The Grammar Matrix Customization System was unable to create a grammar with the provided choices file. You may go back in your browser to try and fix the problem, or if you think there is a bug in the system you may email the choices file to the developers

View Choices File (right-click to download)

In most cases, you can go back in your browser and fix the problems, but if not you may reload an empty questionnaire (this will erase your changes, so be sure to save your choices (above) first).

You may also wish to see the Python error (note: it is very technical, and possibly not useful).

IndexError

Python 3.6.8: /usr/local/python-virt/matrix/bin/python3
Thu Jan 20 22:43:03 2022

A problem occurred in a Python script. Here is the sequence of function calls leading up to the error, in the order they occurred.

/home2/www-matrix/html/customize/matrix.cgi in **<module>**()

```
    243         # Create the customized grammar
    244         try:
=>  245             grammar_dir = customize_matrix(session_path, arch_type)
    246         except:
    247             exc = sys.exc_info()
```

grammar_dir *undefined*, **customize_matrix** = <function customize_matrix>, **session_path** = 'sessions/9695', **arch_type** = 'tgz'

/home2/www-matrix/html/customize/gmcs/customize.py in **customize_matrix**(path='sessions/9695/choices', arch_type='tgz', destination='sessions/9695', force_dest=False)

```
    508     argument_optionality.customize_arg_op(mylang, ch, rules, hierarchies)
    509     direct_inverse.customize_direct_inverse(ch, mylang, hierarchies)
=>  510     case.customize_case(mylang, ch, hierarchies)
    511 
    512     # after all structures have been customized, customize inflection,
```

global **case** = <module 'gmcs.linglib.case' from '/home2/www-matrix/html/customize/gmcs/linglib/case.py'>, case.**customize_case** = <function customize_case>, *global* **mylang** = <gmcs.tdl.TDLfile object>, *global* **ch** = <gmcs.choices.ChoicesFile object>, *global* **hierarchies** = {'aspect': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'case': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'form': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'gender': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'mood': <gmcs.lib.tdlhierarchy.TDLHierarchy

object>, 'person': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'tense': <gmcs.lib.tdlhierarchy.TDLHierarchy object>}

/home2/www-matrix/html/customize/gmcs/linglib/case.py in **customize_case**(mylang=<gmcs.tdl.TDLfile object>, ch=<gmcs.choices.ChoicesFile object>, hierarchies={'aspect': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'case': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'form': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'gender': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'mood': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'person': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'tense': <gmcs.lib.tdlhierarchy.TDLHierarchy object>})

```

    # now output the case hierarchies
=>  customize_case_type(mylang, hierarchies)


```

global **customize_case_type** = <function customize_case_type>, **mylang** = <gmcs.tdl.TDLfile object>, **hierarchies** = {'aspect': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'case': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'form': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'gender': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'mood': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'person': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'tense': <gmcs.lib.tdlhierarchy.TDLHierarchy object>}

/home2/www-matrix/html/customize/gmcs/linglib/case.py in **customize_case_type**(mylang=<gmcs.tdl.TDLfile object>, hierarchies={'aspect': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'case': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'form': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'gender': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'mood': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'person': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'tense': <gmcs.lib.tdlhierarchy.TDLHierarchy object>})

```
def customize_case_type(mylang, hierarchies):
    if 'case' in hierarchies:
=>      hierarchies['case'].save(mylang)


```

hierarchies = {'aspect': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'case': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'form': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'gender': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'mood': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'person': <gmcs.lib.tdlhierarchy.TDLHierarchy object>, 'tense': <gmcs.lib.tdlhierarchy.TDLHierarchy object>},].save *undefined*, **mylang** = <gmcs.tdl.TDLfile object>

/home2/www-matrix/html/customize/gmcs/lib/tdlhierarchy.py in **save**(self=<gmcs.lib.tdlhierarchy.TDLHierarchy object>, tdl_file=<gmcs.tdl.TDLfile object>, define=True)

```

        for h in self.hierarchy:
```

```
=>   45             tdl_file.add(h[0] + ' := ' + h[1] + '.', h[2], True)
     46 
     47     # For each type in the hierarchy, calculate which types it is
```

tdl_file = <gmcs.tdl.TDLfile object>, tdl_file.**add** = <bound method TDLfile.add of <gmcs.tdl.TDLfile object>>, **h** = ['', 'loc', '']

/home2/www-matrix/html/customize/gmcs/tdl.py in **add**(self=<gmcs.tdl.TDLfile object>, tdl_type=' := loc.', comment='', one_line=True, merge=False, section='')

```
    926         definition if possible.
    927         """
=>  928         typedef = TDLparse(tdl_type)
    929         typedef.set_comment(comment)
    930         typedef.set_one_line(one_line)
```

typedef *undefined*, *global* **TDLparse** = <function TDLparse>, **tdl_type** = ' := loc.'

/home2/www-matrix/html/customize/gmcs/tdl.py in **TDLparse**(s=' := loc.')

```
    677     global tok
    678     tok = TDLtokenize(s)
=>  679     return TDLparse_typedef()
    680 
    681 
```

global **TDLparse_typedef** = <function TDLparse_typedef>

/home2/www-matrix/html/customize/gmcs/tdl.py in **TDLparse_typedef**()

```
    667     except IndexError:
    668         raise IndexError("Pop from empty list: the TDL defined probably " +
=>  669                          "is missing the operator (:= or :+)")
    670     elem = TDLelem_typedef(type, op)
    671     elem.add(TDLparse_conj())
```

IndexError: Pop from empty list: the TDL defined probably is missing the operator (:= or :+)

args = ('Pop from empty list: the TDL defined probably is missing the operator (:= or :+)',)

with_traceback = <built-in method with_traceback of IndexError object>