

Class 10: Representing Images as Lists of Spots

Held: Wednesday, 6 February 2008

Summary: We consider one simple representation of images in which we list the positions and colors of the colored spots in an image. Along the way, we explore *lists*, the most important structured data type in Scheme.

Related Pages:

- EBoard.
- Lab: Representing Images as Lists of Spots.
- Reading: Representing Images as Lists of Spots.
- Due: Assignment 3: Algorithmically Creating Simple Images.

Notes:

- Reminder: Quiz 2 on Friday (some code reading, some code writing).
- Reminder: Exam 1 distributed on Friday.
- Reminder: Purcell convocation tomorrow.
- Reading for Friday: Documenting Procedures.

Overview:

- Lists in Scheme.
- Basic list operations.
- Spot lists: Another perspective on images.

Lists

- Lists are a core aspect of Scheme and LISP.
- Lists are ordered collections of information.
- Dynamic: Lists can "grow" and "shrink"
 - Or at least you can build larger or smaller lists from other lists.
- Lists are *pure*: You build new lists from old; you don't modify the previous list.
- Lists look a lot like procedure application (intentionally).
- The simplest list: the empty list is `null` or `()`.
- You extend lists at the front (or, more precisely, build new lists) with

`(cons thing-for-the-front rest-of-the-list)`
- For example,

```
> (cons 'a null)
(a)
> (cons 'b (cons 'a null))
(b a)
```

- You can also create lists with `list`.

```
> (list 'a 'b 'c)
(a b c)
```

- You extract parts of lists with three operations (we'll reflect on the naming later in the semester):
 - `car`: get the initial element of the list.
 - `cdr`: get all but the initial element of the list.
 - `list-ref`: get the *i*th element of the list.
- One disadvantage: Scheme lists look a lot like procedure applications, so it's sometimes hard for beginning programmers to tell what's going on.
 - You can always tell by context, but it takes some time to get used to the context.
- We'll look at some behind-the-scenes issues in a few weeks.

Spot Lists

- Goal: Design our own implementation of images.
- Starting point: Little spots on the screen.
- Idea: Represent the spot as a list.
- Followup idea: Represent an image as a list of spots.
- Today we'll focus on the first of the two ideas.
- Later (in a week or so), we'll look at how you deal with lists of spots.

Copyright © 2007-8 Janet Davis, Matthew Kluber, and Samuel A. Rebelsky. (Selected materials copyright by John David Stone and Henry Walker and used by permission.) This material is based upon work partially supported by the National Science Foundation under Grant No. CCLI-0633090. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. This work is licensed under a Creative Commons Attribution-NonCommercial 2.5 License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc/2.5/> or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.