Elixir Lists

Extended Cut

Lists - Review

- Lists are collections of values of any type
- List values are not stored in contiguous memory
- Each list item points to each subsequent element
- Always put new list items to the front (head) of a list

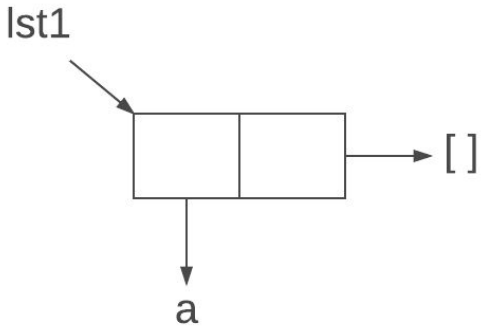
Turtles all the way down...sorta

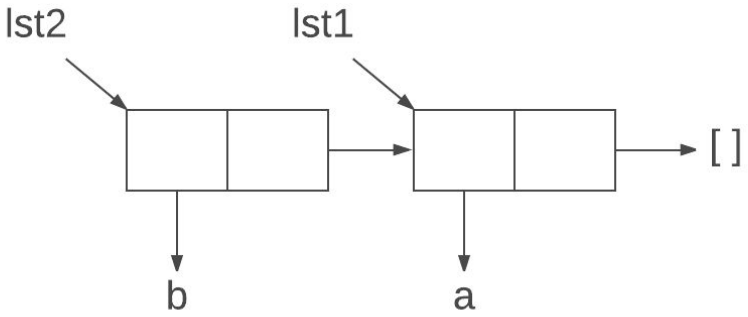
- Understand a single item list → understand lists
- Elixir lists are composed of lists
  - ie: list of lists
- Empty list is baseline for all lists → `[]`
  - No head, no tail, just an empty list
- Remember: **|** is “cons” or the cons cell

Not a “con job” - this is a job for **cons**

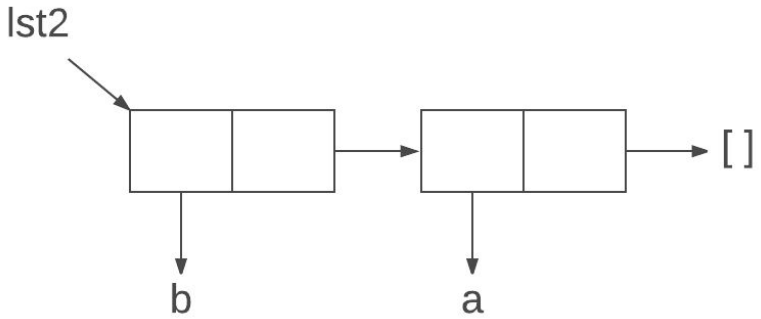
- Cons cells harken back to the Lisp programming language
- Basic building block of FP lists; each list node is a cons cell
- Cons cell made of two parts: **car** (or **first**) and **cdr** (or **rest**)
  - car points to value node is “holding”
  - cdr (pronounced “could-er”) points to next cons cell in list or empty list - []

Con cells visualized





or



Give that list an item!

- Consider the nature of cons-based lists
- *Prepend* always preferred over *append*
- Moar practice . . . *GO!*

```
my_list = []
my_list = [1|[]]
my_list = [2|[1|[]]]
my_list = [3|[2|[1|[]]]]
[3|[2|[1|[]]]] === [3,2,1]    # true
```

Your Turn!

- Practice building lists in IEx
- Build new lists using cons
- Consider: why is prepending ok but not appending?
- Share your answer or thoughts in our online discussion.

- Enjoy your new understanding, it will come in handy frequently.