02257 Applied functional programming

Functional Pearls: Drawing Trees in F#

Anh-Dung Phan, Januar 2014
DTU Compute, Technical University of Denmark
Revised 13-01-2016, 14-01-2017, 06-01-2019, 04-01-2020 by Michael R. Hansen

Tasks

This project based on a functional pearl for drawing general trees by Andrew Kennedy [1]. The article by Kennedy describes a program solution to the layout af trees that is expressed in the functional programming language SML. Your tasks are described as follows:

1. Translate the SML solution into F#.
2. Create a function to convert a general tree to a string in PostScript format and one to write results to PS files. For more information about PostScript, take a look at the appendix below. Aim at a modular design that making it is to experiment with different implementations, especially concerning the handling of strings. (See point 6.).
3. Create functions translating Abstract Syntax Trees (AST) from the compiler project to general trees. The file `AST.fs` handed out with contains all the type declarations for the abstract syntax trees of the compiler project (Project 2). Write a few interesting ASTs to PS files. Inspect the PS files using, for example, a PostScript viewer, for example, GhostView.
4. Declare one or more functions to generate general trees of given sizes, where the size may depend on the width and the depth of a tree.
5. Consider next the part that translates a general trees into PostScript:
   - Analyze the efficiency of the translation.
   - Experiment with the performance of your translation on trees of varying sizes. You could use `#time "on";;` directive in F# Interactive for this purpose.
6. As natural extensions, you can consider solving the following problems:
   - Repeated concatenation of strings using `+` may be inefficient. Compare different string concatenation methods via performance measurement. Use for example the `String.concat` function and/or the `System.Text.StringBuilder` class.
   - Overlapping between long labels could possibly happen. How do/would you cope with that?

Submission and Demo

To submit the solution to this project, you must:

1. give a demo Wednesday, January 8, from 14:00-17:00, and
2. submit your project on Inside **no later than 18:00 on Wednesday, January 8**.

Note that: **All group members must be available at the demo**.

You should hand in two files on Inside:

- A zip-archive containing your working solution.
- A pdf-file with at most 3 pages including the front page, containing:
    1. A front page with names and study numbers of the members in the group. The group members account for (1) the working solution is made by the group members only, (2) no part of their solution is distributed to other groups, and (3) the group members have contributed equally to the solution.
    2. Unless it is otherwise stated, it is expected that all group members can account for all parts of the solution.
    3. The following pages should clearly describe the status of the solution. In particular, it must be stated which input trees your solution can handle. Furthermore, it should describe your own extensions/additions. It must be possible to verify this status by executing your Script.fsx file. There should be a brief section with reflections over the project.

Appendix: Drawing trees in the PostScript format

The language PostScript is used for representing printed pages (containing graphics). A curve consisting of the segments joining the points (10.0, 20.0), (40.0, 50.0) and (70.0, 80.0) may e.g. be represented by the following PostScript instructions:

```
%!
1 1 scale
newpath
10 20 moveto
40 50 lineto
70 80 lineto
stroke
showpage
```

Apart from the initial and final configurations (to please the PostScript interpreter) there has to be a PostScript *moveto* command containing the coordinates of the start point plus PostScript *lineto* commands containing the coordinates of each subsequent point of the curve.

Note that the coordinates are positive integers. You will hence have to make a conversion from real-valued coordinates to integers, and the parts of a tree corresponding to negative coordinate values can not be printed.

A file in PostScript format may also be shown on the screen using the ghostview program in an xterm:

ghostview *filename* &

(On Windows, you may convert ps files to pdf files using Adobe Acrobat XI.)

You also need to set appropriate page sizes and display strings. *View the companion fact1.ps file in a text editor to see the concrete syntax for doing so.* For reference and tutorials about PostScript, please refer to its Wikipedia page [2].

References

[1] Andrew J. Kennedy. Functional pearls. *Journal of Functional Programming*, 6:527–534, 1996.

[2] Wikipedia. PostScript. `http://en.wikipedia.org/wiki/PostScript`, 2012.