Meta-programming in Nim

Peter Munch-Ellingsen

@PMunch – peterme.net
What is Nim?

» Compiled (C/C++/JS)
» Statically typed
» Speed of C, ease of Python, flexibility of Perl

```nim
# Compute average line length
var
  sum = 0
  count = 0

for line in stdin.lines:
  sum += line.len
  count += 1

echo("Average line length: ",
     if count > 0: sum / count else: 0)
```
What is meta-programming?

```c
#if VERBOSE >= 2
    printf("trace message");
#endif

#ifndef __unix__
    # include <unistd.h>
    #elif defined _WIN32
    # include <windows.h>
#endif

when LogLevel >= 2:
    echo "trace message"

when defined(unix):
    import unixlib
else if defined(windows):
    import winlib
```
Why meta-programming?

» Can optimise code – by compile-time rewrites
» Can enforce better coding patterns
» Can increase code read-, and maintainability
Meta-programming in Nim

» Works on the Abstract Syntax Tree
» Respects the type system
» Levels of complexity:
  • Normal procs and inline iterators
  • Generic procs and closure iterators
  • Templates
  • Macros
Templates – AST substitution

template withLock(lock: Lock, body: untyped) =
  acquire lock
  try:
    body
  finally:
    release lock

var ourLock: Lock
initLock ourLock

withLock ourLock:
    echo "Do something that requires locking"
    echo "This might throw an exception"
var logLevel* = Level.debug

template debug*(args: varargs[string, `\$\`]) =
    if logLevel <= Level.debug:
        echo "[\$# $#][\$#]: $#" % [getDateStr(), getClockStr(),
            join args]

proc expensiveDebuggingInfo*: string =
    sleep(milsecs = 1000)
    result = "Everything looking good!"

debuge expensiveDebuggingInfo()
Macros – AST building

» Takes an abstract syntax tree
» Returns an abstract syntax tree
» Input must be syntactically correct
» Can create Domain Specific Languages

```python
import json
var
johnAge = 30
x = %*
    "name": "John",
    "age": johnAge
}
Macros – AST building

```
proc buttonCallback(e: var WxCommandEvent) {.cdecl.} =
gauge.setValue(50)

genui:
mainFrame % Frame(title = "Hello World"):
  Panel | Boxsizer(orient = wxHorizontal):
    StaticBox(label = "Basic controls")[proportion = 1] |
    StaticBoxSizer(orient = wxVertical):
      Button(label = "Button") -> (wxEVT_BUTTON, buttonCallback)
      CheckBox: "Checkbox"
    StaticBox(label = "More controls")[proportion = 1] |
    StaticBoxSizer(orient = wxVertical):
      TextCtrl(value = "Entry")
    gauge % Gauge(range = 100)

mainFrame.show()
runMainLoop()
```
mainFrame = cnew constructWxFrame(title = "Hello World", parent = nil, id = wxID_ANY)
tmp274050 = cnew constructWxPanel(parent = mainFrame, id = wxID_ANY)
tmp274051 = cnew constructWxBoxsizer(orient = wxHorizontal)
tmp274052 = cnew constructWxStaticBox(label = "Basic controls", parent = tmp274050, id = wxID_ANY)
tmp274053 = cnew constructWxStaticBoxSizer(orient = wxVertical, box = tmp274052)
tmp274054 = cnew constructWxButton(label = "Button", parent = tmp274052, id = wxID_ANY)
tmp274055 = cnew constructWxCheckBox(parent = tmp274052, id = wxID_ANY, label = "Checkbox")
tmp274056 = cnew constructWxStaticBox(label = "More controls", parent = tmp274050, id = wxID_ANY)
tmp274057 = cnew constructWxStaticBoxSizer(orient = wxVertical, box = tmp274056)
tmp274058 = cnew constructWxTextCtrl(value = "Entry", parent = tmp274056, id = wxID_ANY)
gauge = cnew constructWxGauge(range = 100, parent = tmp274056, id = wxID_ANY)
tmp274057.add(gauge, border = 5, flag = wxExpand or wxAll)
tmp274051.add(tmp274057, proportion = 1, border = 5, flag = wxExpand or wxAll)
tmp274050.setSizer(tmp274051)
tmp274054.bind(wx.EVT_BUTTON, buttonCallback)
tmp274055.add(tmp274054, border = 5, flag = wxExpand or wxAll)
tmp274051.add(tmp274055, border = 5, flag = wxExpand or wxAll)
Macros – AST building

Hello World

Basic controls
- Button
- Checkbox

More controls
- Entry
Meta-programming in Nim

Peter Munch-Ellingsen

@PMunch – peterme.net