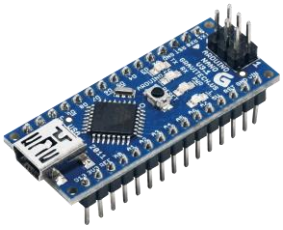
Manual de Instruções

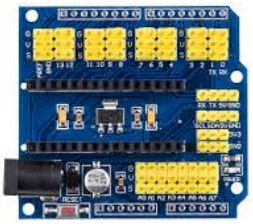
Curso Técnico de Eletrónica, Automação e Comando
Escola Profissional Gustave Eiffel
Luís Santos Nº115693 Turma: 474

Materiais necessários

- 1 Arduíno Nano;



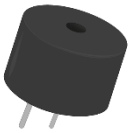
- 1 Shield Arduíno Nano;



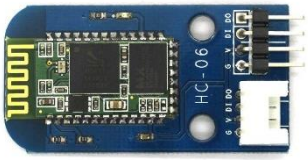
- 4 Micro Servos SG90 9g;

- 1 Sensor Ultrassónico HC-SR04;

- 1 Buzzer;



- 1 Módulo Bluetooth HC-06;



- 1 Suporte para pilhas AA

Ficheiros STL-3D necessários:

- 1 Otto Body; 1 Otto Head; 1 Otto Right Leg e 1 Otto Left Leg; 1 Otto Right Foot e 1 Otto Left Foot;

Microcontroladores - Arduíno

Um microcontrolador é um circuito integrado, onde é possível reunir um certo circuito que se queira implementar para que exerça uma determinada instrução dada pelo programador. São pequenas placas que possuem todos os recursos necessários para que se consiga inserir uma dada programação, sendo ainda composta por um circuito de alimentação próprio, para a alimentação dos componentes que se queira utilizar no circuito

Usa-se microcontroladores para que se possa controlar circuitos a partir de informações que são transmitidas entre um computador e o respetivo microcontrolador.

De uma forma esquemática, um microcontrolador é composto por uma entrada de alimentação, proveniente do cabo que é ligado com computador, em que depois faz funcionar cinco elementos importantes para um bom desempenho do microcontrolador.

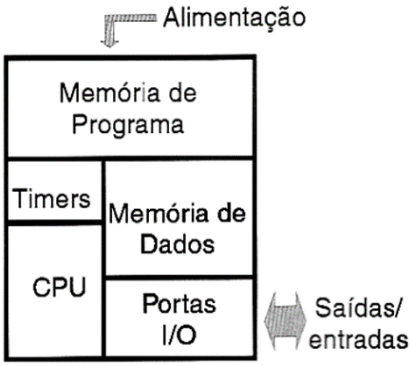


Fig. 1 – Constituição de um microcontrolador

Esses elementos são a **Memória do Programa** que se designa pelo espaço ocupado pelo programa criado em que o tamanho dessa memória varia segundo a finalidade e o tipo de microcontrolador, tem ainda a **Memória de Dados** que é onde são guardadas as informações/instruções dadas através da programação. Existe ainda a ***CPU***, sendo considerada pela parte mais importante de um microcontrolador, que é responsável pelo processamento das informações, onde as siglas significam, em inglês, *Central Processing Unit*, e em português, é Unidade Central de Processamento.

Existem também ***Timers*** como elementos fundamentais, em que consistem no envio de informações através de intervalos de tempo precisos, onde esse processamento é feito com o auxílio de um "*clock*".

Por fim, existem as **portas de *Input* e de *Output*** onde através de um chip existe a conversão das saídas pela ligação com as portas digitais e analógicas da placa, podendo elas funcionar como portas de saída ou como portas de entrada dependendo do tipo de circuito que se queira implementar.

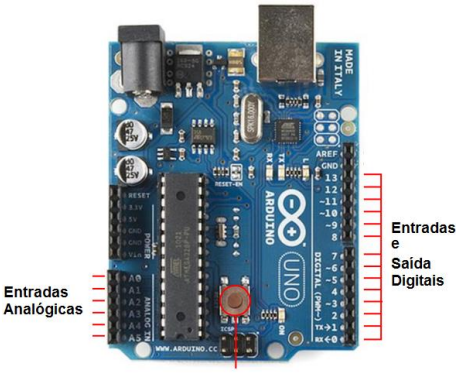


Fig. 2 – Portas Analógicas e Digitais

Neste projeto foi utilizado o Arduíno Nano que é uma placa de microcontrolador baseado no ATmega328. Este Arduíno é composto por 8 portas analógicas, e 22 portas digitais. Dispõe de um "*clock*" com uma velocidade de 16MHz, tem 4 pins de alimentação (5V e *GND*), e pins de comunicação (*RX* e *TX*). Para o desenvolvimento do Otto Robot utilizou-se ainda um shield para que houvesse mais pins de alimentação para ligar os componentes.

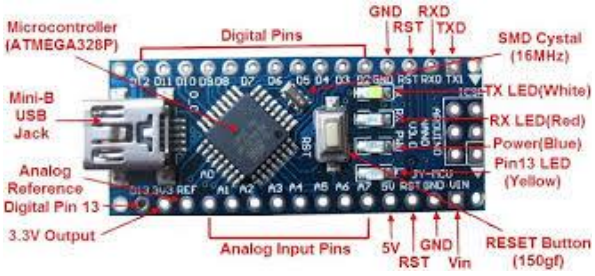


Fig. 3 – Arduíno Nano

Fig. 4 – Shield Arduíno Nano

Módulo Bluetooth HC-06

Um módulo *HC-06* é um dispositivo com o intuito de estabelecer a comunicação, neste caso, entre um aparelho que contenha *Bluetooth* e um Arduíno, no qual também pode ser usado um aparelho como um tablet ou computador.

Com este módulo *Bluetooth* é possível enviar via porta serial, com o *RX*/*TX*, até aproximadamente 10 metros de distância. É um dispositivo que trabalha com 3,3V e com 5V, onde o mais recomendado pelos fabricantes é a utilização dos 5V para um bom funcionamento do módulo, já que as placas Arduíno funcionam com os 5V.

Antes de se comunicar com o módulo, é preciso que haja o emparelhamento com o dispositivo a utilizar. Este processo vai variando dependendo do tipo de Sistema Operativo que se utiliza no dispositivo, mas em termos gerais é necessário que exista a procura de módulos *Bluetooth* ao redor do telemóvel, através do acesso às definições do telemóvel, neste caso. De seguida, é possível encontrar um endereço chamado de "itead", no qual se tem que inserir um código de ativação, sendo esse código "1234".

O módulo *Bluetooth* tem a porta serial totalmente modificada para 3Mbps, com *Bluetooth* V2.0 + *EDR* (*Enhanced Data Rate*), com transcetor de rádio completo de 2,4 GHz e banda de base. Usa um *CSR Bluecore* 04 com um sistema externo para um chip único com tecnologia *CMOS* e *AFH* (Função adaptativa de frequência).

Este dispositivo dispõe de 4 pins, sendo dois deles os de alimentação, onde o G é o pin de alimentação negativo e o V é o pin de alimentação positiva. Os outros dois pins são os de receção e transmissão de dados, em que o que transmite encontra-se indicado por D1 (*TX*) e o outro pin é o recetor representado por D0 (*RX*).

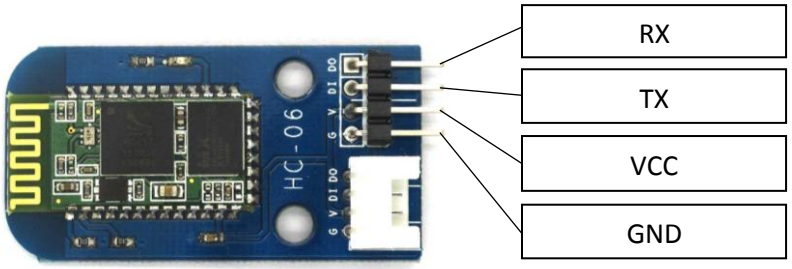


Fig. 5 – Módulo Bluetooth HC-06

Micro Servo

Os servomotores são dispositivos de malha fechada, ou seja, recebem um sinal de controlo, verificam a posição atual e atuam no sistema indo para a posição desejada.

Um servo é sempre de alta qualidade e excelente para as suas necessidades, seja em aeromodelismo ou em projetos de robótica com Arduíno. Os servomotores são usados em várias aplicações quando se deseja movimentar algo de forma precisa e controlada.

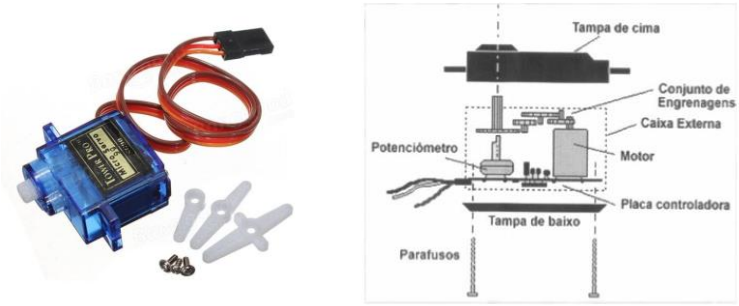


Fig. 6 – Representação de um servomotor

Um servomotor encontra-se sempre com sinais de impulso de 20 ms, em que quando há uma mudança nesse sinal faz com que seja alterada a posição do servo, para que corresponda ao sinal recebido.

Um sinal com largura de pulso de 1 ms corresponde a posição do servo todo a esquerda ou 0 grau. Um sinal com largura de pulso de 1,5 ms corresponde a posição central do servo ou de 90 graus. Um sinal com largura de pulso de 2 ms corresponde a posição do servo todo a direita ou 180 graus.

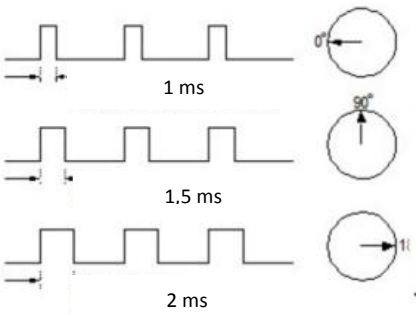


Fig. 7 – Sinais de Impulso

Estes servomotores são compostos por 3 pins para que seja feita a ligação com o Arduíno. Para tal, existem dois desses pins que são para a alimentação do servo (*GND* e *VCC*), pin castanho e vermelho, respetivamente, e um de comunicação/sinal, representado pela cor amarela.

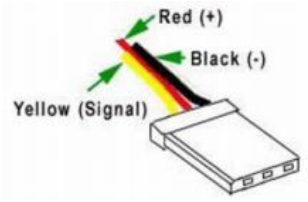


Fig. 8 – Representação dos pinos

Os tópicos seguintes são representativos dos elementos que constituem um micro servo, no qual serão explicados cada um deles:

Sistema Atuador – O sistema atuador é constituído por um motor elétrico, onde na maioria das vezes se utiliza motores de corrente contínua. **O tamanho, o binário e a velocidade do motor, material das engrenagens, liberdade de rotação do eixo e consumo** são características chaves para especificação dos servomotores.

Potenciómetro – É um potenciómetro solidário ao eixo do servo. A qualidade do potenciómetro interfere na precisão, estabilidade e vida útil do servo motor.

Circuito de Controlo – É formado por circuitos integrados e é composto por um oscilador e por um controlador PID (Controlo proporcional integrativo e derivativo) que recebe um sinal do potenciómetro (posição do eixo) e o sinal de controlo e aciona o motor no sentido necessário para posicionar o eixo na posição desejada.

Engrenagens – Reduzem a rotação do motor, transferem mais torque ao eixo principal de saída e movimentam o potenciómetro junto com o eixo.

Sensor Ultrassónico HC-SR04

O sensor ultrassónico *HC-SR04* é capaz de medir distâncias de 2cm a 4m com uma ótima precisão e a baixo preço. Este módulo possui um circuito pronto com emissor e recetor acoplados e 4 pinos, sendo eles o *VCC* e o *GND* como pins de alimentação, o *TRIG* e o *ECHO* com o intuito de detetarem o objeto a uma determinada distância predefinida.

Fig. 9 – Sensor ultrassónico

Para começar a medição é necessário alimentar o módulo e colocar o pino *TRIG* em nível alto por mais de 10us. Assim o sensor emitirá uma onda sonora que ao encontrar um obstáculo irá mandar de volta essa onda em direção ao módulo, sendo que nesse tempo de emissão, o sinal do pino *ECHO* ficará em nível alto. Logo o cálculo da distância pode ser feito de acordo com o tempo em que o pino *ECHO* permaneceu em nível alto após o pino *TRIG* ter sido colocado em nível alto. Desta forma, a fórmula matemática utilizada para determinar a distância é a seguinte:

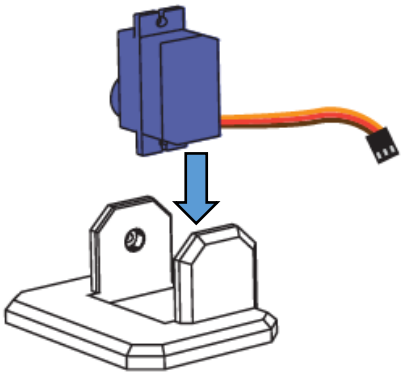
Distância = [Tempo *ECHO* em nível alto * Velocidade do Som] / 2

A velocidade do som poder ser considerada igual a 340 m/s, logo o resultado é obtido em metros se for considerado o tempo em segundos. Na fórmula, a divisão por 2 deve-se ao facto da onda ser enviada e devolvida, logo, a onda percorre 2 vezes a distância procurada, ou seja, o sinal é enviado para se saber se existe algum objeto, e volta de novo até ao módulo para verificar a distância que existe entre o objeto e o sensor.

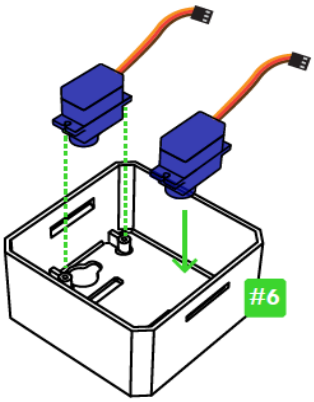
Construção do Otto Robot

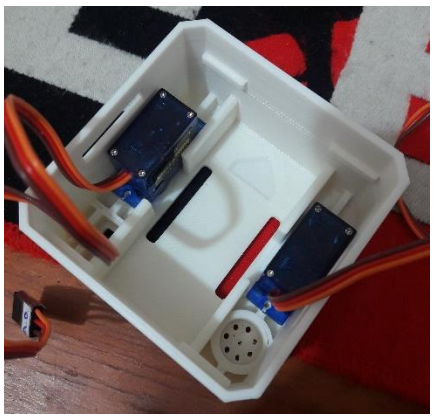
O primerio passo a ser realizado é a construção de cada um dos pés do Otto. Para tal, é necessário que o micro servo fique com a roda dentada colocada dentro do orificio existente em cada um dos pés. Dentro desse orificio coloca-se um parafuso para se fixar o servo ao pé. O processo é igual tanto para o pé direito como para o esquerdo.

NOTA: A posição inicial, quando se compra um micro servo, é sempre de 90 graus, sendo assim que deve ser encaixado primeiramente.

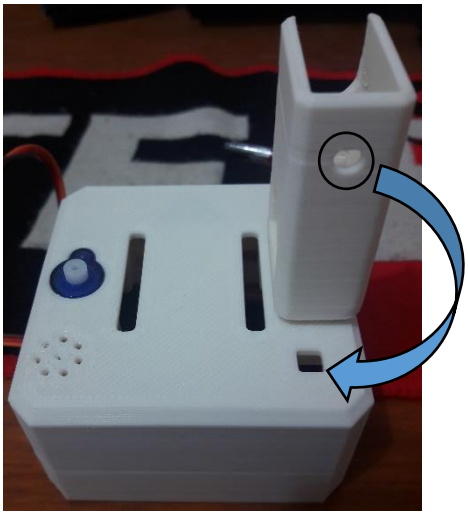


O segundo passo a ser feito, passa por colocar os micro servos no corpo do Otto. É um processo simples já que só é necessário colocar o micro servo de acordo com a forma desenhada na estrutura. Depois de colocados, só é preciso aparafusar cada uma das extremidades dos servos.



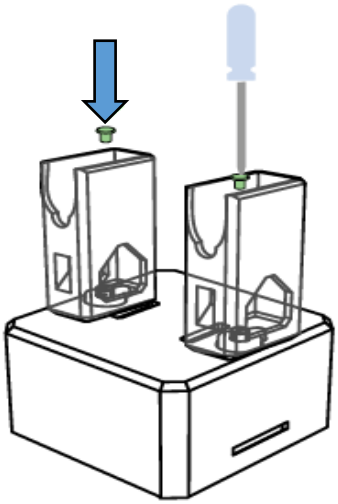


Passando agora para o terceiro passo, é necessário inserir as pernas na parte do corpo do Otto. Para isso, e tendo como principio de orientação que os buracos existentes na parte de baixo do corpo do Otto correspondem à parte da frente do robot, coloca-se as pernas com a parte que contem apenas um orificio, voltado para a zona do buraco no corpo, como mostra a seguinte imagem.

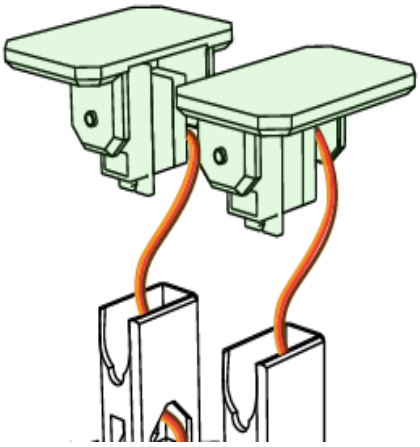


Orifício existente na perna, voltado para a zona onde se encontra o buraco no corpo do Otto.

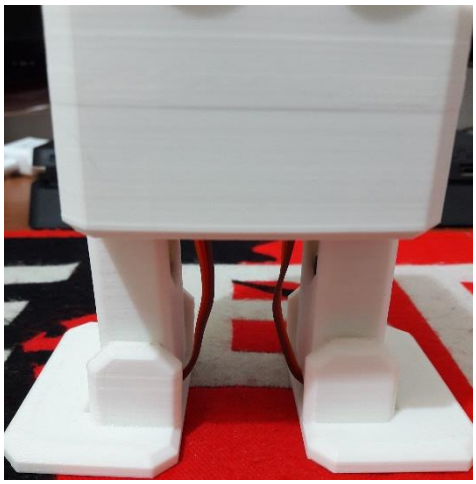
Depois de colocadas as duas pernas como foi demonstrado na imagem anterior, ficará a ver-se dentro da perna a roda dentada do servo. O que se tem de fazer é voltar a usar o parafuso respetivo para ser aplicado na roda dentada do servo e apertá-lo, de maneira a que o corpo e a respetiva perna fiquem interligados com o servo,não havendo a possibilidade de se soltarem as peças da estrutura.



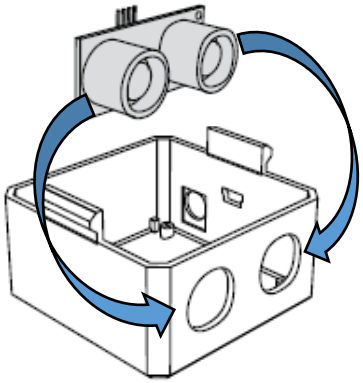
A quarta etapa deste projeto passa por unir a perna com o pé do Otto. Para tal, e voltando a ter em conta que o buraco que existe no corpo demonstra que é a parte da frente do robot, coloca-se o parafuso inserido no pé voltado para a zona da perna que contem o formato do servo, como demonstra as seguintes imagens.



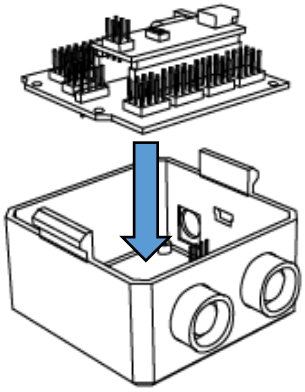
Para completar a fase que une as pernas aos pés, os fios dos micro servos ficam voltados para a zona interior do robot, entrando depois pelas ranhuras existentes no corpo do robot, tal e qual como mostram as imagens seguintes.

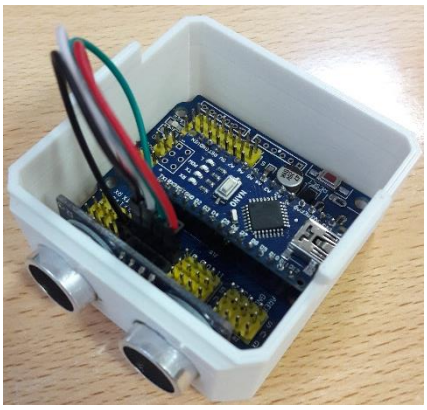


A partir desta etapa, começa-se a inserir a parte da eletrónica dentro do robot. Pode-se começar, por exemplo, pela parte de inserir o Sensor Ultrassónico e o Shield com o Arduíno. Para isso começa-se primeiro por colocar o sensor com os pinos voltados para fora.

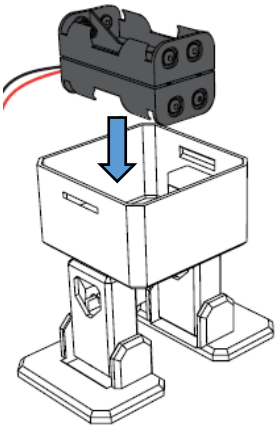


Depois de colocado o sensor, é que se coloca o Shield com o Arduíno. Tem que se ter me conta que o Shield deve ficar junto à base da cabeça do Otto. A entrada para o cabo usb deve condizer com o formato da ranhura da estrutura.



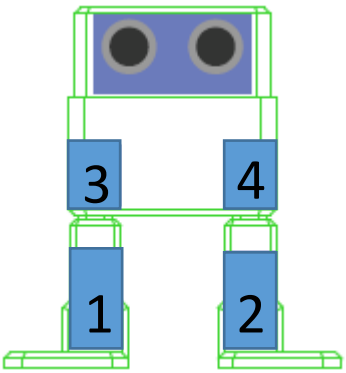


Passando para o suporte de pilhas, dentro do corpo do Otto existe um espaçamento entre os dois servos, sendo esse o sítio utilizado para colocar o suporte para as pilhas.



O resto dos componentes eletrónicos são colocados da melhor maneira possível, não havendo um local específico para os colocar. Tendo os componentes todos colocados dentro do Otto Robot, é só preciso fechá-lo em conjunto com a cabeça, ficando desta forma o robot completo.

De seguida será demonstrado qual será a numeração dos micro servos a usar quando se for programar o Arduíno. Para isso, a imagem seguinte demonstra a numeração que foi usada para cada um dos micro servos.



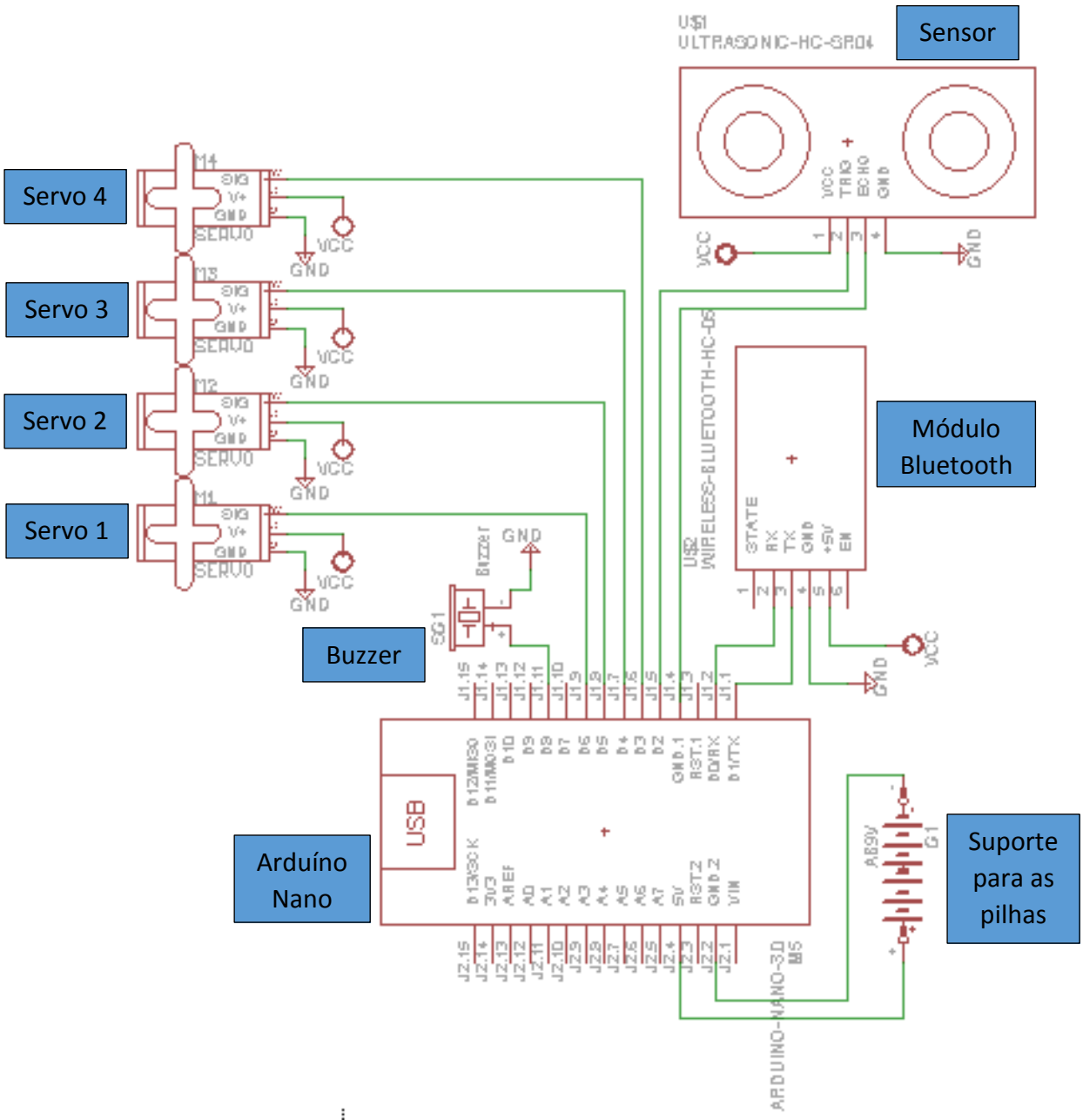

```
Servo myservo1;
Servo myservo2;
Servo myservo3;
Servo myservo4;


myservo1.attach(9);
myservo2.attach(8);
myservo3.attach(7);
myservo4.attach(6);
```

Este processo de númeração dos micro servos é importante porque quando se começar a programar o Otto Robot, não pode haver a troca na ligação dos pinos dos micro servos, e assim existe a numeração como foi vista na imagem anterior, para que depois se faça a correspondência com o código do Arduíno.

Circuito do Otto Robot

Será agora demonstrada a representação do circuito num software que simula ciruitos, onde será possível visualizar as ligações que devem ser feitas para uma montagem correta do Otto Robot. A demonstração foi feita apenas para a ligação direta ao Arduíno Nano, mas basta corresponder o número do pino do Arduíno ao Shield, isto acontece também para as alimentações dos componentes, que é só fazer a ligação do positivo à letra “V” (*VCC*) e o negativo à letra “G” (*GND*).



Programação Arduíno

1ª Fase

Linguagem C (Arduíno)

A linguagem C é uma linguagem de programação estruturada e padronizada pela ISO, em 1972, na altura para desenvolver o sistema operativo Unix. É uma linguagem que fornece acesso de baixo nível à memória e baixos requisitos de *hardware*, tendo como seu principal objetivo, facilitar a criação de programas extensos, com menos erros, recorrendo ao paradigma da programação procedimentar. Desta forma, a linguagem utilizada para a criação de um código no software do Arduíno é a linguagem C.

Software Arduíno

É uma aplicação multiplataforma desenvolvida e escrita em Java. Encontra-se com uma biblioteca designada de "*Wiring*", no qual tem a capacidade de se programar em linguagem C ou C++. Desta forma, é possível criar com mais facilidade as diversas operações de entrada e saída, em que para se concretizarem necessitam de duas funções para se fazer um programa funcional.

Instruções Principais

void setup () – Esta função encontra-se posicionada no início de qualquer programação, no qual tem como principal função a configuração dos pinos a ser utilizados, e ainda que função tem cada um desses pinos, onde podem ser pinos de entrada (*Input*) ou pinos de saída (*Output*).

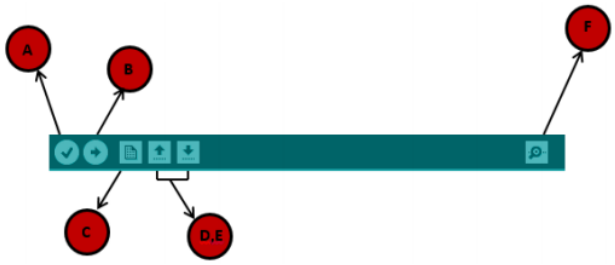
void loop () – esta função encontra-se localizada logo a seguir à função referida anteriormente, no qual serve para introdução principal do código, ou seja, é nesta função que se insere o código a ser executado para o projeto elaborado.

Para além destas duas principais funções para um bom funcionamento do programa, existe ainda a **introdução, opcional, de variáveis ao programa.** Uma variável é uma forma de nomear e armazenar um valor numérico para uso posterior do programa. As variáveis são números que podem ser continuamente alterados em oposição à constante cujo valor nunca muda. As variáveis são colocadas antes do *void setup ()*.

Estrutura do *Software*

O *software* de programação do Arduíno necessita de se ter em atenção, se existe a comunicação com a porta série e qual é o tipo de placa Arduíno que se pretende programar, estes dois aspetos são importantes porque sem eles a transferência do programa para a placa não existe.

Tendo um programa realizado e pronto para ser compilado, existe uma barra de interação com o programa. Nessa barra existem 6 botões sendo eles:



A – Este botão tem como função, verificar todo o código desenvolvido para ver se há algum erro na escrita do código ou se há falta de funções necessárias para o bom funcionamento do programa. Convém ser utilizado antes da compilação do programa.

B – Este botão tem como principal função, compilar o programa desenvolvido, de maneira a que o programa seja implementado na placa do Arduíno e que se consiga colocar a funcionar o projeto.

C – Este terceiro botão tem como função, ser uma maneira mais fácil de se abrir um novo programa, funcionando como um atalho, não sendo necessário abrir a uma das barras de tarefas para o fazer.

D – Tem como função abrir programas que já foram realizados, podendo ser eles programas exemplares existentes no software de programação do Arduíno ou então programas criados anteriormente pelo usuário.

E – Com este botão existe a possibilidade de se guardar o programa a ser realizado, voltando a ser um botão de atalho, como foi referido para o botão C.

F – Este botão serve para que se abra uma janela à parte do programa, onde serão lá colocados todos os dados que se pretende e que tenham sido programados para aparecer nesta janela. Exemplo: Envio de caracteres a partir de uma aplicação, via Bluetooth, aparecem nesta janela caso tenham sido programados a para isso.

Objetivo da programação

Com esta programação tenciona-se que, a partir da aplicação criada no MIT App Inventor, sejam enviados caracteres, via *Bluetooth*, fazendo com que as funções predefinidas e programadas no código sejam executadas, colocando neste propósito, o robot em andamento através das quatro direções principais, ou então colocar o robot a dançar seguindo uma certa sequência de passos.

Void Setup – Programação

Normalmente, para a realização de um programa no Arduíno é necessário predefinir as variáveis. Para tal, e neste caso, as variáveis a predefinir foram as dos servos, alterando apenas os nomes dos servos, mais numa ótica de orientação. De seguida, utilizou-se as váriaveis "*int*" para a configuração dos pinos a utilizar para o sensor e buzzer, colocando ainda uma variável "*char*", com resultado igual a zero, para teste dos caracteres. Este processo acontece sempre antes do *void setup*.

Para o *void setup*, configura-se, neste caso, os pinos que se devem usar, caso não tenham sido configurados anteriormente, que tipos de pinos de pretende utilizar, podendo ser eles de entrada ou de saída, em que no caso do pino *ECHO* do sensor o tipo de pino a utilizar é de entrada, pois é o responsável pelo envio de dados para o microcontrolador, sendo depois os outros pinos de saída, e ainda existe a configuração da velocidade de comunicação da porta serial.

Void Loop – Programação

Saltando para a parte onde é feita a execução das funções do programa, é nesta área onde se coloca as funções que se pretende que sejam realizadas. Para tal, foram programadas as funções do sensor, do buzzer, dos servos e dos caracteres a serem recebidos.

Para o caso do sensor, é feito o envio de sinal por parte do pino *TRIG*, colocando-o sempre ativado e desativado a cada 10 microsegundos, para que seja testado se há objeto ou não. Em sentido contrário, o pino *ECHO* encontra-se sempre ativado para a receção de dados, dando esses dados em formato de tempo.

A distância do objeto ao sensor sabe-se graças à conta que é realizada, tendo ela sido demonstrada, anteriormente, na explicação do sensor ultrassónico.

```
digitalWrite(trigPin, LOW);
delayMicroseconds(10);
digitalWrite(trigPin, HIGH);

tempo = pulseIn(echoPin, HIGH);

distancia = (tempo*0.0343/2);
```

A partir daqui, se o sensor detetar um objeto a uma distância igual ou inferior a 5 cm, executa a função de colocar o robot a fazer o deslocamento para trás, afastando-se assim do objeto, este recuo é sinalizado pelo buzzer.

```
if (distancia <=5){
  digitalWrite(buzzer, HIGH);
  myservo4.write(45);
  delay(325);
  digitalWrite(buzzer,LOW);
  myservo2.write(135);
  delay(325);
```

De seguida, faz-se a programação dos caracteres a serem recebidos. Para tal, esses caracteres são testados através da variável "*char*", referida anteriormente, para que só sejam recebidos caracteres maiores que zero. Tendo isto feito, elabora-se as condições a ter em conta. Por exemplo, se o valor recebido da aplicação for igual a "1", faz com que seja ativada a função que contem o caracter "1", executando a função de colocar o robot a andar para a frente.

```
if (pos == '1'){
digitalWrite(led, LOW);
myservo3.write(45);
delay(325);
myservo1.write(45);
delay(325);
myservo3.write(90);
delay(325);
```

Este processo repete-se para todos os outros caracteres existentes, mudando-se apenas as funções a serem executadas.

Será demonstrado de seguida um fluxograma capaz de refletir todo o procedimento que o programa do Arduíno executa para colocar o Otto Robot em funcionamento.

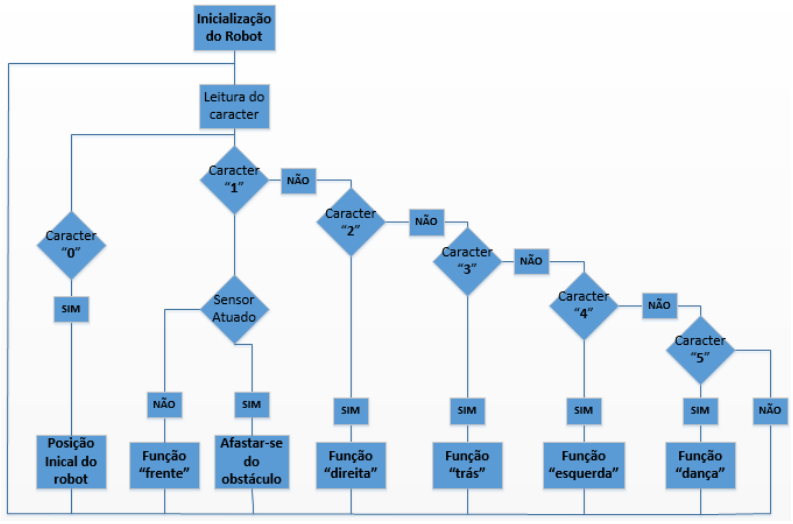


Fig. 10 – Fluxograma do código Arduíno

2ª Fase

MIT App Inventor 2

É uma aplicação de código aberto criado pela Google, em que atualmente é mantida pela *Massachusetts Institute of Technology* (MIT). Permite a iniciantes de programação, que criem aplicações de *software* para o sistema operativo Android, usando uma interface gráfica, de arrastar e juntar blocos para a criação de uma aplicação *Android*.

A plataforma App Inventor ainda se encontra algo amadora, focando-se apenas no desenvolvimento de aplicações para pesquisas escolares.

Design da Aplicação

A aplicação final para o funcionamento do Otto Robot tem como principal objetivo, enviar caracteres de maneira a que seja possível ao robot desempenhar as suas funções de sequências de dança e de se deslocar em alguns sentidos.

A primeira fase desta aplicação consistiu no desenvolvimento de um aspeto visual. Para tal, a programação no MIT App Inventor 2, fornece a opção de "*Designer*", onde é possível, neste caso, personalizar ao gosto do criador, o aspeto/*design* da aplicação.

Para o desenvolvimento desta aplicação, foram necessários seis botões, correspondendo um deles à ativação do *Bluetooth*, quatro deles aos movimentos do robot, e outro botão para realizar as sequências de dança. Para além disto foi necessária a ferramenta "*BluetoothClient*" para estabelecer a conexão *Bluetooth*.

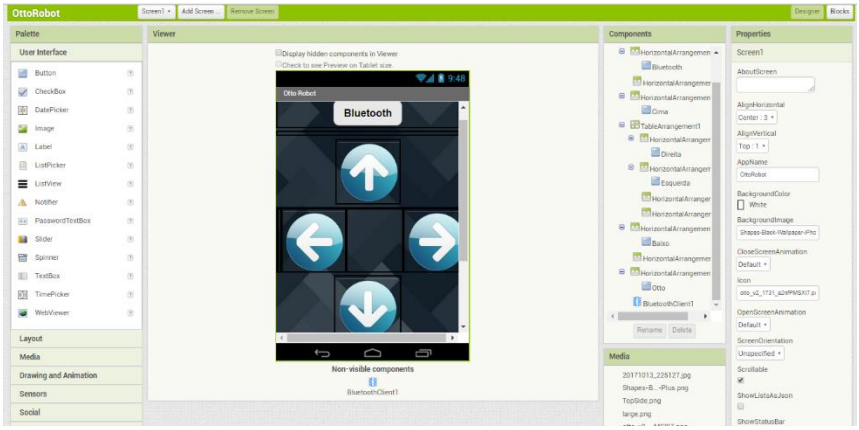


Fig. 11 – Design da aplicação

Blocos da Aplicação

Para além da personalização do aspeto da aplicação, é preciso dar instruções aos componentes inseridos na área de "*Designer*". Para tal, existe outra área no MIT App Inventor 2 designada de "*Blocks*". É na área "*Blocks*" que o utilizador irá definir o comportamento de cada um dos componentes que constituem a aplicação.

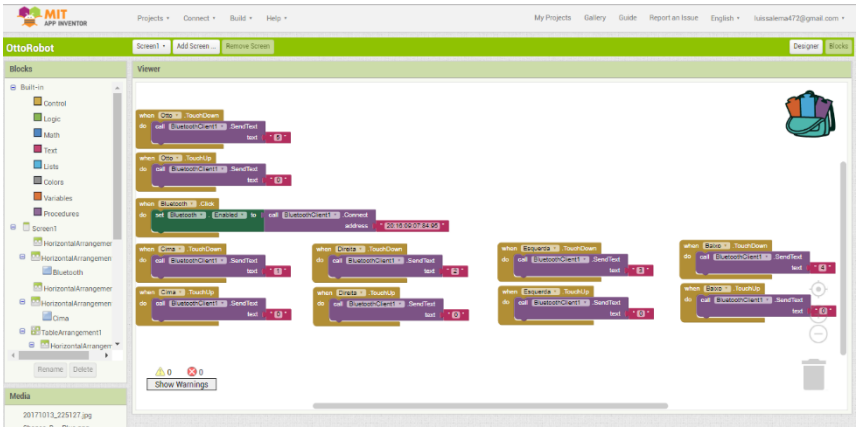


Fig. 12 – Área de blocos

É preciso, para a conexão do módulo *Bluetooth* e a aplicação, de um botão "*Bluetooth*", que quando for pressionado faça ativar o módulo *Bluetooth* e chamar o respetivo *BluetoothClient*, esse mesmo bloco do *BluetoothClient* faz acionar o endereço do módulo *Bluetooth* através de uma caixa de texto, onde se escreve esse mesmo endereço.

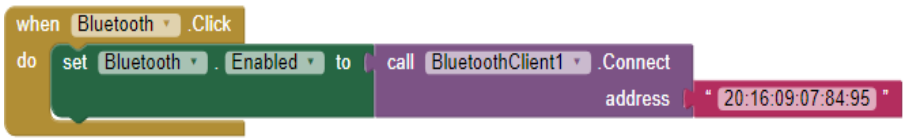


Fig. 13 – Conexão com o módulo Bluetooth

Será agora demonstrado o processo que é feito para que o robot execute, por exemplo, o deslocamento para a frente. Para esse processo é preciso que, enquanto o botão for pressionado execute essa função, fazendo que se chame o *BluetoothClient*, enviando sempre o caracter "1", quando já não houver o acionamento desse botão, volta a ser chamado o *BluetoothClient*, enviando agora o caracter "0" para colocar o robot na sua posição inicial.

Este processo, no MIT App Inventor 2, é chamado de "*TouchDown*", que é quando existe o acionamento do botão para a execução dessa função, e o "*TouchUp*", que é utilizado quando já não existe nada a pressionar o botão.

Este processo é repetido para as outras três direções, apenas havendo a alteração do botão que deve ser pressionado, e o caracter que deve ser enviado para fazer o acionamento da função, tendo sempre em atenção que para se fazer o acionamento da função necessária é preciso usar o "*TouchDown*", e para deixar de executar a função, deve ser utilizado o "*TouchUp*".

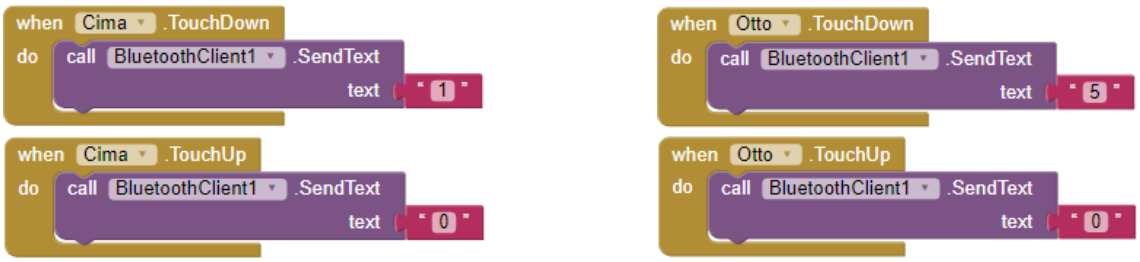


Fig. 14 – Blocos de deslocamento do robot