Pudding Eater [1]

Eric Bailey

November 20, 2017 [2]

[1]

[2] Last updated November 20, 2017

Converting Node Identifiers

1b ⟨* 1a⟩+≡

```
(defun dot-name (exp)
  (substitute-if #\_ (complement #'alphanumericp) (prin1-to-string exp)))
```

Defines:
dot-name, used in chunks 1, 2, and 4.

Adding Labels to Graph Nodes

1c ⟨* 1a⟩+≡

```
(defparameter *max-label-length* 30)

(defun dot-label (exp)
  (if exp
      (let ((s (write-to-string exp :pretty nil)))
        (if (> (length s) *max-label-length*)
            (concatenate 'string (subseq s 0 (- *max-label-length* 3)) "...")
            s))
      ""))
```

Defines:
max-label-length, never used.
dot-label, used in chunk 2.

src/graphviz.lisp:

1a ⟨* 1a⟩≡

```
(in-package :cl-user)
(defpackage lol.graphviz
  (:use :cl :prove)
  (:export dot-name))
(in-package :lol.graphviz)
```

This definition is continued in chunks 1–3.
Root chunk (not used in this document).
Defines:
lol.graphviz, used in chunk 4.
Uses dot-name 1b.

Generating the DOT Information for the Nodes

2a ⟨* 1a⟩+≡

```
(defun nodes→dot (nodes)
  (mapc (lambda (node)
          (fresh-line)
          (princ (dot-name (car node)))
          (princ "[label=\"")
          (princ (dot-label node))
          (princ "\"];"))
        nodes))
```

Defines:
  nodes→dot, used in chunk 3a.
Uses dot-label 1c and dot-name 1b.

Converting Edges into DOT Format

2b ⟨* 1a⟩+≡

```
(defun edges→dot (edges)
  (mapc (lambda (node)
          (mapc (lambda (edge)
                  (fresh-line)
                  (princ (dot-name (car node)))
                  (princ "→")
                  (princ (dot-name (car edge)))
                  (princ "[label=\"")
                  (princ (dot-label (cdr edge)))
                  (princ "\"];"))
                (cdr node)))
        edges))
```

Defines:
  edges→dot, used in chunk 3a.
Uses dot-label 1c and dot-name 1b.

Generating All the DOT Data

3a ⟨* 1a⟩+≡

```
(defun graph->dot (nodes edges)
  (princ "digraph{")
  (nodes->dot nodes)
  (edges->dot edges)
  (princ "}"))
```

Defines:
  graph->dot, used in chunk 3c.
Uses edges->dot 2b and nodes->dot 2a.

Turning the DOT File into a Picture

3b ⟨* 1a⟩+≡

```
(defun dot->png (fname thunk)
  (with-open-file (*standard-output*
                   fname
                   :direction :output
                   :if-exists :supersede)
    (funcall thunk))
  (uiop:run-program (concatenate 'string "dot -Tpng -O " fname)))
```

Defines:
  dot->png, used in chunk 3c.

Creating a Picture of Our Graph

3c ⟨* 1a⟩+≡

```
(defun graph->png (fname nodes edges)
  (dot->png fname
            (lambda ()
              (graph->dot nodes edges))))
```

Defines:
  graph->png, never used.
Uses dot->png 3b and graph->dot 3a.

Tests

4 ⟨*test/graphviz.lisp* 4⟩≡

```
(in-package :lol.graphviz)


(plan 1)


(subtest "Converting Node Identifiers"
  (is (dot-name 'living-room)
      "LIVING_ROOM")
  (is (dot-name 'foo!)
      "FOO_")
  (is (dot-name '24)
      "24"))


(finalize)
```

Root chunk (not used in this document).
Uses dot-name 1b and lol.graphviz 1a.

References