Basic Types

```
1           # integer
0x1F        # integer
1.0         # float
true        # boolean
:atom       # atom / symbol
"elixir"    # string
[1, 2, 3]   # list
{1, 2, 3}   # tuple
```

Identifying functions

- Identified by their name and their arity
- Arity is the number of arguments that a function takes
- `div\2` `round\1`

Atoms

- Atoms are constants whose name is its own value

```
iex> :hello
:hello
iex> :hello == :world
false
iex> true == :true
true
iex> false == :false
true
```

Strings

- Double quotes
- UTF-8 encoded

```
iex> "hellö #{:world}"
"hellö world"
iex> IO.puts "hello\nworld"
hello
world
:ok
```

Anonymous functions

- first class citizens
- closures
- `.` removes ambiguity

```
iex> add = fn a, b -> a + b end
#Function<12.71889879/2 in :erl_eval.expr/5>
iex> add.(1, 2)
3
```

lists

- immutable
- values of any Type

```
iex> [1, 2, 3] ++ [4, 5, 6]
[1, 2, 3, 4, 5, 6]
iex> [1, true, 2, false, 3, true] -- [true, false]
[1, 2, 3, true]
```

Charlist

- list of printable ASCII numbers

```
iex> [11, 12, 13]
'\v\f\r'
iex> [104, 101, 108, 108, 111]
'hello'
iex> 'hello' == "hello"
false
```

Tuples

- store elements contiguously in memory.
- accessing a tuple element by index or getting the tuple size is a fast operation

```
iex> tuple = {:ok, "hello"}
{:ok, "hello"}
iex> elem(tuple, 0)
:ok
```

- immutable

```
iex> tuple = {:ok, "hello"}
{:ok, "hello"}
iex> put_elem(tuple, 1, "world")
{:ok, "world"}
iex> tuple
{:ok, "hello"}
```

Resources

- Elixir Docs - Basic types