. fvrevar c.turn#i(2,3).rep78 L.mpg, tsonly
```
The resulting variable list is

```
   . display "'r(varlist)'"
   2.rep78#c.turn 3.rep78#c.turn __000008
```

Notice that `fvrevar` returned the expanded factor-variable list with the `tsonly` option.

### Technical note

`fvrevar`, `substitute` avoids creating duplicate variables. Consider

```
   . fvrevar i.rep78 turn mpg i.rep78
```

`i.rep78` appears twice in the `varlist`. `fvrevar` will create only one set of new variables for the five levels of `rep78` and will use these new variables once in the resulting `r(varlist)`. Moreover, `fvrevar` will do this even across multiple calls:

```
   . fvrevar i.rep78 turn mpg
   . fvrevar i.rep78
```

`i.rep78` appears in two separate calls. At the first call, `fvrevar` creates five temporary variables corresponding to the five levels of `rep78`. At the second call, `fvrevar` remembers what it has done and uses the same temporary variables for `i.rep78`.

### Stored results

`fvrevar` stores the following in `r()`:

- **Macros**
  - `r(varlist)` the modified variable list or list of base-variable names

### Also see

- [TS] `tsrevar` — Time-series operator programming command
- [P] `fvexpand` — Expand factor varlists
- [P] `syntax` — Parse Stata syntax
- [P] `unab` — Unabbreviate variable list
- [U] 11 Language syntax
- [U] 11.4.4 Time-series varlists
- [U] 18 Programming Stata