Donnie Robot User Documentation

Release 0.1

Alexandre Amory

jun 18, 2019

Sumário

CAPÍTULO 1

Sobre o Donnie

./source/getting_started/images/donnie-robot.png

./source/getting_started/images/donnie-sim.png

Aviso: Este documento é para ser usado por usuários. Se você quer compilar, configurar, adicionar funcionalidades para o seu Donnie, favor referenciar o dofumento Robô Assistivo Donnie: Manual do Desenvolvedor

Donnie is an assistive technology project, whose objective is to use robotics to facilitate programming teaching to visually impaired students. It is divided in two main parts:

- The construction and fine-tuning of the titular mobile robot, Donnie;
- The project's software stack, including an intuitive parser/interpreter and a robot simulation environment;

The project is in its second version, developed in the Laboratório de Sistemas Autônomos (LSA) of the Pontific Catholic University of Rio Grande do Sul (PUCRS), Brazil. Antes de começar os tutoriais, estamos assumindo que o ambiente de programação já está configurado. Se este não é o caso, por favor consulte o Manual do Desenvolvedor Donnie .

1.1 Sobre o Donnie

A robótica tem sido utilizada para ensinar o básico de programação para jovens estudantes. Entretanto, a maioria dos ambientes de programação para crianças é altamente visual, baseados em blocos de arrastar e soltar. Estudantes cegos ou com alguma deficiência visual não conseguem utilizar esses recursos.

O projeto Donnie propõe um ambiente de programação robótica inclusivo, que todos os estudantes (com ou sem deficiências visuais) podem utilizar.

O Donnie tem duas opções de usabilidade: a simulação e o robô físico. É recomendado começar com a simulação, já que não requer a construção do robô. Além disso, o robô físico é funcional mas ainda está em fase de testes.

1.1.1 Características

- Ambiente de programação de robôs para jovens estudantes com ou sem deficiência visual;
- Linguagem de programação assistiva chamada GoDonnie. A GoDonnie é uma *Text-to-speech* e é compatível com leitores de tela;
- Integração com um robô baseado em Arduino e com o middleware robótico chamado Player;
- Extensão do simulador Stage para gerar avisos sonoros quando o robô está se movendo;
- Software desenvolvido para o robô simulado é compatível com o robô Donnie real.

A simulação é recomendada se você quer aprender sobre o Donnie mas não tem os recursos necessários para construir seu próprio robô Donnie.

1.2 Ambiente de Programação Donnie

1.2.1 Introdução

A GoDonnie é uma linguagem de programação, que comanda um robô chamado Donnie por um cenário. Este robô funciona em um ambiente próprio. Para tal, estando em um terminal Linux, é necessário digitar `donnie_player` e pressionar a tecla `ENTER`.

Para comando o robô, o usuário pode utilizar o modo linha de comando ou arquivo. Para utilizar, o modo de linha de comando, deve ser digitado `godonnie -t`. Para entrar no modo arquivo, deve ser digitado `godonnie -f`. Essas expressões podem ser digitadas em minúsculo ou maiúsculo.

No modo de linha de comando, após ter sido digitado `godonnie -t`, o usuário digita os comandos desejados. Pode ser digitado um comando por linha ou vários comandos em uma mesma linha. Após cada linha, o usuário deve pressionar a tecla `ENTER`. Para que o robô execute os comandos, é preciso pressionar a tecla `ESC`. Desta forma, o usuário pode digitar uma comando e pressionar `ENTER` e `ESC`, ou escrever um comando por linha, pressionando a tecla `ENTER` para mudar de linha, e somente pressionar a tecla `ESC` quando desejar executar os comandos.

O usuário também pode criar um arquivo de texto, que contenha vários comandos. Desta forma, pode executar esse arquivo ao invés de ter que digitar repetidamente blocos de comando, por exemplo. Para executar um arquivo, o usuário deve digitar `godonnie -f` acompanhado do nome do arquivo, e pressionar a tecla ENTER. Esse arquivo já deve ter sido criado com a extensão `.txt` ou `.GD`. Um arquivo .GD é um arquivo de texto salvo com essa extensão. Este arquivo deve ter sido salvo na mesma pasta onde está o programa da GoDonnie.

A GoDonnie não diferencia letras minúsculas de maiúsculas. O robô sempre parte da posição 0,0 virado para o norte. Esta posição fica no canto inferior à esquerda.

1.2.2 Linguagem de Programação GoDonnie

Manual da Linguagem Donnie

Seção 1: sair do terminal de programação

Comando `SAIR`

Argumentos Nenhum

Explicação Fecha o ambiente de programação. Só pode ser usado no terminal.

Exemplo

```
SAIR
```

Seção 2: declaração de variáveis

Variável é um objeto que guarda um valor. Essa variável só poderá receber valores inteiros.

Comando `CRIAR x`

Argumentos `x` é a variável que será criada. Essa variável recebe valores inteiros.

Explicação Existem 5 formas de criar uma variável:

1. criar uma variável.
2. criar uma variável e atribui um valor inicial.
3. criar uma variável que recebe uma expressão.
4. criar uma variável dentro do comando de repetição `PARA`.
5. criar uma variável que receberá o valor de outro comando, como: `COR`, `DISTÂNCIA` e `POS`.

As variáveis guardam somente os últimos valores recebidos. As variáveis guardam somente valores inteiros. Desta forma, se houver um resultado com vírgula, esse será descartado e somente a parte inteira será armazenada na variável.

Existem regras para o nome das variáveis:

- Não há diferença entre letras maiúsculas e minúsculas. Desta forma, `CRIAR A` (maiúsculo) será o mesmo que `CRIAR a` (minúsculo).
- Não podem ter caracteres especiais. Exemplo: *, @, #, +
- Não podem iniciar com número. Exemplo: CRIAR 52abc está errado.

Exemplo

1. Para criar uma variável sem valor inicial, pode-se fazer:

```
CRIAR A
```

Cria uma variável com o nome `A`.

```
A = 2
```

Tendo sido criada a variável, pode atribuir um valor diretamente. A variável com o nome `A` vai armazenar o valor 2.

2. Para criar uma variável com valor inicial, pode-se fazer como a seguir:

```
CRIAR B =5
```

   Cria uma variável chamada `B`, que armazena o valor 5

3. Para criar uma variável que recebe uma expressão, pode-se fazer como a seguir:

```
CRIAR C = A + B
```

   Cria uma variável chamada `C`, que recebe o valor da variável `A` somado ao valor da variável chamada `B`. O resultado da variável `C` é 7.

```
C = 1
```

   Altera o valor da variável `C` e armazena o valor 1, perdendo o valor anterior.

4. Para criar uma variável dentro de um comando `PARA` (esse comando será visto na seção 10 do manual), pode ser feito da seguinte forma:

```
PARA CRIAR d = 0;  d < 5; d = d + 1 FAÇA
PF 1
FIM PARA
```

   O robô se deslocará 5 passos para frente.

5. Para criar uma variável que recebe o valor de outro comando, pode-se fazer como a seguir:

```
CRIAR d = DISTÂNCIA F
CRIAR c = COR VERDE
CRIAR px = POS X
```

   - A variável `d` armazenará o valor da distância frontal do robô em relação ao objeto.
   - A variável `c` armazenará a quantidade de cores verdes.
   - A variável `px` armazenará a posição atual do robô no eixo x.
   - (Os comandos Distância F, Cor e Pos x serão vistos na seção x)

```
G = 5
```

   Retornará erro porque a variável `G` ainda não foi criada.

Seção 3: comandos de áudio

Comandos para manipulação e retorno de áudio.

a)

Comando `FALAR x`

Argumentos `x` é uma variável, que deve ter sido criada anteriormente.

Explicação Fala o conteúdo da variável. Este som é emitido pelo robô ou pelo ambiente virtual, dependendo de quem estará ativo.

Exemplo

```
CRIAR x = 5
FALAR x
```

Será falado: 5

b)

Comando `FALAR "x"`

Argumentos `x` é uma palavra ou frase, que deve vir entre aspas duplas.

Explicação Fala a palavra ou frase contida entre as aspas. Este som é emitido pelo robô ou pelo ambiente virtual, dependendo de quem estará ativo.

Exemplo

```
FALAR "oi"
```

Será falado: `"oi"`

c)

Comando `SOM ligado` `SOM desligado`

Argumentos O estado do áudio, é ligado ou desligado.

Explicação Comando que liga ou desliga o áudio do recurso que estiver ativo, que poderá ser o robô ou o ambiente virtual.

Exemplo

```
SOM LIGADO
SOM DESLIGADO
```

Seção 4: operadores

São operadores que fornecem suporte a expressões matemáticas e lógicas.

Comando Operadores

Argumentos

Matemáticos:

`+` soma
`-` subtração
`*` multiplicação
`/` divisão

Comparadores:

`<>` diferente
`==` igual
`<` menor

```
> maior
<= menor ou igual
>= maior ou igual
```

atribuição:
`= atribuição`

Explicação Operadores servem para comparar valores ou expressões.

Exemplo *Para realizar uma soma:*

```
Criar a = 2
```

criando a variável `a` e atribuindo o valor de 2.

```
Criar b = 1
```

Criando a variável `b` e atribuindo o valor de 1.

```
Criar soma
```

Criando a variável `soma`

```
*soma* = a + b
```

atribuindo a `soma` o valor da *soma* da variável `a` e `b`.

```
Falar soma
```

Será falado: 3

Para realizar uma divisão:

```
Criar c = 2
```

criando a variável `c` e atribuindo o valor de 2.

```
Criar d = 2
```

Criando a variável `d` e atribuindo o valor de 2.

```
Criar divisão
```

Criando a variável divisão

```
divisão = c / d
```

Atribuindo o valor da divisão dos conteúdos das variáveis `c` e `d`.

```
Falar divisão
```

Será falado: 1

Seção 5: comandos de movimentação

São comandos que movimentam o robô no ambiente.

a)

Comando `PF n`

Argumentos `n` é o número de passos. Este comando aceita somente números inteiros e positivos, ou variáveis que armazenam números inteiros, ou expressões matemáticas que resultem em números inteiros.

Explicação Anda `n` passos para frente.

Exemplo

```
PF 5
```

O robô andará 5 passos para frente. Supondo que o robô está na posição 0, 0 e virado para o norte, o comando PF 5 colocará o robô na posição 5, 0, mantendo a direção para o norte.

```
CRIAR A = 10
PF A
```

Fará com que o robô ande 10 passos para frente.

```
CRIAR A = 10
CRIAR B = 20
PF A + B
```

Fará com que o robô ande 30 passos para frente.

Se o robô colidir em algo antes de completar a quantidade de passos solicitados. Será informado ao usuário: "`Andei somente X passos para frente. Encontrei obstáculo`".

Se for digitado o comando com um número negativo como abaixo:

```
PF -5
```

Será informado ao usuário que o robô andou 0 passos.

b)

Comando `PT n`

Argumentos `n` é o número de passos. Este comando aceita somente números inteiros e positivos, ou variáveis que armazenam números inteiros, ou expressões matemáticas que resultem em números inteiros.

Explicação Anda `n` passos para trás. É como se andasse de ré.

Exemplo

```
PT 5
```

O robô andará 5 passos para trás. Supondo que o robô está na posição 5, 0 e virado para o norte, o comando PT 5 colocará o robô na posição 0, 0, mantendo a direção para o norte.

```
CRIAR A = 10
PT A
```

Fará com que o robô ande 10 passos para trás.

```
CRIAR A = 10
CRIAR B = 20
PT A + B
```

Fará com que o robô ande 30 passos para trás.

Se o robô colidir em algo antes de completar a quantidade de passos solicitados. Será informado ao usuário: "`Andei somente X passos para trás. Encontrei obstáculo`".

Caso seja digitado o comando com número negativo como abaixo:

```
PT -6
```

Será informado, "`andei 0 passos`".

Seção 6: comandos de Rotação

Rotação sem movimento do robô

a)

Comando `GD n`

Argumentos `n` é número de graus. Este comando aceita somente números inteiros positivos e negativos, ou variáveis que armazenam números inteiros, ou expressões matemáticas que resultem em números inteiros.

Explicação Gira `n` graus para direita. Não há deslocamento do robô.

Exemplo

```
GD 90
```

O robô irá girar 90 graus para direita. Supondo que o robô está virado para o norte, o comando GD 90 irá girar o robô 90 graus para a direita, mantendo-o na direção leste.

```
CRIAR A = 45
GD A
```

Fará com que o robô gire 45 graus para a direita.

```
CRIAR A = 80
CRIAR B = 10
GD A + B
```

Fará com que o robô gire 90 graus para a direita.

```
GD -90
```

O robô gira para o lado esquerdo 90 graus.

b)

Comando `GE n`

Argumentos `n` é número de graus. Este comando aceita somente números inteiros positivos e negativos, ou variáveis que armazenam números inteiros, ou expressões matemáticas que resultem em números inteiros.

Explicação Gira `n` graus para esquerda. Não há deslocamento do robô.

Exemplo

```
GE 90
```

O robô irá girar 90 graus para esquerda. Supondo que o robô está virado para o leste, o comando GE 90 irá girar o robô 90 graus para a esquerda, mantendo-o na direção norte.

```
CRIAR A = 45
GE A
```

Fará com que o robô gire 45 graus para a esquerda.

```
CRIAR A = 80
CRIAR B = 10
GE A + B
```

Fará com que o robô gire 90 graus para a esquerda.

```
GE -90
```

O robô gira para o lado direito 90 graus.

Seção 7: comandos de visualização do ambiente

São comandos para obter informações sobre o ambiente em que o robô está. Não é possível armazenar o retorno desses comandos em variáveis.

a)

Comando `ESPIAR`

Argumentos nenhum

Explicação Retorna a identificação do objeto, um ângulo aproximado e a distância aproximada de colisão entre o robô e o objeto identificado. O rastreamento para identificação dos objetos ocorre a 90 graus a esquerda e a direita da frente do robô.

Exemplo Supondo que o robô está na posição 2,3, virado para o norte, e que há um obstáculo verde na posição 0,5 e outro obstáculo vermelho na posição 6,3.

```
ESPIAR
```

Será falado: `a 40 graus a esquerda: 1 objeto de cor verde a 2 passos. 90 graus a direita: 1 objeto da cor vermelha a 4 passos.`

No caso de dois objetos no mesmo ângulo será informado: `a 30% a esquerda: dois objetos de cores verde, vermelho a 17 passos.`

b)

Comando `ESTADO`

Argumentos nenhum

Explicação Retorna a posição no eixo X, Y e o ângulo do robô e informa o último comando digitado de rotação ou de deslocamento, anterior ao comando ESTADO.

Exemplo

```
PF 3 ESTADO
```

Supondo que o robô estava em 0,0. O robô andará 3 passos para frente e informará "`Comando 1 foi PF 3, andou 3, não bateu, posição [3,0,0]`". O 3 corresponde ao eixo x, o primeiro 0 ao eixo y e o último 0 ao ângulo do robô.

Caso o robô tenha colidido em algo completando apenas 2 passos com sucesso, o ESTADO retornará: "`Comando 1 foi PF 3, andou 2, bateu, posição [2,0,0]`". O 2 corresponde ao eixo x, o primeiro 0 ao eixo y e o último 0 ao ângulo do robô.

Não havendo comandos digitados anteriormente, retornará: "`Nenhum comando executado, Posição [0, 0, 0]`".

Seção 8: comandos de posição e percepção do ambiente

São comandos para obter informações sobre o ambiente em que o robô está. É possível armazenar o retorno desses comandos dentro de variáveis.

a)

Comando `DISTÂNCIA d`

Argumentos `d` é a direção do sensor do robô (`f` - frontal; `fd` - frontal direita; `fe` -frontal esquerda; `td` - traseiro direito; `t` - traseiro; `te` - traseiro esquerda)

Explicação Retorna a quantidade de passos do sensor do robô até um obstáculo, de acordo com a direção escolhida.

Há três formas de se utilizar o comando DISTÂNCIA:

1. Se o usuário desejar escutar o retorno, deve utilizar o comando `FALAR` junto com o comando `DISTÂNCIA`.
2. Se deseja somente armazenar em uma variável.
3. Se deseja usar diretamente dentro de outro comando, por exemplo: `SE`, `PARA`, `REPITA` ou `ENQUANTO`.

- Distância F retorna o número de passos do robô até um objeto que foi detectado pelo sensor da parte da frente do robô.
- Distância FD retorna o número de passos do robô até um objeto que foi detectado pelo sensor da parte da frente lateral direita do robô.
- Distância TD retorna o número de passos do robô até um objeto que foi detectado pelo sensor da parte da trás lateral direita do robô.
- Distância T retorna o número de passos do robô até um objeto que foi detectado pelo sensor da parte da traseira do robô. E, assim, sucessivamente.

Não havendo obstáculos, retorna a quantidade de passos que o sensor consegue identificar, que geralmente é até 60 passos.

Exemplo

```
DISTÂNCIA F
DISTÂNCIA FD
DISTÂNCIA FE
DISTÂNCIA T
DISTÂNCIA TE
DISTÂNCIA TD
```

1. Supondo que o robô está na posição 0,0, virado para o norte e há obstáculos nas seguintes posições, o resultado será:

Obstáculo em 0, 3:

```
FALAR DISTÂNCIA F
```

Resposta: 3 passos

2. Você pode criar uma variável previamente, para depois utilizar para armazenar o retorno do comando DISTÂNCIA

```
CRIAR  d =  DISTÂNCIA T
```

Armazena na variável `d` a distância traseira do robô até o obstáculo que está diretamente atrás dele. Supondo que o Robô está na posição 0,3 virado para o norte e existe um obstáculo em 0,0. O valor armazenado em d será 3.

3.

```
SE DISTÂNCIA F>3 ENTÃO
PF 1
SENÃO
FALAR "não é possível andar para frente"
FIM SE
```

No exemplo acima, se a distância frontal do robô for maior que 3, o robô andará 1 passo para frente. Se for igual ou menor a 3, irá falar "`não é possível andar para frente`".

```
ENQUANTO DISTÂNCIA F>3
FAÇA
PF 1
FIM ENQUANTO
```

No exemplo acima, enquanto a distância frontal do robô em relação ao objeto for maior que 3, andará 1 passo para frente.

b)

Comando `POS k`

Argumentos `k` é um eixo do plano cartesiano (X ou Y) ou ângulo (A).

Explicação Retorna a posição atual do robô no eixo X ou no eixo Y ou o ângulo atual do robô.

Há três formas de se utilizar o comando POS k:

1. Se o usuário deseja escutar o retorno, deve utilizar o comando FALAR junto com o comando POS x, POS y ou POS a.
2. Se deseja somente armazenar em uma variável.
3. Se deseja usar diretamente dentro de outro comando, por exemplo: SE, PARA, REPITA ou ENQUANTO.

Exemplo

1. Se o usuário desejar escutar o retorno, pode-se fazer como a seguir: Supondo que o robô está na posição 0,0 virado para o norte:

```
FALAR POS x
```

será falado 0

```
FALAR POS y
```

será falado 0

```
FALAR POS a
```

Será falado 0

2. Se deseja somente armazenar o valor da posição, pode-se fazer como a seguir:

```
CRIAR z = POS x
```

A variável z possui a posição do robô no eixo x.

```
CRIAR b = POS y
```

A variável b contém a posição do robô no eixo y.

```
CRIAR i = POS a
```

A variável i contém o ângulo do robô.

3. Se deseja utilizar diretamente dentro de outros comandos, pode-se fazer como a seguir:

```
SE POS b > 0 ENTÃO
PF 5
SENÃO
PT 5
FIM SE
```

c)

Comando COR c

Argumentos c é a cor desejada (azul; vermelho; verde)

Explicação Verifica quantos objetos de determinada cor o robô consegue identificar num ângulo de 180 graus a sua frente.

Há três formas de se utilizar o comando COR:

1. Se o usuário desejar escutar o retorno, deve utilizar o comando FALAR a frente do comando COR.
2. Se deseja somente armazenar em uma variável, declarando-a anteriormente.

3. Se deseja usar diretamente dentro de outro comando, por exemplo: SE, PARA, REPITA ou ENQUANTO.

Exemplo

1. Se o usuário desejar escutar o retorno, pode-se fazer como a seguir: Supondo que há 1 objeto verde e 2 azuis

```
FALAR COR azul
```

será falado 2

```
FALAR COR verde
```

será falado 1

2. Se deseja somente armazenar o valor da cor, pode-se fazer como a seguir:

```
CRIAR A = COR AZUL
```

A variável A possui a quantidade de objetos azuis

```
CRIAR V = COR VERDE
```

A variável V contém a quantidade de objetos verdes.

3. Se deseja utilizar diretamente dentro de outros comandos, pode-se fazer como a seguir:

```
SE COR AZUL > 0 ENTÃO
FALAR "Número de objetos azuis"
FALAR COR AZUL
SENÃO
FALAR "Não encontrei objetos azuis"
FIM SE

SE COR VERDE > 0 ENTÃO
FALAR "Número de objetos verdes"
FALAR COR VERDE
SENÃO
FALAR "Não encontrei objetos verdes"
FIM SE
```

Seção 9: comandos de condição

São comandos condicionais que permitem ao programa fazer a escolha do que executar, de acordo com uma condição estipulada.

a)

Comando

SE *expressão operador lógico expressão*
ENTÃO *comandos*
SENÃO *comandos*
FIM SE

Argumentos expressão = variável ou expressão.

Explicação Testa se uma condição é verdadeira e, em caso afirmativo, executa os primeiros comandos. Caso contrário, executa os comandos da expressão SENÃO.

Exemplo Supondo que, se a variável `a` for menor do que 4 o robô tenha que andar para frente 5 passos e caso contrário tenha que girar 45 graus para esquerda:

```
CRIAR a = 0
SE a < 4
ENTÃO PF 5
SENÃO GE 45
FIM SE
```

b)

Comando

`SE` *expressão operador lógico expressão*
`ENTÃO` *comandos*
`FIM SE`

Argumentos expressão = variável ou expressão.

Explicação Testa se uma condição é verdadeira e, em caso afirmativo, executa os primeiros comandos.

Exemplo

```
CRIAR a = 0
SE a < 4
ENTÃO PF 5
FIM SE
```

Se a variável `a` tiver um valor menor do que 4 então o robô andará 5 passos para frente.

Seção 10: comandos de repetição

São comandos de repetição que permitem uma ou mais instruções serem executadas um determinado número de vezes.

a)

Comando

`PARA` *inicialização; expressão operador lógico expressão; incremento ou decremento*
`FAÇA` *comandos*
`FIM PARA`

Argumentos

Inicialização: variável = algum valor inteiro

variável ou Expressão operador lógico variável ou expressão: variável ou expressão - operador lógico - variável ou expressão

Incremento: variável + constante ou variável + variável

Decremento: variável - constante ou variável - variável

Explicação Repete a sequência de comandos um determinado número de vezes.

Exemplo O exemplo faz com que o robô precise andar em direção a um obstáculo que está a sua frente e a cada passo fale "oi".

```
CRIAR obstaculo = DISTÂNCIA F
PARA CRIAR x = 1; x <= obstaculo; x = x + 1
FAÇA
PF 1
FALAR "oi"
FIM PARA
```

A variável `x` começará com o valor 1 e o robô andará um passo para frente e falará `"oi"`, enquanto seu valor for menor ou igual a linha do obstáculo que está à sua frente.

b)

Comando

`REPITA n VEZES` comandos
`FIM REPITA`

Argumentos `n` é o número de vezes que os comandos serão repetidos.

Explicação Repete os comandos `n` vezes.

Exemplo

```
REPITA 4 VEZES
GD 90
PF 2
FIM REPITA
```

Supondo que o robô comece na posição 0,0. Os comandos PF 3 GD 90 serão repetidos 4 vezes. Ao final, o robô terá feito um trajeto similar a um quadrado e finalizará na posição 0,0 virado para o norte.

c)

Comando

`ENQUANTO` *expressão operador lógico expressão*
`FAÇA` *comandos*
`FIM ENQUANTO`

Argumentos variável ou Expressão operador lógico variável ou expressão: variável ou expressão - operador lógico - variável ou expressão

Explicação Repete os comandos enquanto a Expressão-operador lógico-expressão for verdadeira.

Exemplo O exemplo faz com que o robô precise andar em direção a um obstáculo que está a sua frente e a cada passo fale "estou chegando".

```
ENQUANTO DISTÂNCIA F >3
FAÇA
PF 1
FALAR "estou chegando"
FIM ENQUANTO
```

Enquanto a distância da frente do robô em relação ao objeto for maior que 3, o robô andará um passo para frente e falará "estou chegando"

Seção 11: declaração de procedimentos

Procedimento e um programa menor (subprograma) que permite decompor e resolver um problema mais complexo em um mais simples. Pode ser chamado em outras partes do programa.

Comando

APRENDER *nome*: *variável1*, *variável2*, *variável3*, ...

FAÇA *comandos*

FIM APRENDER

Argumentos *nome* é o nome do subprograma e *variavel1*, *variavel2*, *variavel3* são os argumentos da mesma

Explicação

Serve para criar um subprograma.

Este comando somente funciona via arquivo.

Exemplo O robô precisa caminhar simulando um retângulo. Esse retângulo pode ter tamanhos diferentes, conforme a atividade. Por isso, pode ser utilizado o comando APRENDER para criar um procedimento único chamado RETÂNGULO que receberia duas variáveis, uma para o tamanho da altura e a outra para o tamanho da base. Assim, esse procedimento poderia ser utilizado para fazer retângulos de tamanhos diferentes.

```
APRENDER RETÂNGULO: base, altura
FAÇA
PF base GD 90
PF altura GD 90
PF base GD 90
PF altura GD 90
FIM APRENDER
```

Ou

```
APRENDER RETÂNGULO: base, altura
FAÇA
REPITA 2 VEZES
PF base GD 90
PF altura GD 90
FIM REPITA
FIM APRENDER
```

chamada do subprograma

```
RETÂNGULO [5,3]
RETÂNGULO [8,4]
RETÂNGULO [9,5]
```

Seção 12: comandos variados

a)

Comando `ESPERAR t`

Argumentos `t` é o tempo em segundos

Explicação Espera `t` segundos para executar o próximo comando.

Exemplo Se o robô deve andar para frente 2 passos, esperar 3 segundos e andar mais 4 passos:

```
PF 2
ESPERAR 3
PF 4
```

b)

Comando `--`

Argumentos nenhum

Explicação Após esse símbolo `--` tudo que for escrito na linha que possui `--` não será executado. São lembretes sobre o código.

Exemplo

```
-- Isto é um comentário.
```

manual da linguagem e eexemplos de uso. colocar os exercicios como se fossem subsecoes.

1.2.3 GoDonnie Interpreter

modos de operacao, exemplos de uso

1.3 Donnie Robot Simulator

nao falar sobre como montar um novo cenario. para isso, aponte para o manual do desenvolvedor. mencionar brevemente que o simulador eh baseado no Stage.

1.3.1 Donnie Scenarios

falar sobre os cenarios prontos.

um dos cenarios deve incluir muitiplos robos.

1.4 Donnie Robot

nao mostrar como montar e configurar um robo. para isso, aponte para o manual do desenvolvedor. nesta secao assume-se que o robo esta pronto para uso.

1.4.1 Power Up

procedimentos de inicialização

1.4.2 Running GoDonnie

como executar o GoDonnie com robô fisico

1.4.3 Shutting Down

como desligar o robo

1.5 Cinto Vibratório Donnie

A melhora na qualidade de vida das pessoas veio junto com a tecnologia, que auxilia em atividades do dia-a-dia e facilita a vida do ser humano. Com a criação de novos recursos e com o intuito de melhorar a vida de pessoas que possuem algum tipo de deficiência, tem sido crescente o número de tecnologias assistivas em desenvolvimento. Tendo assim o objetivo de melhorar a qualidade de vida de pessoas com deficiências visuais e auxiliar na sua mobilidade através de uma melhor percepção do ambiente foi criado um cinto vibratório. Capaz de atuar com intensidades diferentes, faz com que o usuário perceba, através de vibrações, um objeto se aproximar ou se afastar. Este cinto inicialmente contava com um sensor Kinect que fazia a "visualização" do ambiente ao seu redor e identificava obstáculos, e com 35 motores que cobriam todo o abdômem e vibravam conforme as informações recebidas. Agora adaptado e implantado no Projeto Donnie, que é um projeto de uso de robótica para o ensino de programação baseada em interface multimodal para pessoas que são cegas, passou a ter apenas 14 motores, mas que englobam desde a parte da frente do abdômen até as costas, passando por ambos os músculos oblíquo externos. O cinto conta com quatro diferentes tipos de vibrações e, através da criação de um cliente específico para o ele, se conecta com um robô chamado Donnie e recebe as informações dos seus sensores. Da mesma forma ele se conecta a plataforma GoDonnie, que é um ambiente de simulação, e recebe as informações dos sensores do robô simulado. Ao receber essas informações, tanto do robô Donnie quanto do robô simulado, o cinto permite que a pessoa perceba através de vibrações a distância dos objetos ao redor do robô.

1.5.1 Ligando

procedimentos de inicialização

1.5.2 Executando o Cinto

como executar o cinto

1.5.3 Desligando

como desligar o cinto

1.6 Donnie Contributors

The list of contributors to this document.

- @Alexandre Amory
- @Darlan Alves Jurak

CAPÍTULO 2

Papers

If you are using Donnie and/or its software on your research projects, please cite our papers:

```
@inproceedings{oliveira2017teaching,
  title={Teaching Robot Programming Activities for Visually Impaired Students: A␣
↪Systematic Review},
  author={Oliveira, Juliana Damasio and de Borba Campos, M{\'a}rcia and de Morais␣
↪Amory, Alexandre and Manssour, Isabel Harb},
  booktitle={International Conference on Universal Access in Human-Computer␣
↪Interaction},
  pages={155--167},
  year={2017},
  organization={Springer}
}
```

```
@inproceedings{guilherme2017donnie,
  title={Donnie Robot: Towards an Accessible And Educational Robot for Visually␣
↪Impaired People},
  author={Guilherme H. M. Marques, Daniel C. Einloft, Augusto C. P. Bergamin, Joice A.
↪ Marek, Renan G. Maidana Marcia B. Campos, Isabel H. Manssour, Alexandre M. Amory},
  booktitle={Latin American Robotics Symposium (LARS)},
  year={2017}
}
```

CAPÍTULO 3

Disclaimer

Donnie and its software are protected under the MIT License:

CAPÍTULO 4

Feedback

Don't hesitate to ask about additional info or the next guides, and also if you find some mistakes, please let us know. Issues and push requests can be done on github.