Questions & Answers

This page covers Q&As covering some of the software downloads, lecture resources on these pages and also questions we get asked from time to time.

calling native functions in LKit

Q: is it possible to execute program code or reference external data structures from LKit grammar rules?

A: yes. Using a "Lisp" tag, you can execute Lisp code and reference any Lisp structures loaded into the same Lisp world as LKit. Here is a simple example...

```
(build-grammar
  '((s (s -> n1 n2)
       (val . (lisp ($* n1 n2)))
    )))
```

persistent Java objects in NetLogo extensions

Q: I'm writing a NetLogo Java extension. I want to set up some data & refer to it each time I enter code for the extension – can I do this? Does it only work for simple types?

A: you are not limited to simple types in netlogo - you can return more or less anything (nlboris returns a boris portal to netlogo for example) but when it comes to accessing what is returned within NL it either (i) needs to be a type NL script supports (lists are often convenient for this) or (ii) you need to write accessors within the extensions.

For keeping data between calls to the java-side extension you can (i) pass a suitable instance back to NL (ii) create some persistent data object - possibly by having some "static" class/variables (iii) the NL extension manager & anything it creates is persistent so you can initialise other classes, etc within that.

iterating through an agentset in NetLogo

Q: I want to iterate though an agentset but NetLogo won't let me – what can I do?

A: change the agentset into a list using *sort* and then iterate through the list.

redirecting Java's System output

Q: can I redirect Java's System output & System error to a file?

A: yes you need to create a PrintStream from a file name then set output to use that stream, something like...

```
String filename = "D:\\debug.txt";
try
{    outStream = new PrintStream( fileName );
} catch (IOException e)
{    e.printStackTrace();
}
System.setOut( outStream );          // direct output to the file
System.out.println( "debug file starts" );
```

debug output from the Boris monitor

Q: can I get to read the output from print statements from agents loaded via the Boris monitor?

A: yes either by (i) redirecting System.out – see other question on this page or (ii) by calling the monitor from a command shell (rather than double-clicking boris.jar), eg: in windows call it using...

```
java -cp .;boris.jar boris.monitor.MasMonitor
```

tasks in NetLogo

Q: can NetLogo tasks be stored in variables etc

A: Yes. check out the link on tasks on the NetLogo pages.
[NB: we are not involved in supporting NetLogo – just interested!]

nlboris for NetLogo 5

Q: will nlboris work with NetLogo 5?

A: the current version of nlboris is for NetLogo 4, when NetLogo 5 is fully released we will release an update for nlboris. We currently provide a version which has been (partially) tested on NetLogo 5.0RC1 and 5.0RC2.
[tech note: all extensions require recompiling for NetLogo 5 and NetLogo 5 also has a modified interface for the LogoList class, we are also using upgrade to NetLogo 5 as an opportunity to add a couple of extra features to nlboris]

NLoops for NetLogo 5

Q: will NLoops work with NetLogo 5?

A: Yes but note: NLoops works with NetLogo trial release 5.0RC2 but does not work with the earlier trial 5.0RC1.
[tech note: we intend to develop a new release of NLoops to take advantage of the new task primitives provided in NetLogo 5]

syntactic agreement in LKit

Q: Can (context sensitive) syntactic agreements be included in LKit rules

A: Yes. We have added some notes about this on the Lkit pages. Check the part about word agreements & intersecting tags.

Lisp Resources

Q: do the Lisp resources work with versions of common lisp other than Allegro Lisp?

A: Yes – with the exception of *threads* (our multi-threading forms) and Boris in Lisp (which uses threads)
[tech note: Lisp does not have a standard form for multi-threading, we have written one for Allegro but cannot guarantee that this will work with other Lisp implementations]