

# Bizarrices *Just in Time*

Como você não deve projetar uma linguagem de programação

Marcelo "Haskell" Camargo  
[marcelocamargo@linuxmail.org](mailto:marcelocamargo@linuxmail.org)  
[github.com/haskellcamargo](https://github.com/haskellcamargo)

# Sobre o Autor



Criador da linguagem de programação *Capybara*, uma linguagem fortemente tipada com tipos dependentes que compila para ZPL.

Tradutor e revisor de projetos abertos e gratuitos, como phpMyAdmin, CakePHP, WPS Office, Bitbucket, Geany e artigos da Mozilla Foundation.

# Sumário

- 1 → Erros de análise
- 2 → Erros em sistemas de tipos
- 3 → Coisas que não deveriam estar lá
- 4 → Incoerência na nomenclatura
- 5 → Erros comuns (ou nem tanto)
- 6 → Bônus!

# 1 → Erros de análise

1.1 · Associatividade de operadores

1.2 · Avaliação sintática de expressões

1.3 · *Weak string interpolation*

1.4 · *Shut up operator*

1.5 · *Trailing **else***

## 1.1 → Associatividade de operadores



```
echo true  
  ? "car"  
  : false  
  ? "horse"  
  : "feet";
```

## 1.2 → Avaliação sintática de expressões

```
// Acesso de índice a partir do retorno  
// de uma função.  
// PHP < 5.4  
some_function_call()[10];  
  
// Utilização do retorno da expressão para  
// chamada de função anônima.  
// PHP < 7.0  
(function($x) {  
    return $x * 2;  
} )(10);  
  
// Gambiarra:  
$doubleMe = function($x) { return $x * 2; }  
$doubleMe(2);
```

## 1.3 → *Weak string interpolation*

```
$x = new Person("Tibúrcio");  
echo "Hello, $x->name!";  
// Object of class Person could not  
// be converted to string
```

```
$x = new Person("Tibúrcio");  
echo "Hello, {$x->name}";  
// Hello, Tibúrcio!
```

```
Person person = new Person("Tibúrcio");  
Console.WriteLine("Hello, {0}", person.name);
```





## 1.4 → *Shut up operator*

```
echo @$nobody_cares;
```

```
echo @@@@@@@@@@@@@@@@@@  
@@@@@@@@@@@@@@@@@@@@  
@@@@@@@@@@@@@@@@@@@@  
$nobody_really_cares_about_warning
```

```
$x = @$undefined_variable;  
echo @(&$x + 1);  
// ￣\_ (ツ) _/￣
```





## 1.5 → *Trailing else*

```
if (booleanValue)
  if (anotherValue)
    doX()
  else doY();
//else doY() ?
```

```
if boolean_value:
  if another_value:
    do_x()
else:
  do_y()
```

# 2 → Erros em sistemas de tipos

## 2.1 · Inferência e coerção em sistemas de tipos fracos

## 2.1 → Inferência e coerção em sistemas de tipos fracos

```
[] + [] ; // ""  
[] + 1 ; // "1"  
1 + {} ; // "1[object Object]"  
{ } + { } ; // NaN  
{ } + [] ; // 0!?  
Array(10).join("abc" - 1) + ", Batman! Batman!";
```



## 2.1 → Inferência e coerção em sistemas de tipos fracos

```
"1 cachorro" + "2 gatos" == "3 mamíferos";  
// true;
```



# 3 → Coisas que não deveriam estar lá

3.1 · *Checked exceptions*

3.2 · *Keywords* não utilizadas

3.3 · Variáveis implicitamente globais

## 3.1 → *Checked exceptions*

```
try {  
    FileInputStream f = new FileInputStream("naked_capybaras.mp4");  
}  
catch (FileNotFoundException e) {}  
  
// public FileInputStream(String name) |throws FileNotFoundException
```



## 3.2 → *Keywords não utilizadas*

```
Integer goto = new Integer(1);  
String const = "Why in the world!";  
  
const int x = 1; // Syntax error  
goto label; // Não, não há goto. |
```





## 3.3 → Variáveis implicitamente globais

```
var capy = "JoeJoe";  
if (true) {  
    capyGirlfriend = "Sweet";  
}  
console.log(capy + " <3 " + capyGirlfriend);
```



# 4 → Incoerência na nomenclatura

4.1 · *Case-insensitiveness*

4.2 · Verbosidade excessiva

## 4.1 → *Case-insensitiveness*

```
$x = 1;  
$X = 2;  
$x == $X; // false  
  
function y() { echo "Mah, oeh!"; }  
  
y(); // Mah, oeh!  
Y(); // Mah, oeh!
```

## 4.2 → Verbosidade Excessiva

```
SimpleBeanFactoryAwareAspectInstanceFactory  
    simpleBeanFactoryAwareAspectInstanceFactory =  
new SimpleBeanFactoryAwareAspectInstanceFactory();
```

### “Hello, World”

- C

```
#include <stdio.h>

int main(int argc, char ** argv)
{
    printf("Hello, World!\n");
}
```
- Java

```
public class Hello
{
    public static void main(String argv[])
    {
        System.out.println("Hello, World!");
    }
}
```
- now in Python

```
print "Hello, World!"
```

## 4.2 → Verbosidade Excessiva



# 5 → Erros comuns (ou nem tanto)

5.1 · Falta de suporte a funções de primeira classe

5.2 · Problemas de precisão com valores decimais

5.3 · Mensagens de erros de compilação confusas

5.4 · Ponteiros nulos



## 5.1 → Falta de suporte a funções de primeira classe

```
-- Haskell  
sum = add 10 20  
  where add =  $\lambda x\ y \rightarrow x + y$ 
```

```
// JavaScript  
sayUsers.onclick = event => {  
  userList  
    .filter(user => user.isActive)  
    .forEach(user => alert(user))  
}
```



## 5.2 → Problemas de precisão com valores decimais

→ 0.1 + 0.2

← 0.300000000000000000000004

## 5.3 → Mensagens de erro de compilação confusas



## 5.4 → Ponteiros nulos

```
// Java
Person person;
person.sayMyName();
// java.lang.NullPointerException;
```

```
-- Haskell
person :: Maybe Person
person = Just Person { name = "John Doe" }

sayMyName (Just p) = putStrLn $ name p
sayMyName Nothing = putStrLn "¬\_(\ツ)\_/_"
```

# 6 → Bônus

6.1 · Chaves em Python

6.2 · NaN

## 6.1 → Chaves em Python

```
from __future__ import braces|
```

6.2 → NaN

```
typeof NaN
```

```
NaN === NaN
```



Obrigado! Perguntas? Sugestões? Doações de capivaras?

