

Elixir Lesson 2018/7/10

アジェンダ

- これまでの復習
 - 環境構築とmix,iexのインストール
 - モジュールと関数
- Elixirの型を学ぶ
 - 文字列型
 - 数値型
 - 真偽値
 - Atom型
 - List型
 - Tuples型
 - Anonymous Function型(無名関数)

このレッスンの所要時間

- 1時間

Elixirの型を学ぶ

文字列型

- 特徴
 - Elixirでは、ダブルクォート(")で囲った値が 文字列型(String)の値として扱われます
 - 文字列は UTF-8にエンコードされます
- サンプルコード

```
iex> str0 = "A"  
"A"  
  
iex> is_bitstring(str0)  
true  
  
iex> bit_size(str0)  
8
```

```
iex> str1 = "hello"  
"hello"  
  
iex> is_bitstring(str1)  
true  
  
iex> bit_size(str1)  
40  
  
iex> str2 = "こんにちは"  
"こんにちは"  
  
iex> is_bitstring(str2)  
true  
  
iex> bit_size(str2)  
120
```

数値型

- 特徴
 - 数値型はElixirの基本型の1つ
 - 整数値と浮動小数値を表すことが可能
 - 数値型は、10進数、16進数がありそれぞれ表現可能
- サンプルコード

```
iex 1> 1  
1  
iex 2> is_integer(1)  
true  
iex 3> 0x1F  
31  
iex 4> is_integer(0x1F)  
true  
iex 5> 1.0  
1.0  
iex 6> is_float(1.0)  
true
```

真偽値型

- 特徴

- 真偽値型はElixirの基本型の1つ
 - 真(true)と偽(false)の2つの値を取り、Elixir内で定義されている
- サンプルコード

```
iex> true
true
iex> true == false
false

iex> is_boolean(true)
true

iex> is_boolean(false)
true

iex> age = 20
true

iex> is_smoking_allowed_jp = (age >= 20)
true

iex> is_boolean(is_smoking_allowed_jp)
true
```

Atom

- 特徴
 - 数値に名前をつけられる
 - true/falseもatom, クラス名はatomとして管理されている
- サンプルコード

```
iex> true == :true
true
iex> is_atom(false)
true
iex> is_boolean(:false)
true

iex> is_atom(Tuple)
true

iex> is_atom(File)
true
```

List型

- 特徴
 - Elixirの基本型の1つ
 - リスト型の値の書き方
 - 角括弧 `[]` で囲った範囲にカンマ区切りで要素を記載する
 - リスト内の要素には、どのような型の値でも含めることが可能
 - 数値型、文字列型、真偽値型, など
- サンプルコード

```
iex> [1, 2, true, 3]
[1, 2, true, 3]

iex> [1, 2, 3] |> length
3

iex> [1, 2, 3] ++ [4, 5, 6]
[1, 2, 3, 4, 5, 6]
iex> test_list_123 = [1, 2, 3]
iex> test_list_456 = [4, 5, 6]
[1, 2, 3, 4, 5, 6]

iex> [1, true, 2, false, 3, true] -- [true, false]
[1, 2, 3, true]

iex> listTest = [10, 5, 1]
[1, 2, 3]
iex> Enum.at(listTest, 0)
10
iex> Enum.at(listTest, 1)
5
iex> Enum.at(listTest, 2)
1
iex> Enum.at(listTest, 3)
nil
```

Tuples

- 特徴
 - Elixirの基本型の1つ
 - タプル型の値の書き方
 - 波括弧 `{ }` で囲った範囲にカンマ区切りで要素を記載する
 - タプル内の要素には、どのような型の値でも含めることが可能

- 数値型、文字列型、真偽値型, など
 - タプル内の各要素に添字を用いてアクセスすることが可能
 - 添字は0から始まる
 - メモリアドレス上で連続した場所に格納される
 - 添字を用いての要素アクセス、サイズの取得処理が高速
- サンプルコード

```
iex(1)> tuple = {:v1, :v2, :v3}
{:v1,:v2,:v3}
iex(2)> tuple
{:v1,:v2,:v3}
iex(3)> tuple_size(tuple)
3
iex(4)> elem(tuple,1)
:v2
iex(5)> elem(tuple,2)
:v3
iex(6)> tuple = {:a, :b, :c, :d}

iex(7)> put_elem(tuple, 2, :e)
{:a, :b, :e, :d}
```

Anonymous Function(無名関数)

- 関数とは
 - 入力を与えると、出力が得られる装置
 - 入力とは
 - 出力とは
 - 関数に何らかの入力xを与えて実行すると、何らかの出力yが得られる
 - 参照: 入力xを与えたらx に100を足した値を出力する関数を考える
 - 関数を実行するには
 - まず関数を定義する
 - 定義した呼び出す(ここで実行される)
- 特徴
 - Elixirでは関数も1つの型として規定されている
 - 文字列や数値など他の型を扱う際と同じレベルで、`関数` 値を表現する方法を学ぶ
- サンプルコード
 - 入力xを与えたらx に100を足した値を出力する関数を考える

```
iex(1)> x = 42  
iex(2)> f = (fn (value) -> value + 100 end) # 変数fに `関数` 値を格納する  
iex(2)> x = f.(x) # 関数を実行する(`関数` 値を格納した変数f を実行する)  
iex(3)> x # 何が表示されるか
```