

MURPHY

“A Lei de Murphy não significa que algo ruim vai acontecer. Isso significa que tudo o que pode acontecer, acontecerá.”- Interstellar



POR QUE MURPHY?

A inspiração do filme Interstellar e Murphy é o nome da personagem principal que o nome tem inspiração na lei de murphy. Com isso, resolvi criar uma linguagem onde o tempo é uma dimensão fundamental para a execução.

CARACTERÍSTICAS

Busca uma experiência de programação mais expressiva, poética e criativa. Além disso, explorar como conceitos abstratos como o tempo podem ser traduzidos em estruturas de código.

SINTAXE POÉTICA E INTUITIVA

Comandos temáticos que evocam a exploração espacial e temporal.

BLOCOS TEMPORAIS

`past { ... }`: Define ações ou estados que "já aconteceram" ou são pré-condições.

`now { ... }`: O fluxo principal de execução, o presente momento da computação.

`future { ... }`: Define ações que "irão acontecer" ou são projeções.



CARACTERÍSTICAS

MANIPULAÇÃO DA “REALIDADE”

Comandos temáticos que evocam a exploração espacial e temporal.

manifest: "Materializar" variáveis, trazê-las à existência.

adjust: "Ajustar" os valores, como calibrar instrumentos.

disintegrate: "Desintegrar" variáveis, removendo-as da existência.

emit: "Emitir" sinais ou mensagens, comunicando-se através do tempo/espço do código.

NAVEGANDO PELO ESPAÇO-TEMPO

loop_horizon: Laços para explorar eventos repetitivos ou horizontes de eventos.

distort + fallback: Condicionais que "distorcem" o fluxo normal, com um caminho alternativo (fallback) se a distorção não for aplicável.



CURIOSIDADES SOBRE A LINGUAGEM MURPHY

01

A ORIGEM DO NOME

Começou como “InterestelarLang, uma homenagem direta ao filme mas depois para Murphy pois o código tem conexão com a Lei de Murphy abraçando o conceito de imprevisibilidade

02

LINGUAGEM TEMÁTICA PROFUNDA


Não apenas os blocos temporais, mas cada palavra-chave (manifest, disintegrate, loop_horizon, distort) foi escolhida para reforçar a imersão no tema.

03

EXPERIMENTAL POR NATUREZA

O objetivo não é competir com linguagens de produção, mas explorar novas formas de expressão e paradigmas.

EXEMPLO DE CÓDIGO



```
past {  
  manifest signal with 1;  
  emit "Mensagem enviada ao passado";  
}  
  
now {  
  adjust signal by 2;  
  emit "Ajuste realizado no presente";  
}  
  
future {  
  loop_horizon i below 3 {  
    emit "Mensagem vinda do futuro";  
  } end_loop  
}
```



```
Mensagem enviada ao passado  
Ajuste realizado no presente  
Mensagem vinda do futuro  
Mensagem vinda do futuro  
Mensagem vinda do futuro
```

OBRIIGADA!

