



Programação Funcional com Elixir

The background of the slide features a series of overlapping, semi-transparent gray shapes that resemble stylized leaves or petals. These shapes are layered to create a sense of depth and movement, primarily concentrated on the left side of the frame. The word "Listas" is positioned in the lower-left area, partially overlapping these gray shapes.

Listas

Listas

- As listas são delimitadas por colchetes e elas podem conter tipos diferentes, veja:
 - `[43, :yes, "hello", 67.32, true]`
- Listas podem ser concatenadas com "++" ou subtraídas com "--"
 - `[43, :yes, "hello"] ++ [67.32, true]`
 - `[43, :yes, "hello"] -- ["hello", true]`

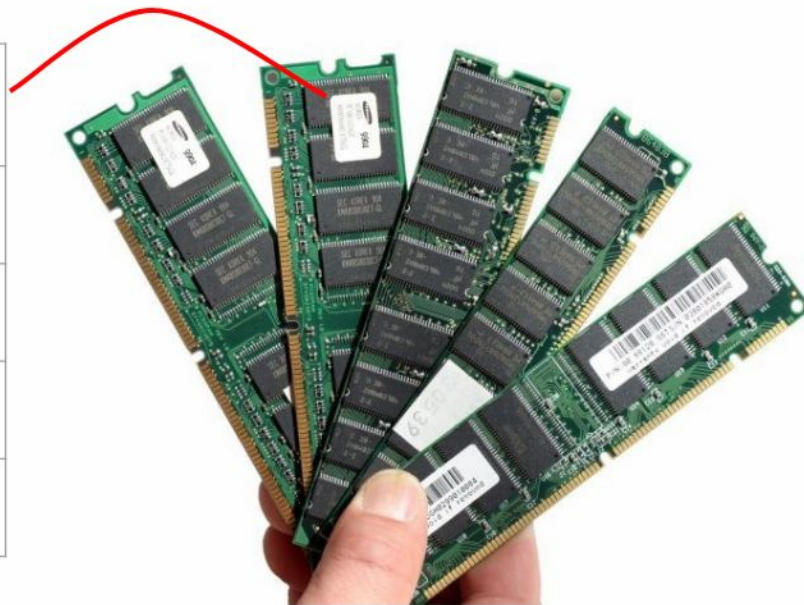
Listas

- Listas no Elixir são **Listas Encadeadas** em sua essência, sendo assim os elementos não são indexados e não podemos acessar um elemento diretamente como em um array/vetor. Vamos voltar um passo para entender esse conceito.

Listas

- O que são variáveis? Apenas uma célula de memória reservado, no qual atribuímos um nome.

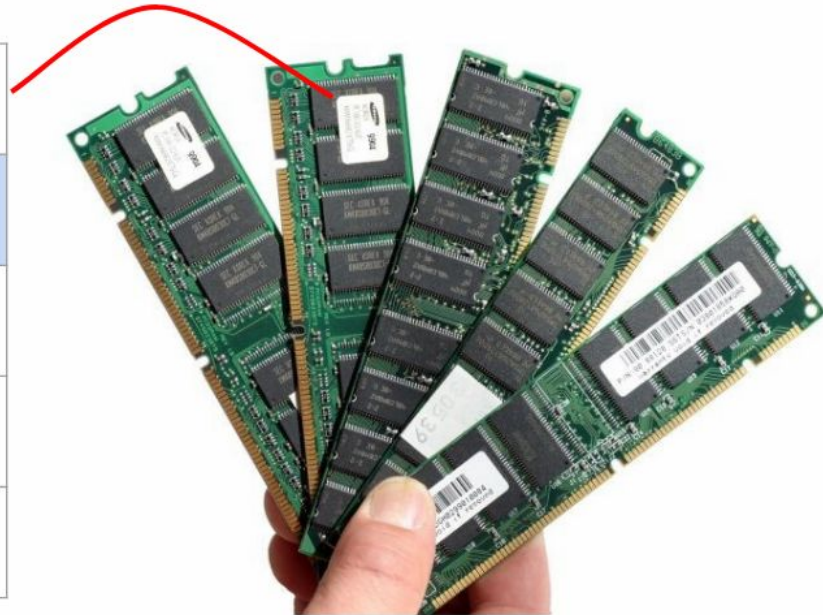
FFF1A	FFF1B : N1 32	FFF1C	FFF1D : S1 <i>"Jackson"</i>	FFF1E
FFF1F	FFF2A	FFF3A	FFF4A	FFF5A
FFF6A	FFF6A	FFF8A	FFF8B	FFF4B
...
...



Listas

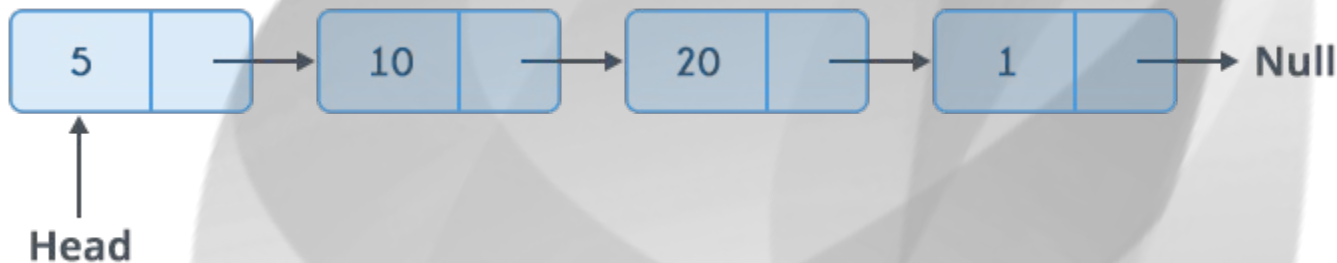
- A Lista Encadeada parte do princípio de que podemos ligar/encadear células de memória.

FFF1A Valor: "videos" Próx: FFF3A	FFF1B	FFF1C	FFF1D	FFF1E
FFF1F	FFF2A	FFF3A Valor: "de" Próx: FFF6A	FFF4A	FFF5A Valor: 2018 Próx: ..
FFF6A Valor: "ti" Próx: FFF5A	FFF7A	FFF8A	FFF8B	FFF4B
...
...



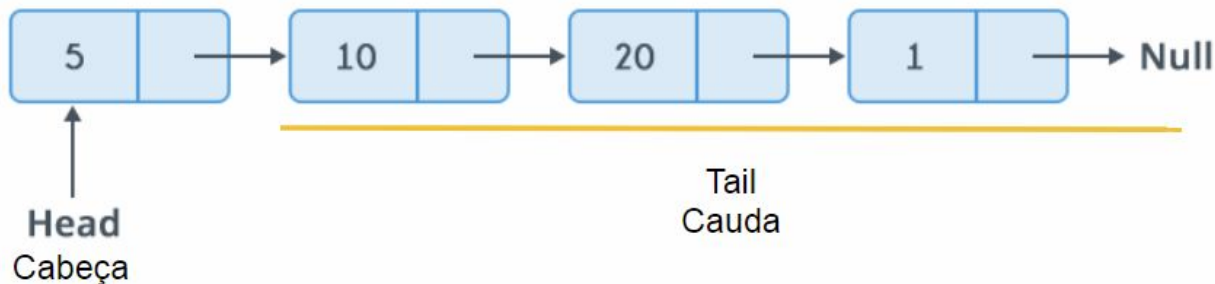
Listas

- Normalmente, a representação para as listas de forma resumida é:



Listas

- Agora que conhecemos como funcionam as listas no Elixir, podemos conhecer duas importantes funções a `head` `hd/1` e a `tail` `tl/1`.
- Essas funções "devolvem" a cabeça e a cauda de uma lista, respectivamente.



Listas

- Exemplo:
 - `hd([43, :yes, "hello", 67.32, true])`
 - `tl([43, :yes, "hello", 67.32, true])`