Elixir in a nutshell

Summary

- What is Elixir?
- Hello world in Elixir.
- What is functional programming?
- What is the Erlang VM?
- Why use Elixir?
- Code examples.
- Conclusion.
- References.

What is Elixir?

Elixir is a programming language that was created in 2011 by José Valim. It is a functional, dynamic language that is built on top of the Erlang virtual machine (EVM). Just like Java, a language that is also built on top of a virtual machine (JVM), it is portable and it can run on any operating system. Elixir is known for its simplicity, concurrency, and scalability.

Hello world in Elixir.

```
 # the basic syntax
IO.puts("Hello, World!")
# the ruby inspired syntax
IO.puts "Hello, World!"
```

What is functional programming?

Functional programming is a programming paradigm in which functions are the primary unit of computation. In functional programming, functions are first-class citizens, which means that they can be passed around and used just like any other value. This allows for concise, elegant code that is easy to reason about and test.

Functional programming vs Object-oriented programming

We have already talked about what functional programming is but what is Object-oriented programming?

Object-oriented programming (OOP) is a way of organizing and designing computer programs. In OOP, you write "objects" that represent real-world things and ideas. These objects contain data and can perform actions. OOP makes it easier to create and maintain complex programs.

Some of the main differences between functional programming and object-oriented programming include:

- Data mutability: In functional programming, data is immutable, while in OOP it is mutable.
- Functions: In functional programming, functions are first-class citizens, while in OOP they are associated with objects.
- Code organization: In functional programming, code is organized around functions, while in OOP it is organized around objects and their interactions.
- Problem-solving approach: Functional programming often uses recursion to solve problems, while OOP uses inheritance and polymorphism.

What is the Erlang VM?

The Erlang VM (also known as the BEAM) is a virtual machine that was designed specifically to run concurrent, fault-tolerant programs. The VM is highly efficient and can handle millions of concurrent connections with minimal overhead. This makes it ideal for building distributed systems and applications that need to be available 24/7.

Why use Elixir?

There are several reasons to consider using Elixir for your next project:

- Concurrency: Elixir makes it easy to write concurrent programs that can take full advantage of multicore systems.
- Elegant syntax: Elixir has a simple, readable syntax that makes it easy to write expressive code.
- Scalability: Elixir's support for concurrency and its integration with the Erlang VM make it easy to scale applications horizontally.
- Portability: Elixir can run on any operating system.

Code examples

```
 # This function reads a number from a file
# and returns it
def read() do
  {age, _} = IO.gets("Give: ") |> Integer.parse()
  age
end
```

```
 # This function finds the factorial of a given number n
def factorial(n) do
      cond do
        n > 0 -> n * factorial(n-1)
        true -> 1
    end
end
```

```
 # This module called Math_stuff contains two function, one public and one private
defmodule Math_stuff do
  def sum(a, b) do
    do_sum(a, b)
  end

  defp do_sum(a, b) do
    a + b
  end
end
```

Conclusion

Elixir is a powerful, functional programming language that runs on the efficient Erlang VM. Its support for concurrency and fault tolerance make it ideal for building scalable, distributed systems. Its simple syntax and expressive power make it a joy to use.

References

https://hexdocs.pm/elixir/Code.html
https://www.tutorialspoint.com/elixir/elixir_overview.htm
https://en.wikipedia.org/wiki/Functional_programming
https://en.wikipedia.org/wiki/Object-oriented_programming