Elixir

Levi Moore
(levimoore@email.arizona.edu)

Benhur Tadiparti
(tadiparti@email.arizona.edu)

April 7, 2020

1 Abstract

2 Introduction

3 History

The Elixir programming language was created by José Valim, the co-founder of Plataformatec, and member of the Ruby on Rails Core Team. He aimed to create a programming language for large-scale sites and apps by combining the best features of Ruby, Erlang, and Clojure.

Elixir was designed to handle large data volumes. The speed and capabilities spread Elixir into telecommunication, eCommerce, and finance industries. (Wiki)

Elixir is definitely not dead nor barely alive due to the avid number of Elixir users, however, compared to big programming languages like Python and Java - Elixir is in-between alive and barely moving.

4 Control Structures

In a program, a control-structure determines the order in which statements are executed. Elixir supports three control-structures. These include

'if and unless', 'case' and 'cond'. In this section these three control-structures will be explored individually, code snippets of each will be included for reference.

In programming, the most well-known control-structure would have to be the if-statement. In simple terms, if a statement is proven true the code below is executed, otherwise a different block of code is executed for when its proven false. Secondly, Elixir also supports unless, this could be thought of as an if statement that works negative. Below is a snippet of how the if-statement is used in Elixir.

SNIPPET HERE:

```
if x == true do
    IO.puts("Hello User!")
unless x == null do
    IO.puts("User might be on the wrong computer")
else
    IO.puts("Unauthorized Access!")
end
```

What determines what we do as humans every day is what day of the week it is. On Sundays we get ready for the work or school week on Monday, whereas on Fridays we get ready to relax and take It easy for a couple days. This is a real-life example of the case control-structure that Elixir supports. This control-structure allows us to compare a value against various patterns until we find the case that suits it and branch into that direction. Below is an example of how the case control-structure is used in Elixir.

SNIPPET HERE:

```
case {"Hi", "Hello"} do
    {"Hi", x} ->IO.puts("The computer response woud be to say 'Hello'")
    {"What", "Excuse Me"} ->IO.puts("That was not a greeting")
end
```

When we need to check many different conditions and find one that evaluates to true, its best to use the cond control-structure. Cond allows us to enter as many conditions as we want to check, similar to an else-if in other languages, in order to evaluate to true. In some cases, all conditions will evaluate to false, so we must add a true condition at the end, similar to an

else statement, to end the conditional. This is useful when we have various conditions needing to be tested. Below is an example of how the cond control-structure is used in Elixir.

SNIPPET HERE:

```
cond do
    1 + 0 == 2 ->IO.puts "This is also false"
    5 * 10 == 0 ->IO.puts "This is false"
    true ->IO.puts "Hello world"
end
```

5 DataTypes

6 Subprograms

7 Summary

8 References

References

[1] Jan Dudulski. How we learned elixir—our story and tutorial for beginners, January 2019. Last accessed 6 April 2020.

[2] elixir. elixir, 2011. Last accessed 6 April 2020.

[3] Reddit. The elixir programming language, November 2012. Last accessed 6 April 2020.

[4] Elixir School. Basics. Last accessed 6 April 2020.

[5] tutorialspoint. Learn elixir. Last accessed 6 April 2020.

[6] Wikipedia. Elixir (programming language), 2011. Last accessed 6 April 2020.

[2] [5] [1] [4] [3] [6]