Learn Rholang Tutorial

Tuesday, February 12, 2019 8:12 PM

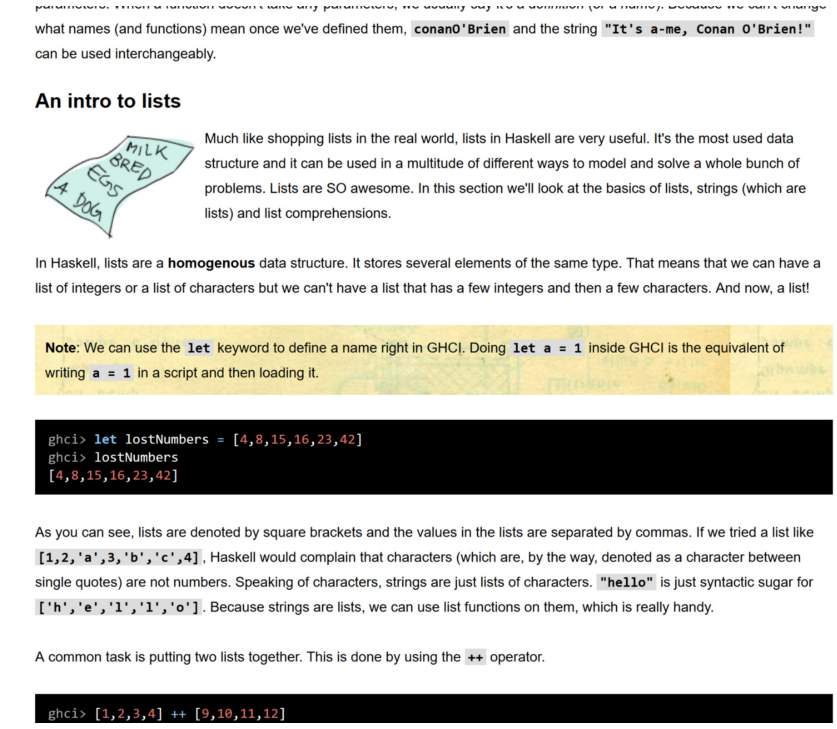
Goal

I've written a tutorial for the rchain coop's flavor of rholang (coop rholang). The content is written in markdown and lives https://github.com/JoshOrndorff/LearnRholangByExample . There is currently nowhere to read the content in a user-friendly way.

- Display the content nicely as described later in the doc
- Allow the content to be regularly upgraded through a git workflow
- Make the site light-weight

Inspiration

My tutorial's style was largely inspired by a similar haskell tutorial http://learnyouahaskell.com/chapters



what names (and functions) mean once we've defined them, `conanO'Brien` and the string `"It's a-me, Conan O'Brien!"` can be used interchangeably.

An intro to lists

Much like shopping lists in the real world, lists in Haskell are very useful. It's the most used data structure and it can be used in a multitude of different ways to model and solve a whole bunch of problems. Lists are SO awesome. In this section we'll look at the basics of lists, strings (which are lists) and list comprehensions.

In Haskell, lists are a **homogenous** data structure. It stores several elements of the same type. That means that we can have a list of integers or a list of characters but we can't have a list that has a few integers and then a few characters. And now, a list!

Note: We can use the `let` keyword to define a name right in GHCI. Doing `let a = 1` inside GHCI is the equivalent of writing `a = 1` in a script and then loading it.

```
ghci> let lostNumbers = [4,8,15,16,23,42]
ghci> lostNumbers
[4,8,15,16,23,42]
```

As you can see, lists are denoted by square brackets and the values in the lists are separated by commas. If we tried a list like `[1,2,'a',3,'b','c',4]`, Haskell would complain that characters (which are, by the way, denoted as a character between single quotes) are not numbers. Speaking of characters, strings are just lists of characters. `"hello"` is just syntactic sugar for `['h','e','l','l','o']`. Because strings are lists, we can use list functions on them, which is really handy.

A common task is putting two lists together. This is done by using the `++` operator.

```
ghci> [1,2,3,4] ++ [9,10,11,12]
```

I'd like the code to be inline and syntax highlighted just like it is in the haskell tutorial. Syntax highlighting could be done with https://github.com/rchain-community/codemirror-rholang

Quizzes

Must have

The answer to the quiz questions shouldn't be shown until the user submits a guess.
Simple choose the correct answer questions.

Nice to have

Multiple correct answers.
Show feedback/hint on wrong answer.

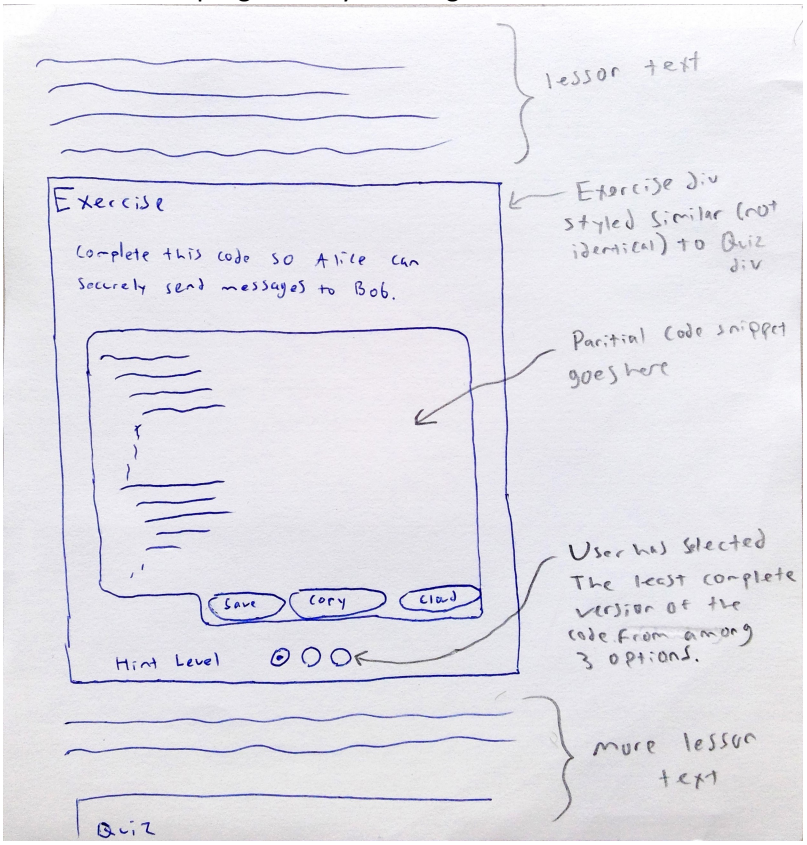
Exercises

Must have

Specific styling for exercises.
An exercise will have a text description, usually some starter code, and an example solution example solution should not be shown until requested (a simple link and page load is fine)

Nice to have

An interface for progressively showing more and more of a solution.



Previous attempts

A few people have made quick attempts to make this tutorial look nice, but I haven't followed up enough to get any of them fully working with my write in markdown commit via git workflow.

- https://github.com/rchain-community/rholang-lessons
- Pieter's attempt
- There was also a similar rholang tutorial
  https://web.archive.org/web/20180821160158/https://developer.rchain.coop/tutorial/
  It doesn't look good on the archive, but it can be built using
  https://github.com/rchain/rchain/tree/dev/rholang-tutorial