Blocos

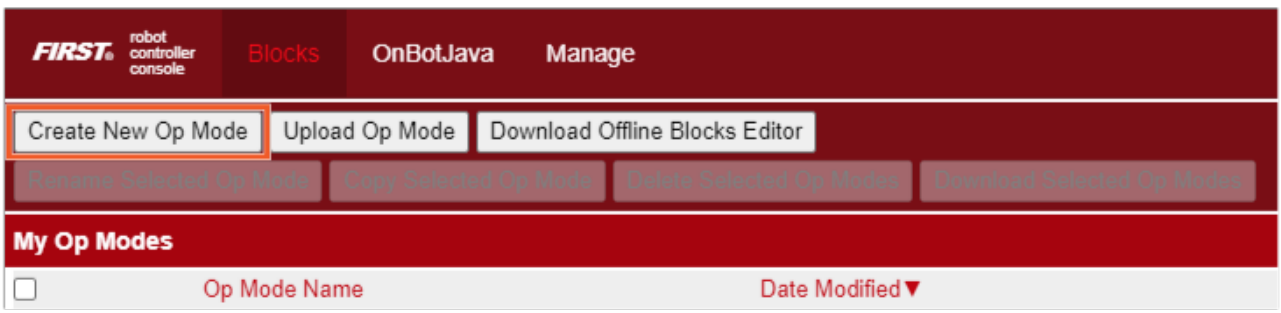
A Ferramenta de Programação por Blocos é uma ferramenta visual de programação que permite aos programadores usar um navegador da web para criar, editar e salvar seus modos operacionais (op modes). Os blocos, assim como outras ferramentas de programação baseadas em blocos, são uma coleção de trechos de código predefinidos que os usuários podem arrastar e soltar na linha de código apropriada. Nesta seção, os usuários podem aprender a criar um modo operacional, bem como os conceitos básicos de programação dos atuadores e sensores apresentados no banco de testes. Siga o guia para obter uma compreensão aprofundada do trabalho com Blocos ou navegue até a seção que atenda às suas necessidades:

Criando um Op Mode

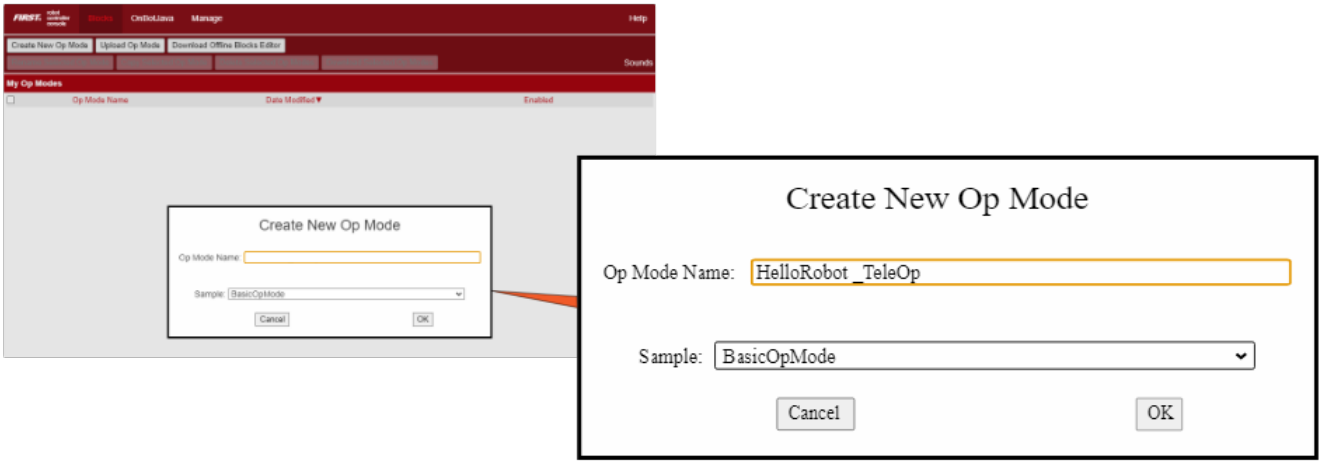
Antes de mergulhar e criar o seu primeiro modo operacional (op mode), é importante considerar o conceito de convenções de nomenclatura. Ao escrever código, o objetivo é ser o mais claro possível sobre o que está acontecendo dentro do código. É aqui que entra o conceito de convenções de nomenclatura. Convenções de nomenclatura comuns foram estabelecidas pelo mundo da programação para denotar variáveis, classes, funções, etc. Modos operacionais compartilham algumas semelhanças com classes. Portanto, a convenção de nomenclatura para modos operacionais tende a seguir a convenção de nomenclatura para classes, onde a primeira letra de cada palavra é maiúscula.

> Esta seção pressupõe que você já acessou a plataforma de Blocos durante a introdução à programação no Guia Olá Robô. Se você não souber como acessar os Blocos, por favor, reveja esta seção antes de continuar.

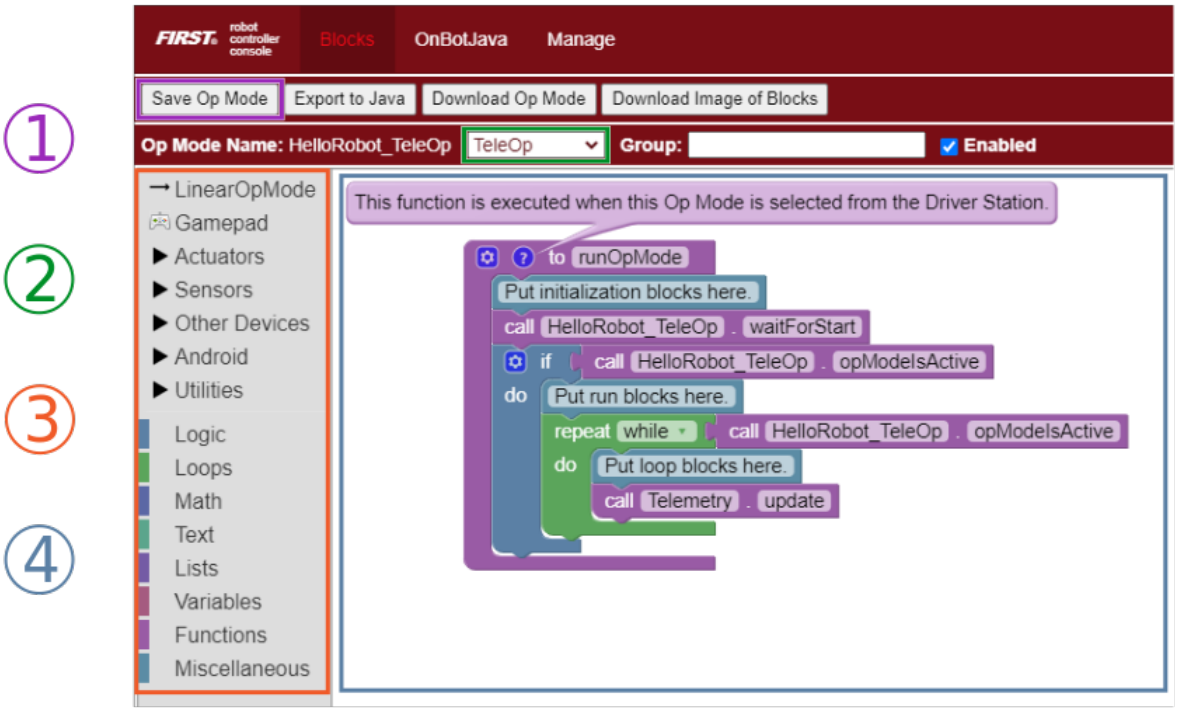
Para começar, acesse o Console do Controlador do Robô e vá para a página de Blocos. No canto superior direito, há um botão "Criar Novo Modo Operacional", clique nele.



Clicar no botão "Criar Novo Modo Operacional" abrirá a janela "Criar Novo Modo Operacional". Esta janela permite que os usuários nomeiem seus modos operacionais e selecionem um código de exemplo para desenvolver. Para este guia, utilize o exemplo padrão "BasicOpMode" e nomeie o modo operacional como "HelloRobot_TeleOp", conforme mostrado na imagem abaixo.



Depois de nomear o modo operacional, clique em 'OK' para prosseguir. A criação de um modo operacional abrirá a página principal de programação por blocos. Antes de prosseguir com a programação, reserve um tempo para aprender e entender os seguintes componentes-chave dos Blocos, apresentados na imagem abaixo.



Salvar Modo Operacional - Clique neste botão para salvar um modo operacional no robô. É importante salvar o modo operacional sempre que parar de trabalhar em um código para que o progresso não seja perdido.

TeleOperado/Autônomo - Esta seção de blocos permite aos usuários alternar entre os dois tipos de modos operacionais: teleoperado e autônomo.

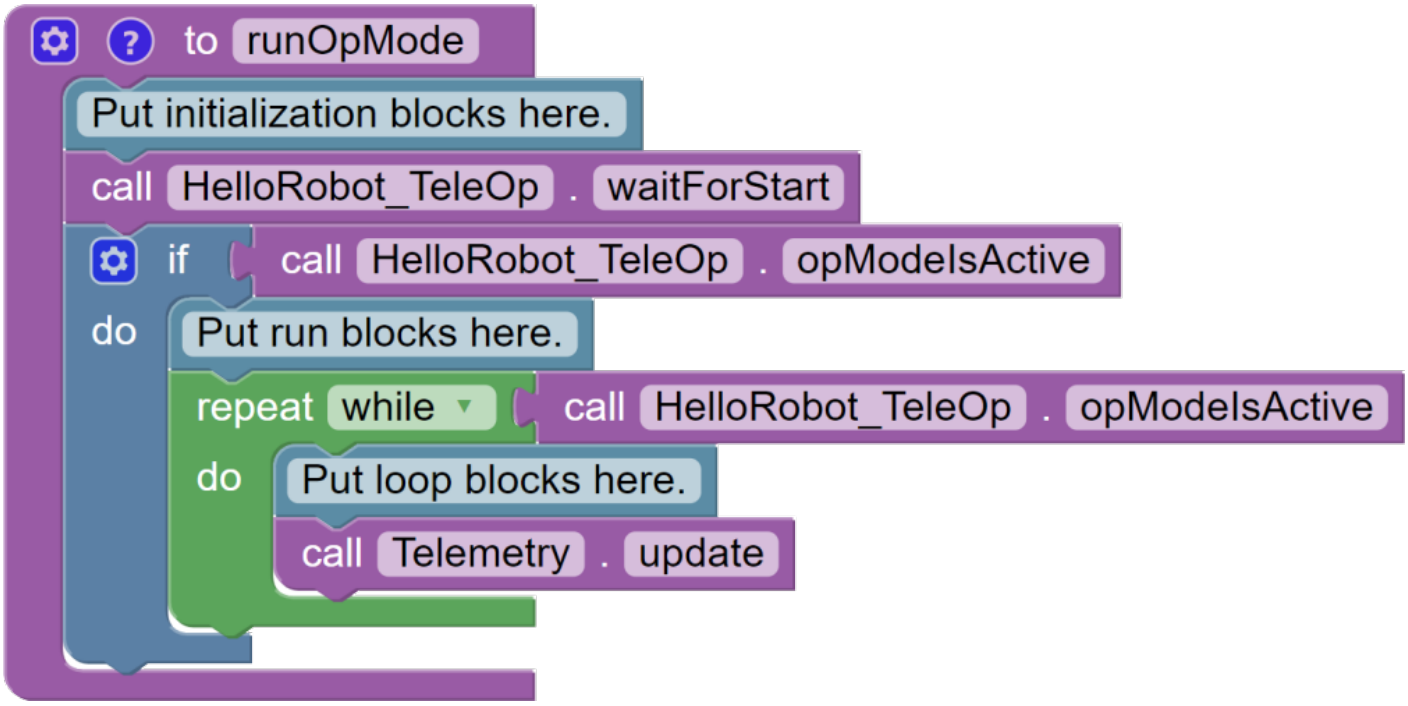
Blocos Categorizados - Esta seção da tela é onde os blocos de programação são categorizados e acessíveis. Por exemplo, clicar em Lógica abrirá o acesso a blocos de programação como declarações if/else.

Espaço de Programação - Este espaço é onde os blocos são adicionados para construir programas.

> Se uma configuração foi feita, os Atuadores, Sensores e Outros Dispositivos na seção de Blocos Categorizados devem aparecer como menus suspensos, onde blocos específicos para hardware podem ser acessados. Se isso não acontecer, um arquivo de configuração não foi criado. Para obter mais informações, visite a página de Configuração antes de prosseguir com a programação.

Fundamentos da programação

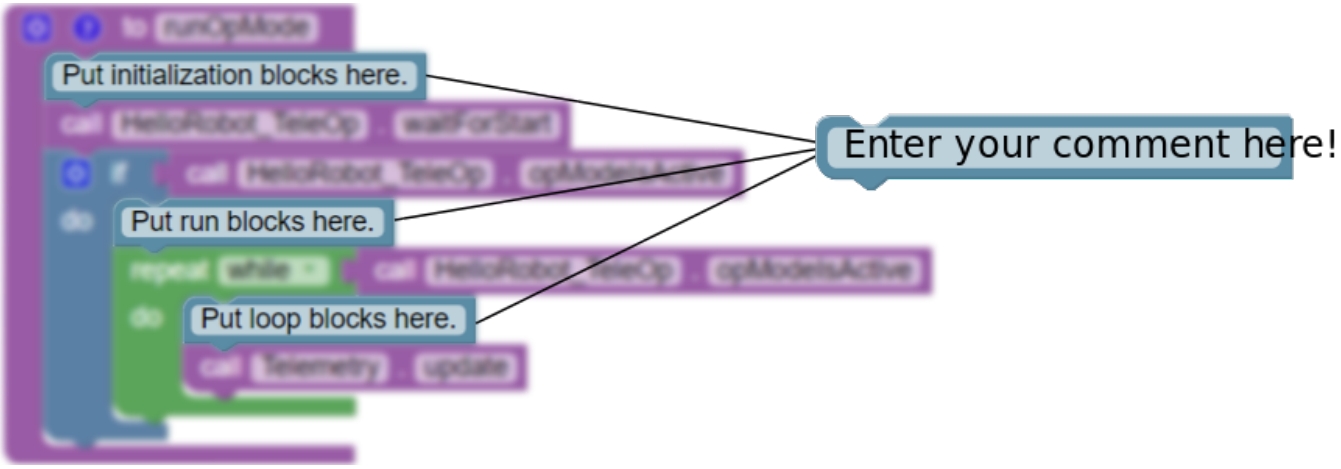
Durante o processo de criação de um modo operacional, a ferramenta de Blocos solicitou a seleção de um código de exemplo. Nos Blocos, esses exemplos funcionam como modelos, fornecendo os blocos e a estrutura lógica para diferentes casos de uso em robótica. Na seção anterior, o código de exemplo BasicOpMode foi selecionado. Este código de exemplo, visto na imagem abaixo, é a estrutura base necessária para ter um modo operacional funcional.



Um modo operacional pode frequentemente ser considerado um conjunto de instruções para um robô seguir a fim de entender o mundo ao seu redor. O BasicOpMode fornece o conjunto inicial de

instruções necessárias para que um modo operacional funcione adequadamente. Embora este exemplo seja fornecido aos usuários para reduzir algumas das complexidades da programação à medida que aprendem, ele introduz alguns dos blocos de código mais importantes. Também é importante entender o que está acontecendo na estrutura do BasicOpMode, para que os blocos de código sejam colocados na área correta.

Principais blocos do modo operacional



Comentários são blocos de código que beneficiam o usuário humano. Eles são usados pelos programadores para explicar a função de uma seção de código. Isso é especialmente útil em ambientes de programação colaborativa. Se o código é repassado de um programador para outro, os comentários comunicam a intenção do código ao outro programador. Blocos como // são comentários escritos pela equipe técnica da FIRST para informar ao usuário o que acontecerá quando blocos forem adicionados diretamente abaixo do comentário.

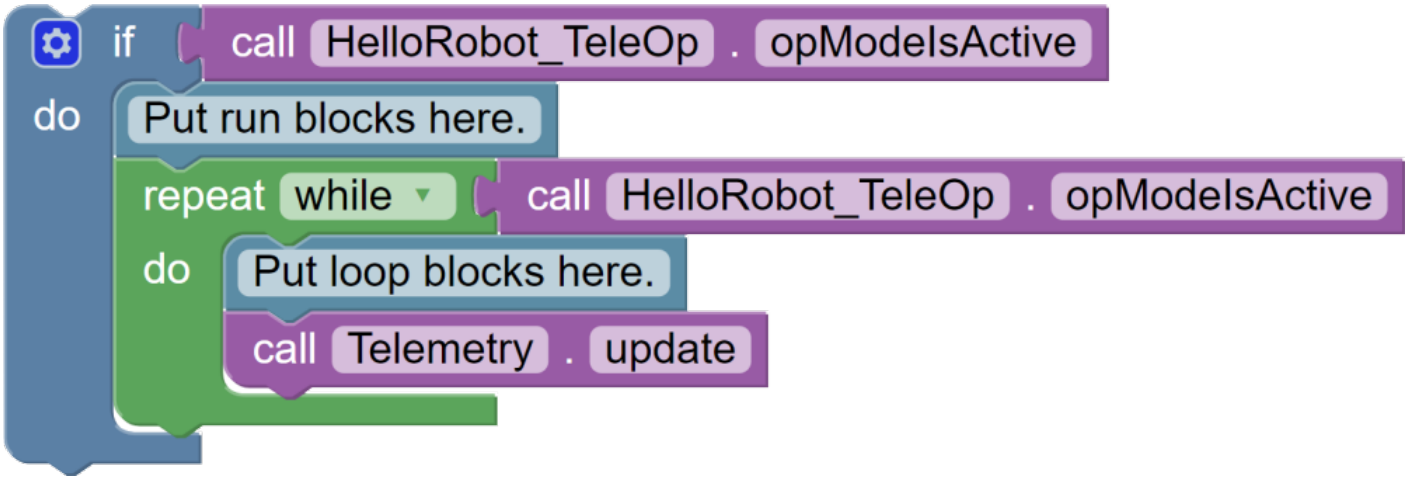
Por exemplo, quaisquer blocos de programação que são colocados após o comentário (e antes do *waitForStart* bloco) serão executados quando o modo operacional for selecionado pela primeira vez por um usuário na Estação do Condutor. Normalmente, os blocos colocados nesta seção têm o propósito de criar e definir variáveis entre as fases de inicialização e início do modo operacional.

> Uma variável é uma localização de armazenamento com um nome simbólico associado, que contém alguma quantidade conhecida ou desconhecida de informações referidas como um

valor. Variáveis podem ser números, caracteres ou até mesmo motores e servos.

Quando o Controlador do Robô atinge o bloco waitForStart, ele irá parar e aguardar até receber um comando de Iniciar da Estação do Condutor. Um comando de Iniciar não será enviado até que o usuário pressione o botão Iniciar na Estação do Condutor. Qualquer código após o bloco waitForStart será executado após o botão Iniciar ser pressionado.

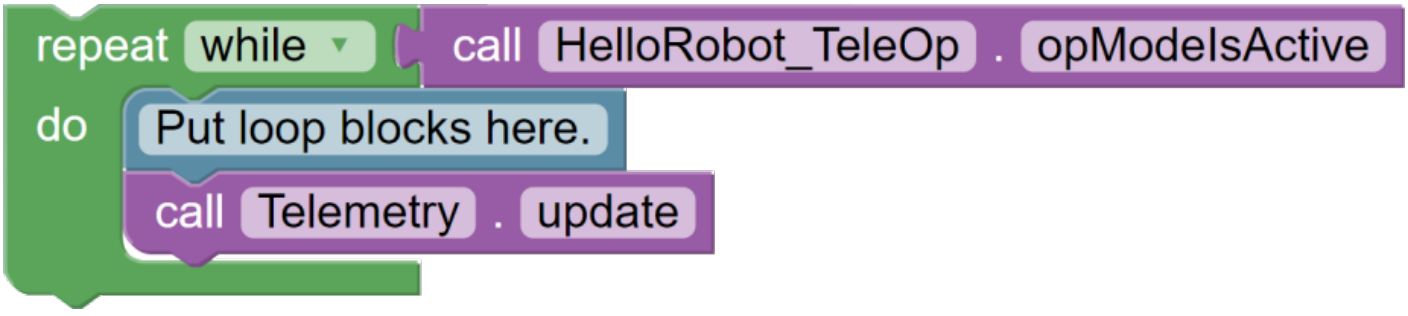
Após o bloco waitForStart, há um bloco condicional if que só será executado se o modo operacional ainda estiver ativo (ou seja, um comando de parada não foi recebido).



As instruções if-then (if-else) são semelhantes ao conceito de causa e efeito. Se a causa (ou condição) acontecer, então execute o efeito.

Quaisquer blocos que são colocados após o comentário // *pur run blocks here* e antes do bloco // *OpMode is active* serão executados sequencialmente pelo Controlador do Robô depois que o botão Iniciar for pressionado.

O bloco while é uma estrutura de controle iterativa ou de loop.



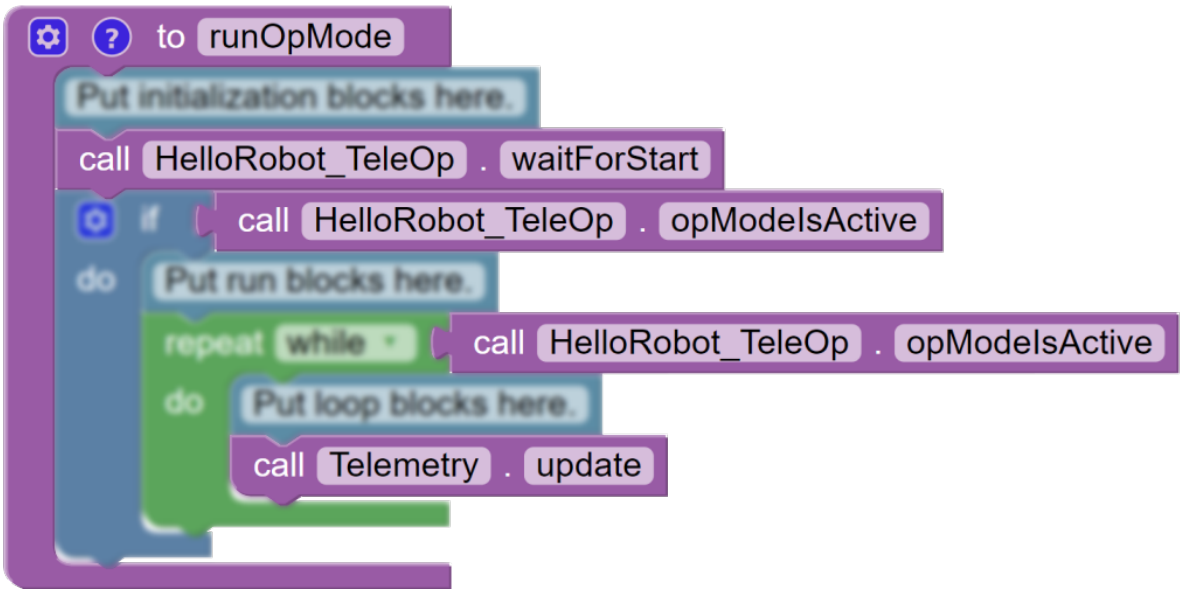
Este controle realizará as etapas listadas na parte "do" do bloco enquanto a condição **opModeIsActive()** for verdadeira. Isso significa que as instruções incluídas na parte "do" do bloco serão repetidamente executadas enquanto o modo operacional HelloRobot_TeleOp estiver em execução.

Assim que o usuário pressiona o botão Parar, a condição **opModeIsActive()** não é mais verdadeira e o loop while para de se repetir.

Funções e métodos

A seção anterior não entrou em uma discussão detalhada dos blocos de função (ou método) roxos. Funções e métodos são procedimentos semelhantes em programação que são mais avançados do que o que será abordado neste guia.

Por enquanto, a coisa mais importante a saber é que ocasionalmente será necessário chamar métodos dentro das bibliotecas SDK para realizar uma determinada tarefa. Por exemplo, a linha while (opModeIsActive()) chama o método opModeIsActive, que é o procedimento no SDK que consegue dizer quando o robô foi iniciado ou parado.



Quando suas habilidades de programação estiverem mais avançadas, reserve um tempo para explorar os conceitos de funções e métodos, e descubra como eles podem ajudar a aprimorar seu código.

Programando atuadores

Noções básicas do servo

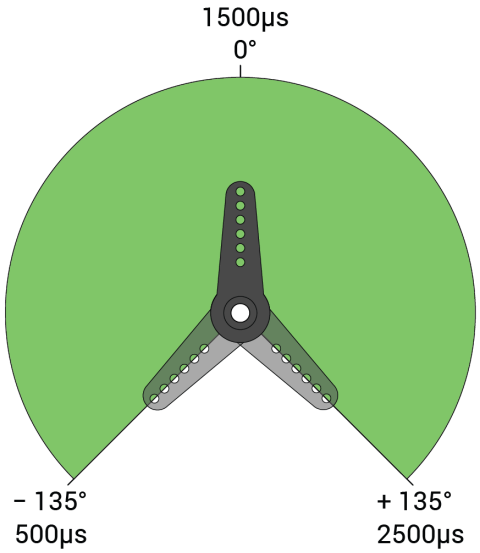
O objetivo desta seção é cobrir alguns dos conceitos básicos de programação de um servo nos Blocos. No final desta seção, os usuários devem ser capazes de controlar um servo com um gamepad, além de compreender algumas das necessidades fundamentais de programação do servo.

> Esta seção está considerando o Smart Robot Servo no seu modo padrão. Se o seu servo foi alterado para funcionar no modo contínuo ou com limites angulares, ele não se comportará da mesma forma utilizando os exemplos de código abaixo. Você pode aprender mais sobre o Smart Robot Servo ou alterar o modo do servo através do SRS Programmer clicando nos hiperlinks abaixo.

Servo robô inteligente

Programador SRS

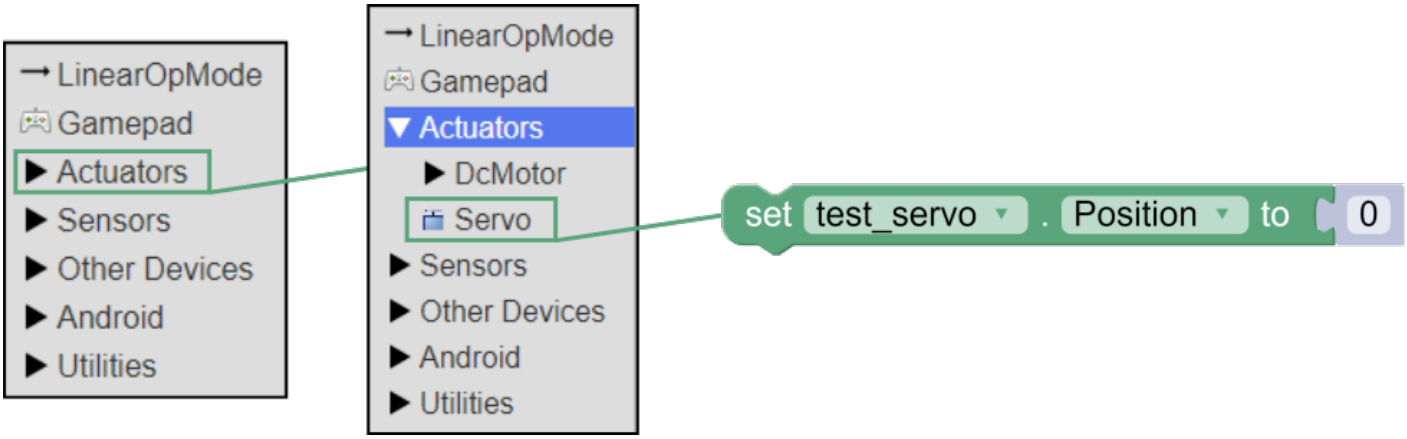
Com um servo típico, você pode especificar uma posição alvo para o servo. O servo girará o eixo do motor para mover-se até a posição alvo e, em seguida, manterá essa posição, mesmo se for aplicada uma força moderada para tentar perturbar sua posição.



Para ambos os Blocos e o OnBot Java, você pode especificar uma posição alvo que varia de 0 a 1 para um servo. Para um servo com um alcance de 270°, se a faixa de entrada for de 0 a 1, então um sinal de entrada de 0 faria com que o servo virasse para -135°. Para um sinal de entrada de 1, o servo viraria para +135°. Entradas entre o mínimo e o máximo têm ângulos correspondentes distribuídos uniformemente entre o ângulo mínimo e máximo do servo. Isso é importante ter em mente ao aprender a programar servos.

Como esta seção se concentrará em servos, é importante entender como acessar servos dentro dos Blocos. No topo da seção de Blocos Categorizados, há um menu suspenso para Atuadores. Quando o menu é selecionado, ele mostrará duas opções: DcMotor ou Servo. Selecionar Servo

abrirá uma janela lateral preenchida com vários blocos relacionados a servos.

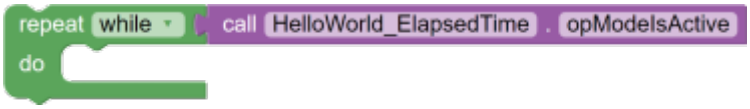


Programando um servo

No menu Servo selecione o bloco

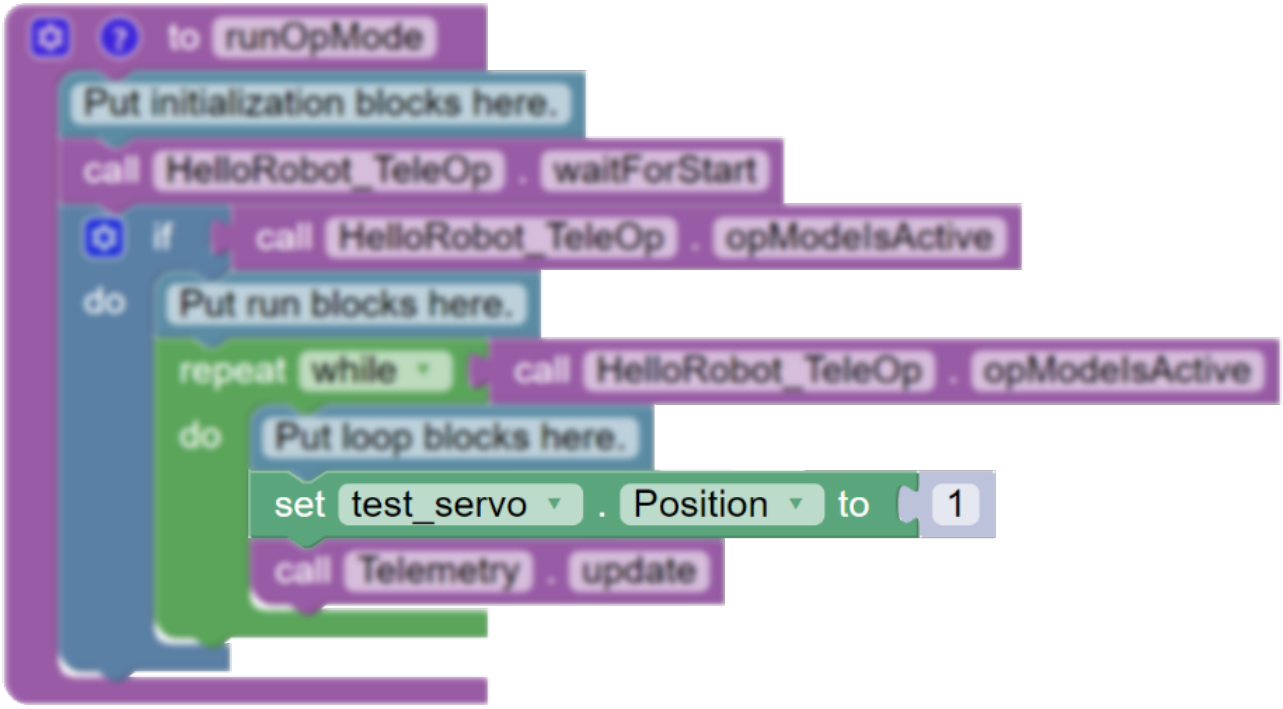
> O bloco acima mudará de nome dependendo do nome do servo em um arquivo de configuração. Se houver vários servos em um arquivo de configuração, a seta ao lado de "test_servo" irá abrir um menu com todos os servos na configuração.

Adicione este bloco ao código do modo operacional dentro do



Clique no bloco numérico para mudar de

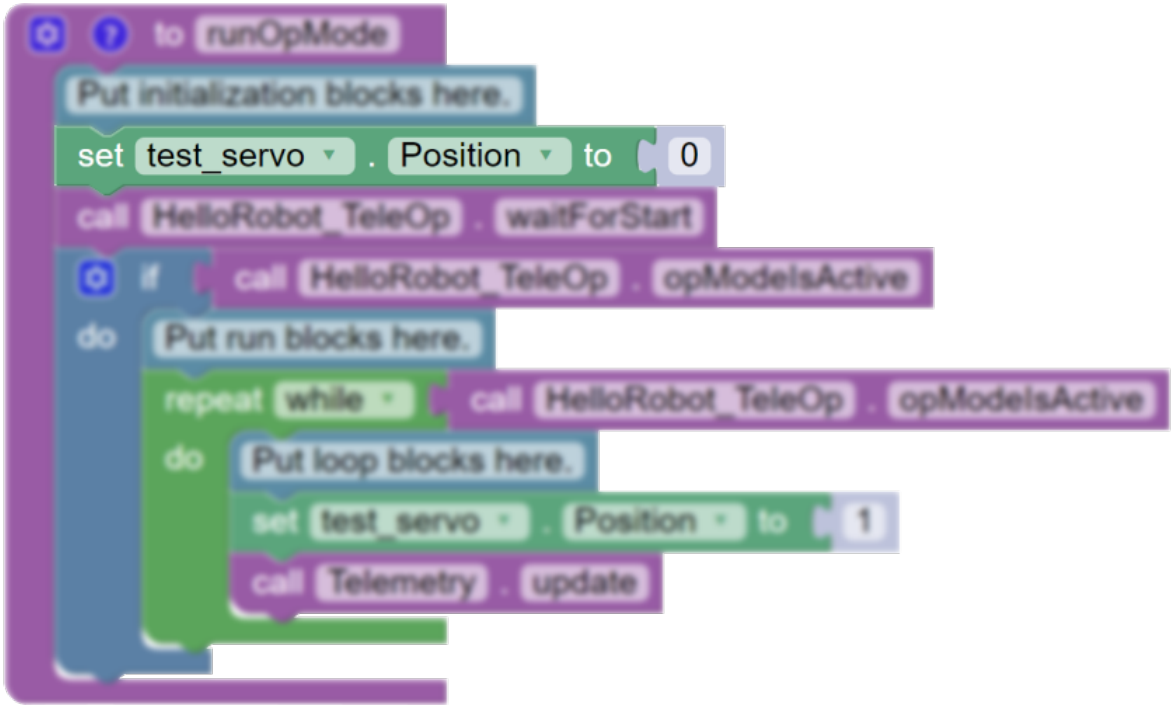
Para:



Selecione "Salvar Modo Operacional" no canto superior direito no Console do Controlador do Robô.

> Tente executar este modo operacional no banco de testes duas vezes e considere as seguintes perguntas:
>
> - O servo se moveu durante a primeira execução?
> - O servo se moveu durante a segunda execução?
> - Se o servo não se moveu, mude de volta para e tente novamente.

A intenção do bloco setPosition é definir a posição do servo. Se o servo já estiver na posição definida quando o código for executado, ele não mudará de posição. Vamos tentar adicionar outro bloco setPosition e ver o que muda. Arraste mais um bloco setPosition para o código do modo operacional, abaixo do bloco de comentário *Put initialization blocks here*.



> Tente executar este modo operacional no banco de testes e considere a seguinte pergunta: O que é diferente da execução anterior?

O bloco setPosition(0) que foi adicionado na etapa acima altera a posição do servo para 0 durante a fase de inicialização, portanto, quando o modo operacional é executado, o servo sempre se moverá para a posição 1. Para algumas aplicações, iniciar o servo em um estado conhecido, como na posição zero, é benéfico para o funcionamento de um mecanismo. Definir o servo no estado conhecido na inicialização garante que ele esteja na posição correta quando o modo operacional é executado.

Programando um servo com um controle

O foco deste exemplo é atribuir determinadas posições do servo a botões no gamepad. Para este exemplo, o estado conhecido permanecerá na posição 0, de modo que após a inicialização o servo estará na posição de -135 graus do intervalo do servo. A lista a seguir mostra quais botões correspondem a quais posições do servo.

> Se você estiver usando um controle PS4, como o Etpark Wired Controller para PS4 (REV-39-1865), consulte a seção Usando Controles na página anterior para determinar como o código do controle usado nesta seção se traduz para o controle do PS4.

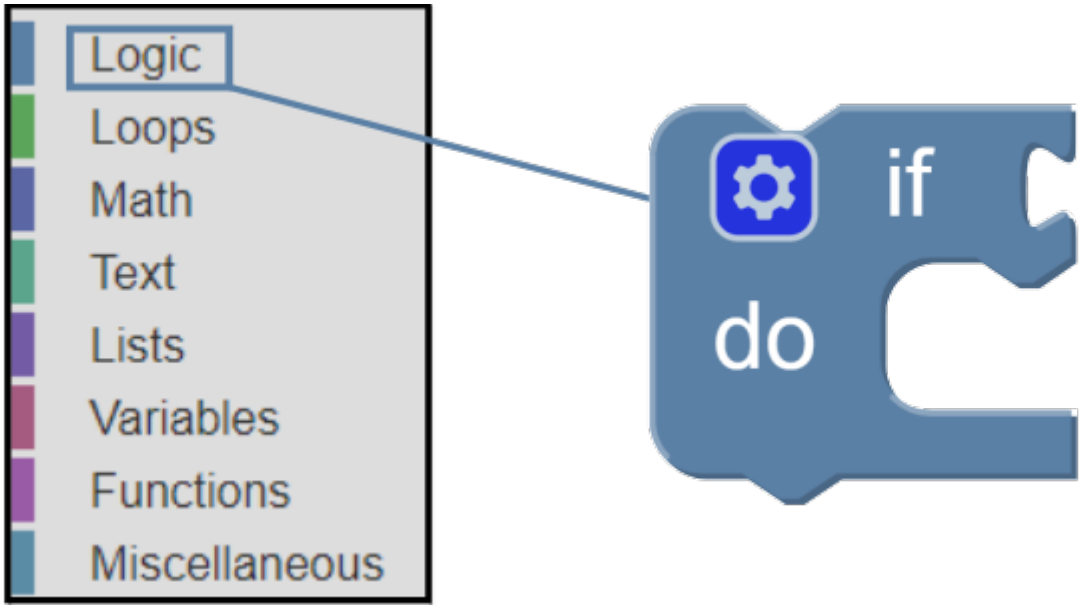
Botões	Posição em graus	Posição no código

Y	-135	0
X	0	0.5
B	0	0.5
A	135	1

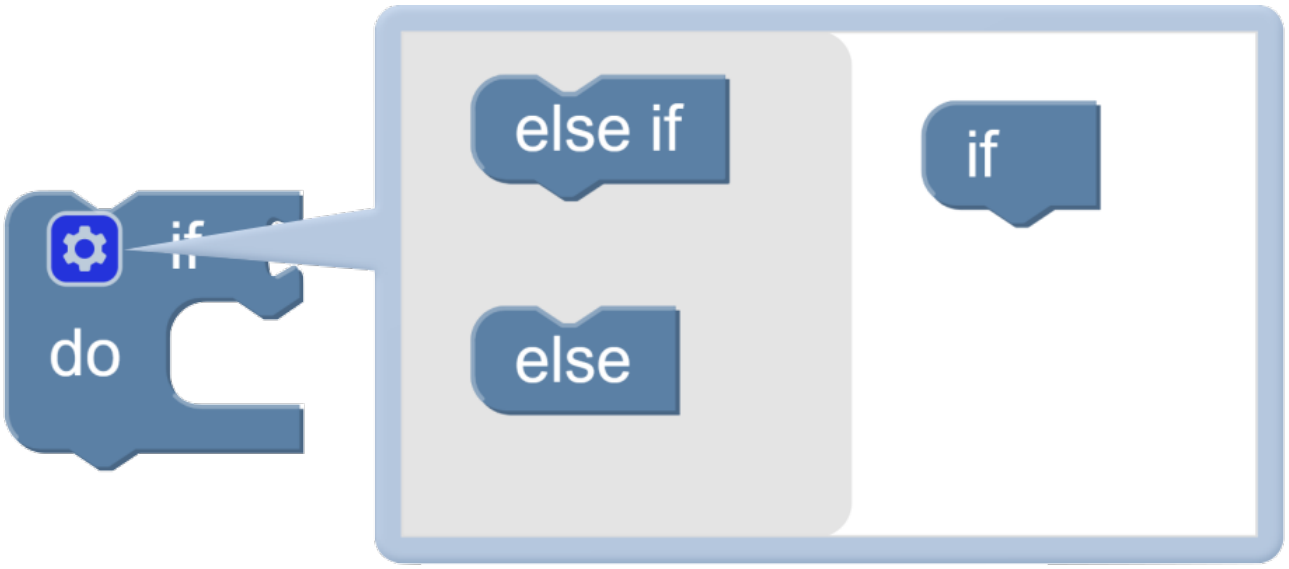
A melhor maneira de alternar a posição do servo será usar uma instrução condicional if/else if. Uma instrução if avalia se uma declaração condicional é verdadeira ou falsa. Se a declaração condicional for verdadeira, uma ação definida (como o movimento do servo) é executada. Se a declaração condicional for falsa, a ação não é executada.

Uma instrução if/else if aceita várias declarações condicionais diferentes. Se a primeira declaração condicional for considerada falsa, então a segunda declaração condicional é analisada. Cada declaração no if/else if será analisada uma por uma até que uma declaração seja considerada verdadeira ou todas as declarações sejam consideradas falsas. Para este exemplo, haverá três condições que precisarão ser verificadas.

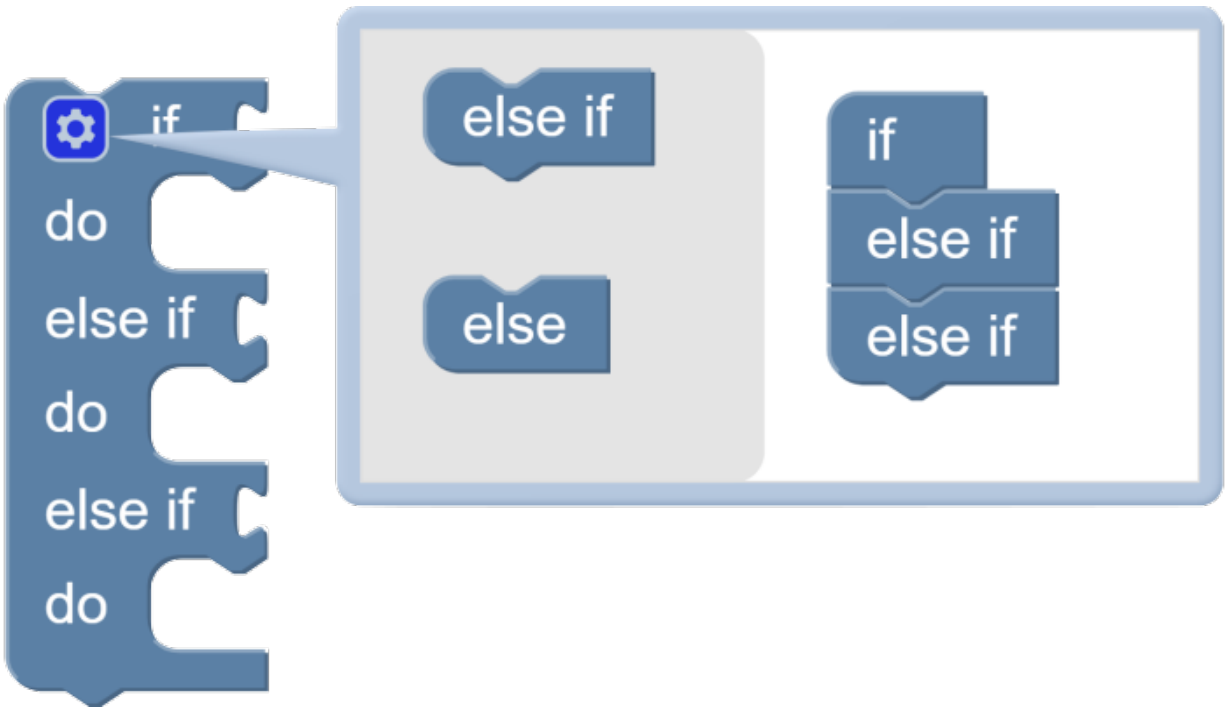
No menu Lógica em Blocos, selecione o bloco if e arraste-o para dentro do loop while do modo operacional.



Clique no ícone de Configurações azul e branco para o bloco if/else if. Isso exibirá um menu pop-up que permite modificar o bloco if/else if.



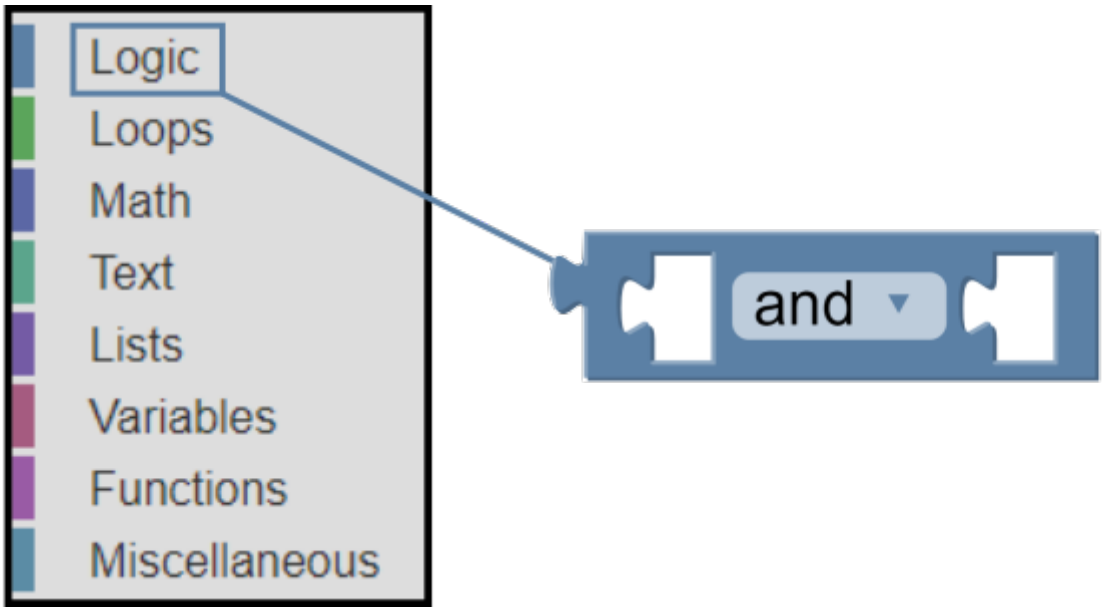
Arraste um bloco else if da parte esquerda do menu pop-up e encaixe-o sob o bloco if. Arraste um segundo bloco else if da parte esquerda e encaixe-o no lado direito sob o primeiro bloco else if.



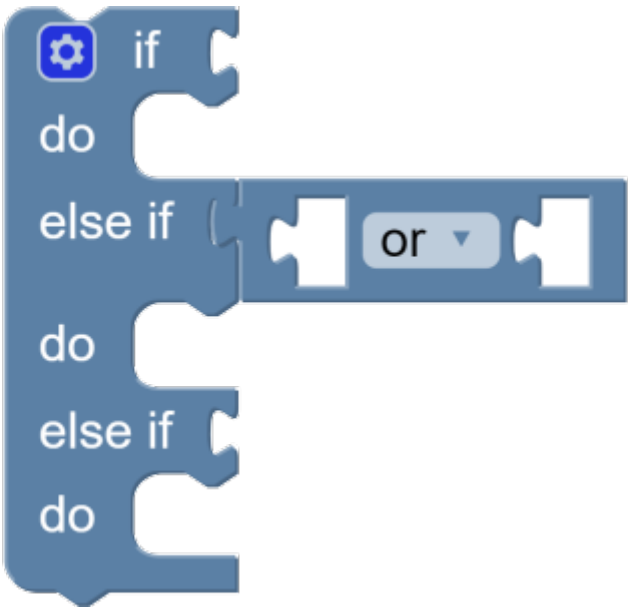
Existem três caminhos diferentes neste bloco if/else if. Cada um corresponde a uma das três posições escolhidas do servo: 0, 0.5 e 1. No entanto, existem quatro botões diferentes que serão usados para este exemplo. Tanto o botão B quanto o botão X devem ser capazes de mover o servo para a posição 0.5. Para fazer isso, o operador lógico "or" precisa ser usado.

> O operador lógico "or" considera dois operandos; se qualquer um (ou ambos) forem verdadeiros, a declaração "or" é verdadeira. Se ambos os operandos forem falsos, a declaração "or" é falsa.

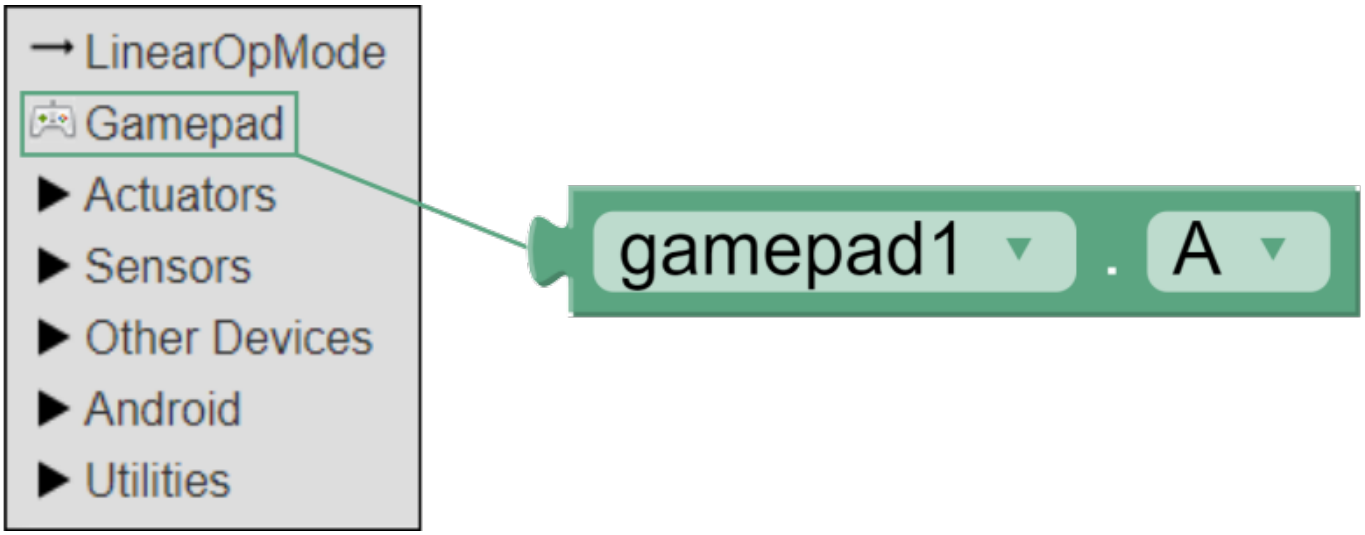
Da categoria Lógica no Blocks, selecione o bloco and.



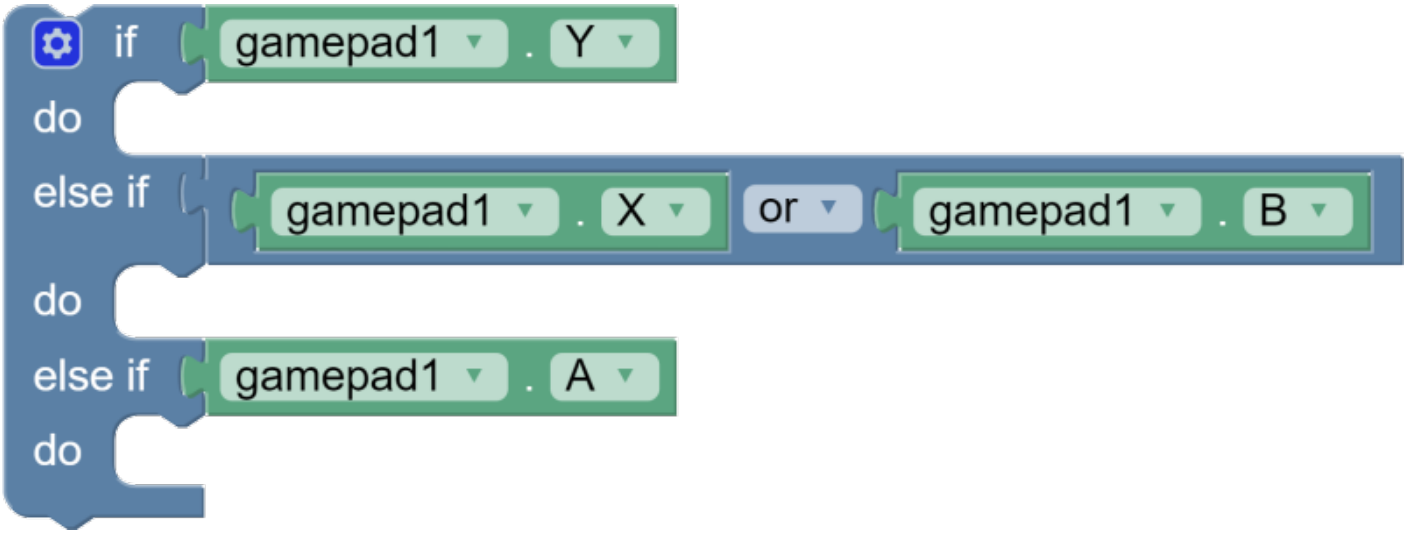
Adicione este bloco ao bloco if/else if, como mostrado na imagem abaixo. Use o menu suspenso no bloco para alterá-lo de um bloco and para um bloco or.



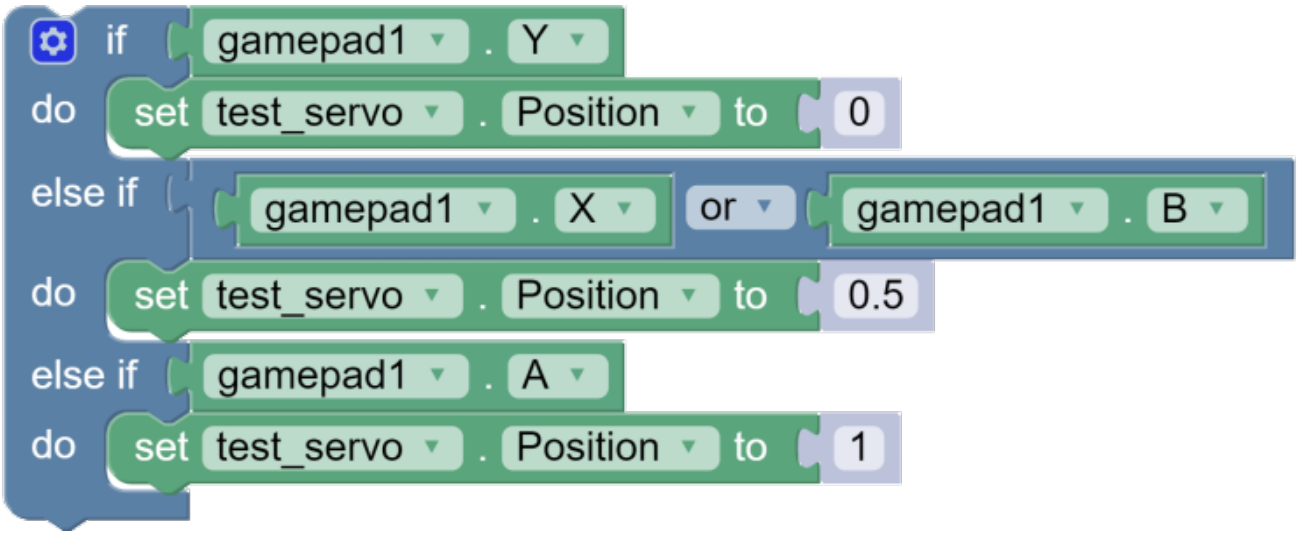
Todos os blocos relacionados ao gamepad estão no Menu do Gamepad.



Adicione cada botão ao bloco if/else if conforme visto na imagem abaixo.

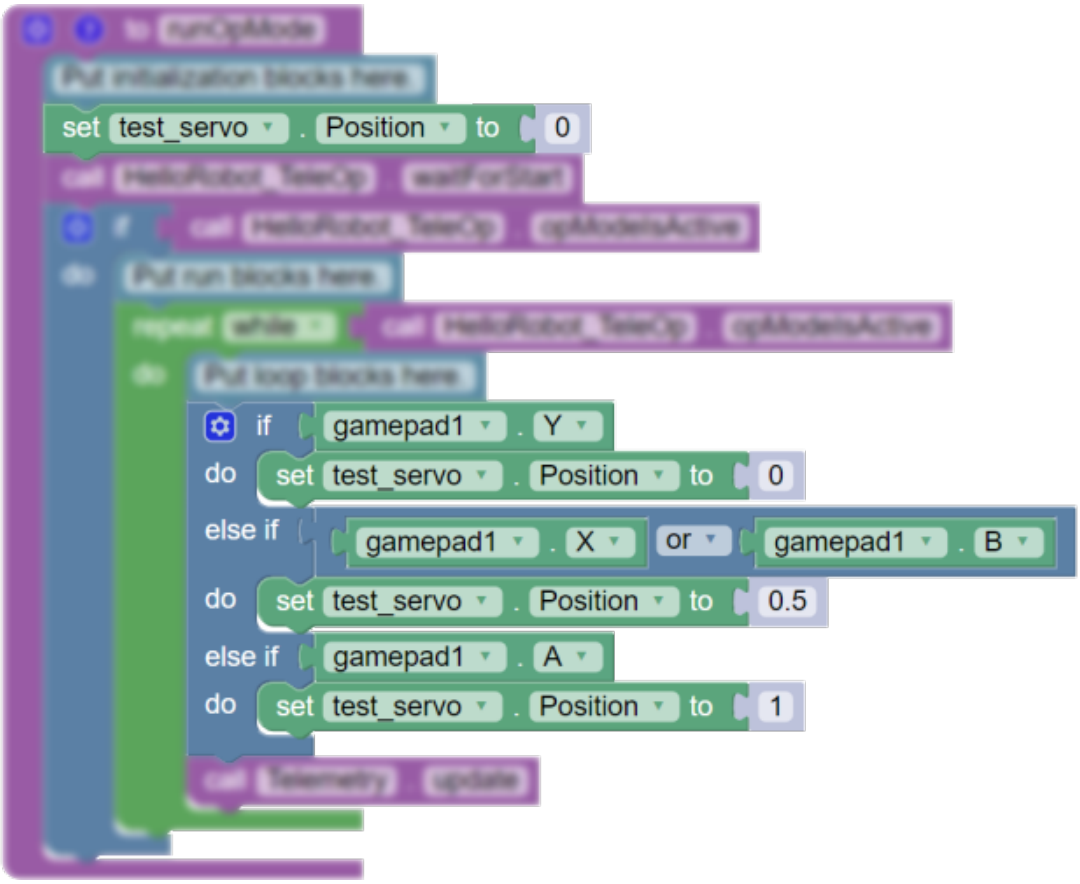


Adicione blocos de setar a posição do servo para 1 em cada seção do bloco if/else if. Defina a posição do servo para corresponder a cada botão.



Existem três caminhos diferentes neste bloco if/else if. Se a primeira declaração condicional for verdadeira (o botão Y está pressionado), o servo se move para a posição de código 0 e as outras

declarações condicionais são ignoradas. Se a primeira condição for falsa (o botão Y não está pressionado), a segunda condição é analisada. Lembre-se de que esse comportamento se repete até que uma condição seja atendida ou todas as condições tenham sido testadas e consideradas falsas.

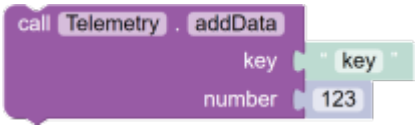


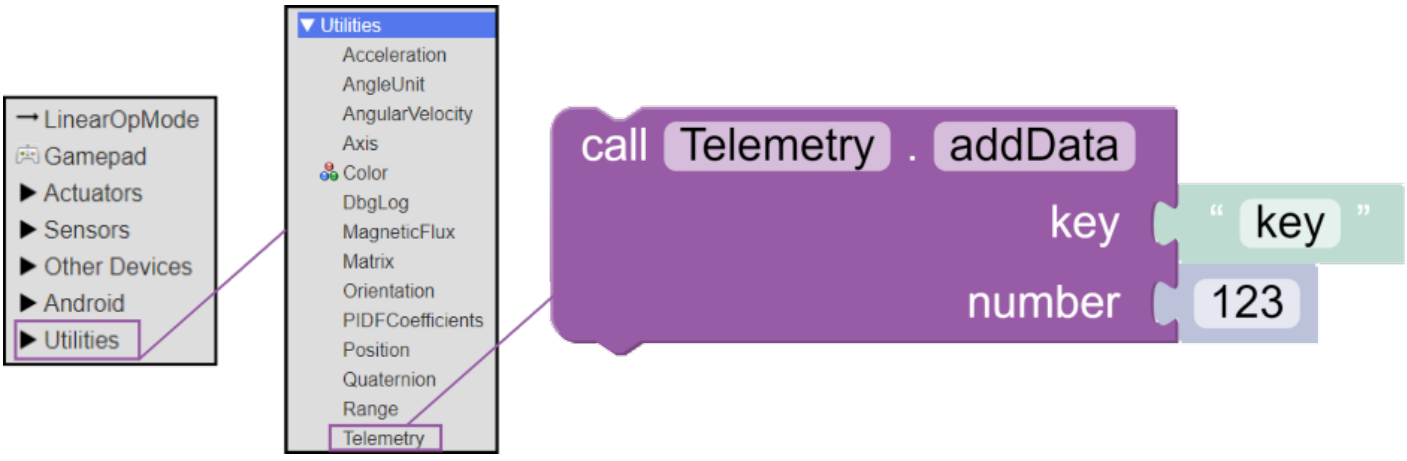
Servoss e telemetria

Telemetria é o processo de coletar e transmitir dados. Na robótica, a telemetria é usada para enviar dados internos de atuadores e sensores para a Estação do Condutor. Esses dados podem então ser analisados pelos usuários para tomar decisões que possam melhorar o código.

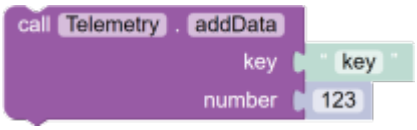
A telemetria mais útil do servo é a posição do servo ao longo de sua faixa de 270 graus. Para obter essa informação, a seguinte linha precisa ser usada.

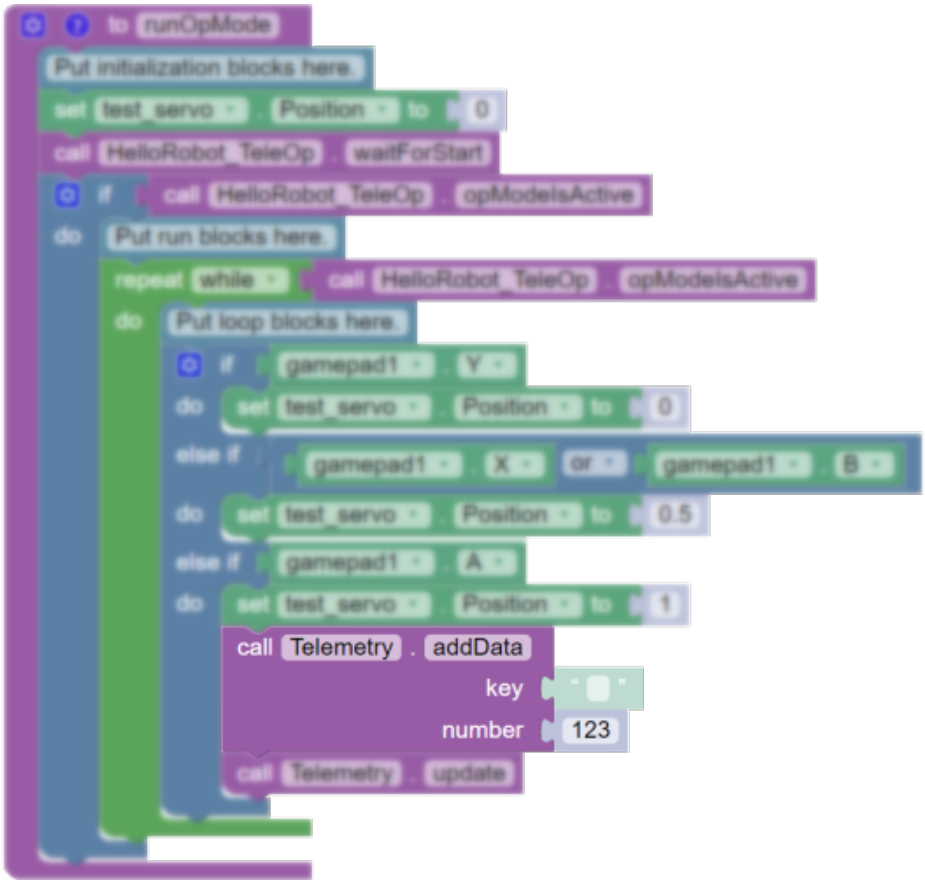
Para acessar os blocos de telemetria, selecione o menu suspenso de Utilitários. O menu de utilitários está em ordem alfabética, então a telemetria está localizada mais para o final das opções do menu suspenso. Selecione o seguinte bloco do menu de telemetria:





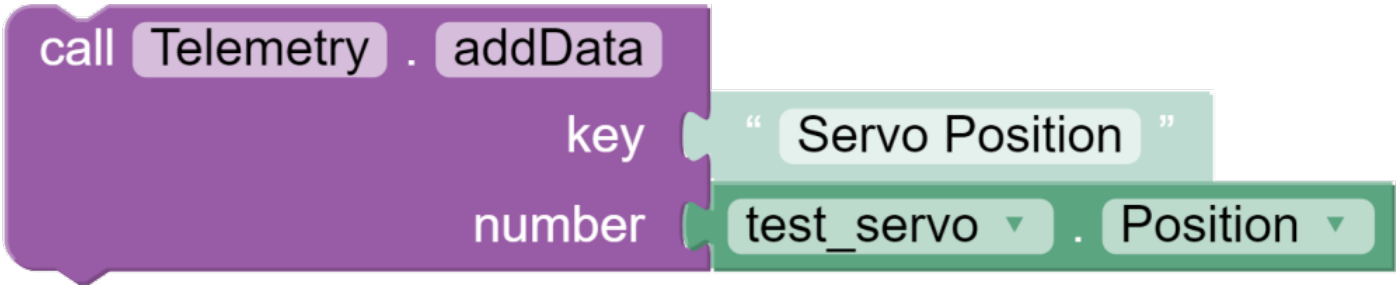
Arraste o seguinte bloco e coloque-o abaixo do bloco if/else if:





Do menu do Servo, puxe o bloco abaixo. Arraste e conecte-o ao parâmetro de número nos blocos de telemetria.

Mude o parâmtro *key* para "Servo Position"



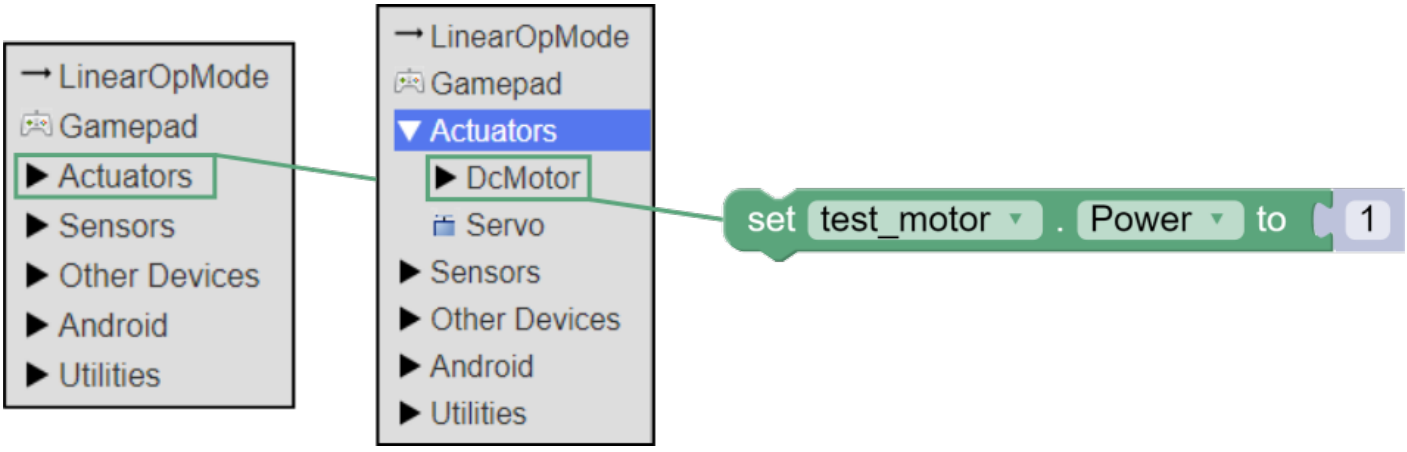
Quando o modo operacional é executado, o bloco de telemetria exibirá as informações de posição atual com a Chave de Posição do Servo. O número correspondente à posição atual mudará conforme a posição do eixo do servo se altera.

Noções básicas de motor

> Modifique o seu modo operacional para adicionar o código relacionado ao motor. Isso pode ser feito limpando suas modificações de código atuais ou adicionando o código relacionado ao motor ao seu modo operacional atual.

O objetivo desta seção é abordar alguns conceitos básicos de programação de um motor dentro do ambiente de Blocos. Ao final desta seção, os usuários deverão ser capazes de controlar um motor usando um gamepad, além de compreender alguns princípios básicos de trabalho com encoders de motor.

Dado que esta seção se concentrará em motores, é importante compreender como acessar motores dentro do ambiente de Blocos. Na parte superior da seção de Blocos Categorizados, há um menu suspenso para Atuadores. Quando o menu é selecionado, serão exibidas duas opções: DcMotor ou Servo. Selecionar DC Motor abrirá uma janela lateral preenchida com vários blocos relacionados a motores.

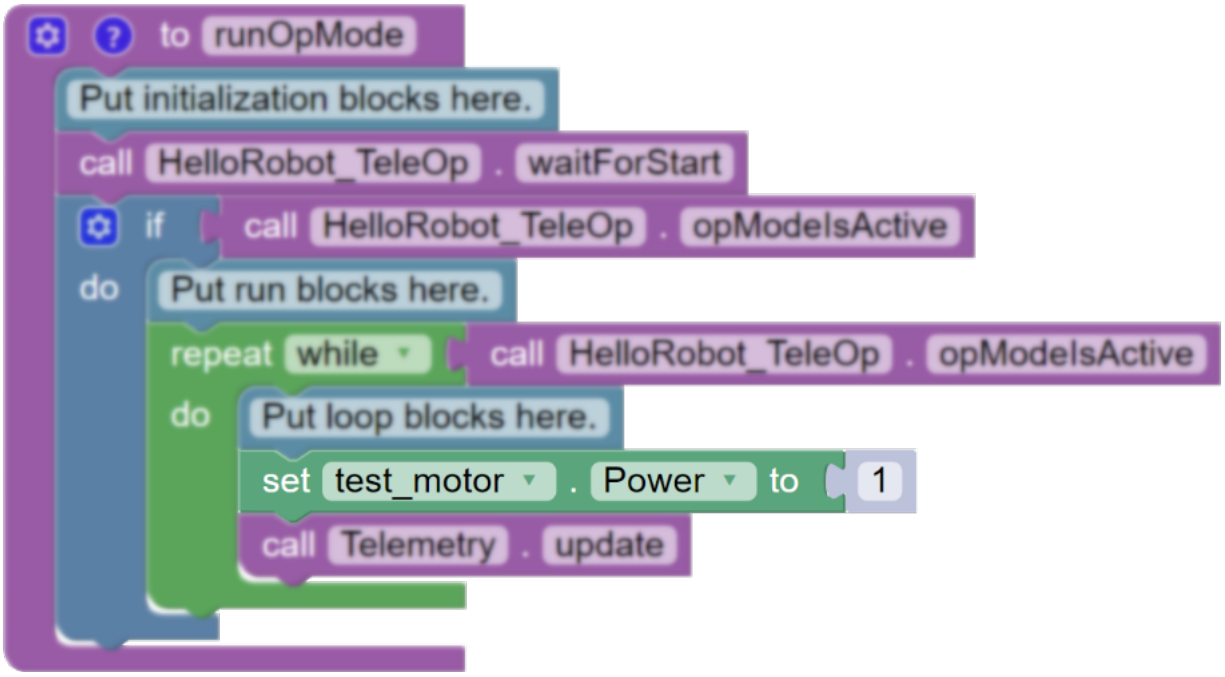


Controlando motores

No menu Dc Motor em Blocos, selecione o bloco:

> O bloco acima terá nomes diferentes dependendo do nome do motor em um arquivo de configuração. Se houver vários motores em um arquivo de configuração, a seta ao lado de "test_motor" abrirá um menu com todos os motores presentes na configuração.

Adicione este bloco ao código do modo operacional (op mode) dentro do loop while.



Selecione "Salve Op Mode" no canto superior direito na Console do Controlador do Robô.

> Tente executar este modo operacional no banco de testes e considere as seguintes perguntas:
>
> - A que velocidade o motor está funcionando?
> - O que acontece se você alterar a potência de 1 para 0.3?
> - O que acontece se você mudar a potência para -1?

O nível de potência enviado ao motor depende do número numérico atribuído ao motor. A mudança de 1 para 0,3 reduziu a velocidade do motor de 100% do ciclo de trabalho para 30% do ciclo de trabalho. Enquanto isso, a mudança para -1 permitiu que o motor girasse a 100% do ciclo de trabalho na direção oposta. Assim, a potência pode ser variada para mover um motor para frente ou para trás.

No entanto, o bloco de motor **setPower(1)** executará o motor na direção atribuída até que algo no código pare o motor ou cause uma mudança na direção.]

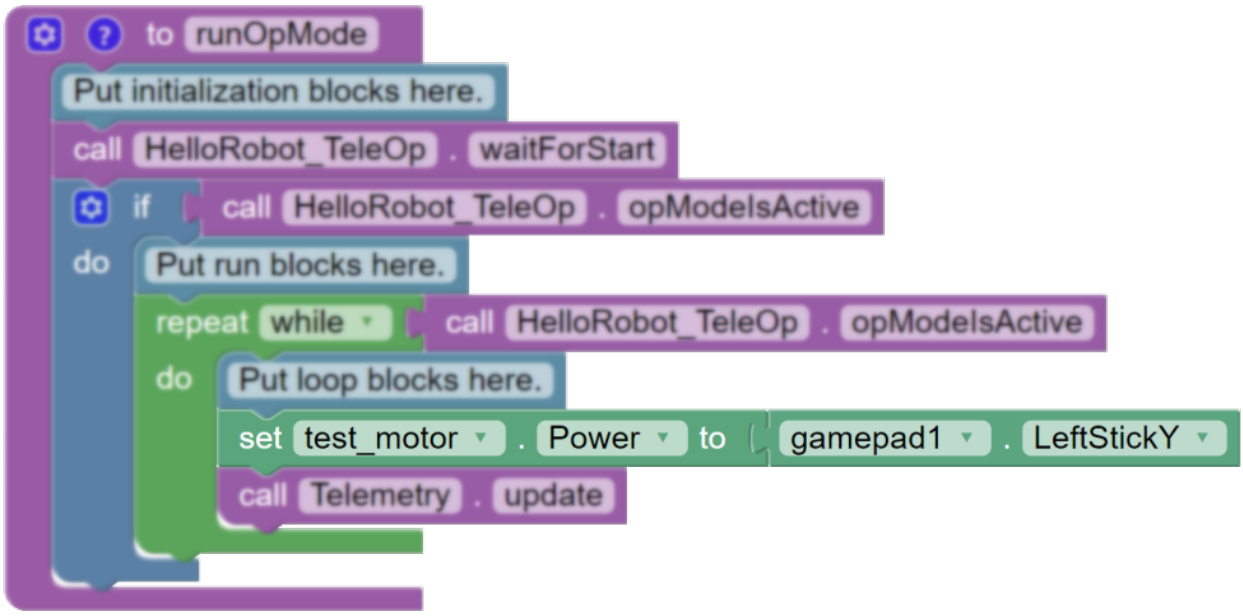
> Para compreender melhor os motores e o conceito de ciclo de trabalho, consulte a documentação sobre Motores e Escolha de Atuadores da REV Robotics

Controlando motores pelo controle

Na seção anterior, você aprendeu como configurar o motor para funcionar em um nível de potência específico em uma direção específica. No entanto, em algumas aplicações, pode ser necessário controlar o motor com um gamepad para alterar facilmente a direção ou o nível de potência de um mecanismo.

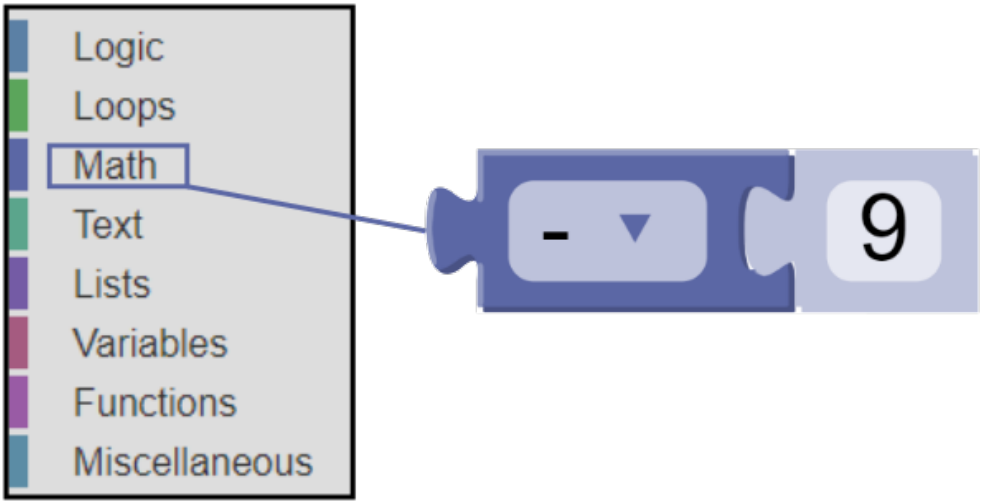
Do menu Gamepad em Blocos, selecione o bloco **leftStickY**.

Arraste o bloco leftStickY para que ele se encaixe no lado direito do bloco Power. Este conjunto de blocos irá repetidamente ler o valor do joystick esquerdo (posição y) do gamepad #1 e definir a potência do motor para o valor Y do joystick esquerdo.



Observe que, para os gamepads Logitech F310, o valor Y de um joystick varia de -1, quando o joystick está em sua posição mais alta, até +1, quando o joystick está em sua posição mais baixa. Se o motor não estiver se movendo na direção desejada, adicionar um símbolo negativo ou um operador de negação à linha de código alterará a direção do motor em relação ao gamepad.

Da seção Matemática no Blocks, selecione o bloco da imagem abaixo.



Arraste o bloco do símbolo de negativo para que ele se encaixe entre os blocos de **Power** e **leftStickY**, como na imagem abaixo:

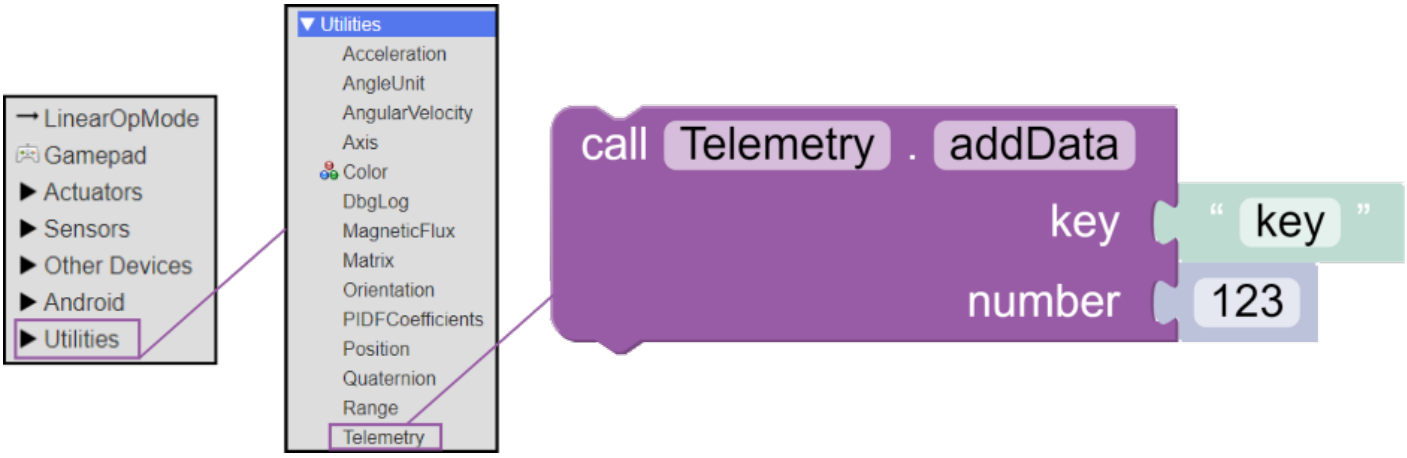
Motores e telemetria

Lembre-se de que telemetria é o processo de coletar e transmitir dados. Na robótica, a telemetria é usada para enviar dados internos de atuadores e sensores para a Estação do Motorista. Esses

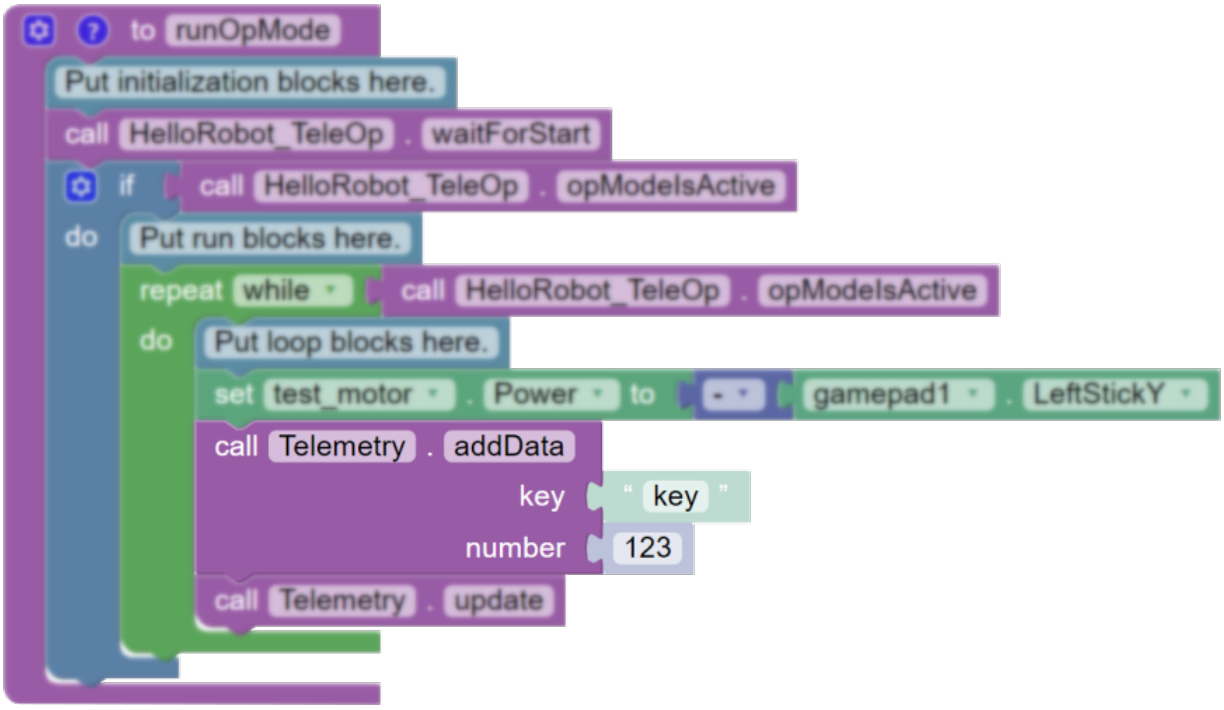
dados podem ser analisados pelos usuários para tomar decisões que melhorem o código.

Para obter dados de telemetria do motor, é necessário usar encoders de motor. Os motores DC REV, como o Core Hex Motor, são equipados com encoders internos que transmitem informações na forma de contagens.

Para acessar os blocos de telemetria, selecione o menu Utilitários. O menu Utilitários está em ordem alfabética, então a telemetria está localizada na parte inferior das opções do menu. Selecione o bloco abaixo e faça o seguinte:

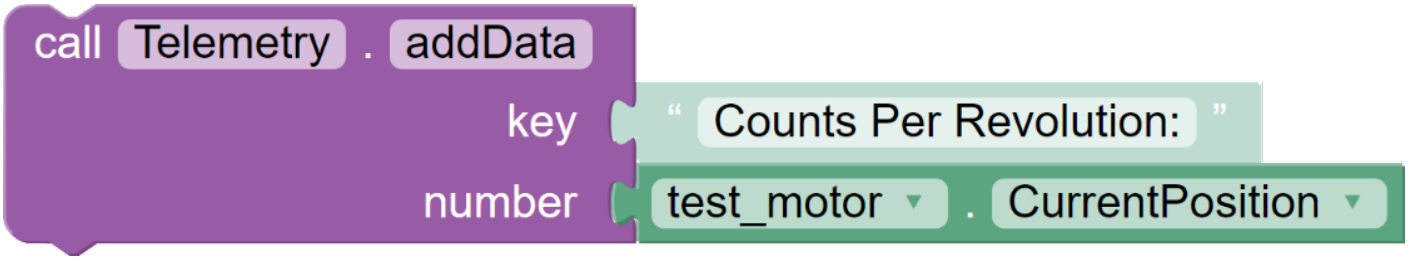


Arraste esse bloco para baixo do bloco anteriormente feito, como mostra a seguinte imagem:



Do menu DC Motor, pegue o bloco "CurrentPosition". Arraste o bloco e conecte-o ao parâmetro numérico nos blocos de telemetria.

Mude o parâmetro *key* para "Counts per Revolution"



Quando o modo operacional é executado, o bloco de telemetria exibirá as informações de posição atual com a chave "Counts Per Revolution". O número correspondente à posição atual mudará à medida que a posição do eixo do motor for alterada.

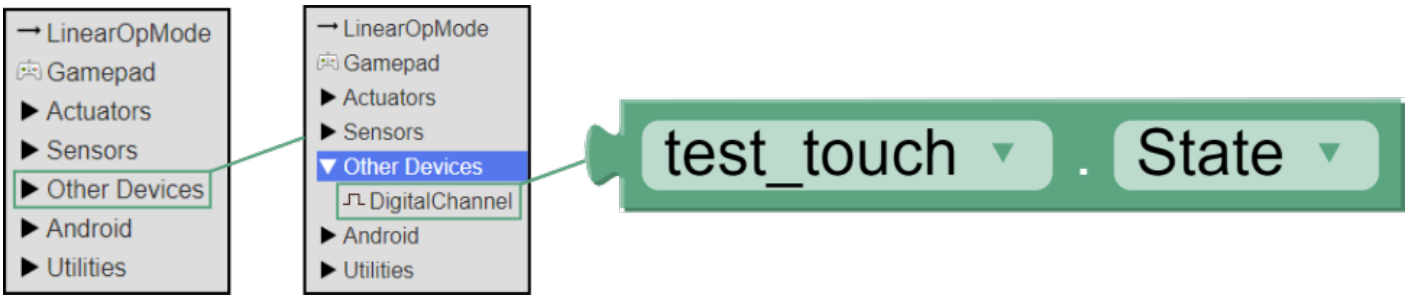
> Para obter mais informações sobre a programação de encoders, consulte a página "Using Encoders" (Usando Encoders). Para mais informações sobre a métrica de contagens por revolução e como utilizá-la, confira a página "Encoders".

Programando um sensor

Noções básicas de sensor

> Modifique o seu modo operacional para adicionar o código relacionado ao dispositivo digital. Isso pode ser feito limpando as modificações atuais do seu código ou adicionando o código do dispositivo digital ao seu modo operacional.

O objetivo desta seção é abordar alguns dos conceitos básicos de programação de um dispositivo digital, ou Sensor de Toque, dentro do Blocks. Como esta seção se concentrará em dispositivos digitais, é importante entender como acessar blocos específicos para esses dispositivos. No topo da seção "Categorize Blocks", há um menu suspenso para "Other Devices". Quando o menu é selecionado, ele exibirá uma opção para "Digital Devices". Selecionar "Digital Devices" abrirá uma janela lateral preenchida com vários blocos relacionados a dispositivos digitais. O bloco mais frequentemente usado é o de **State**

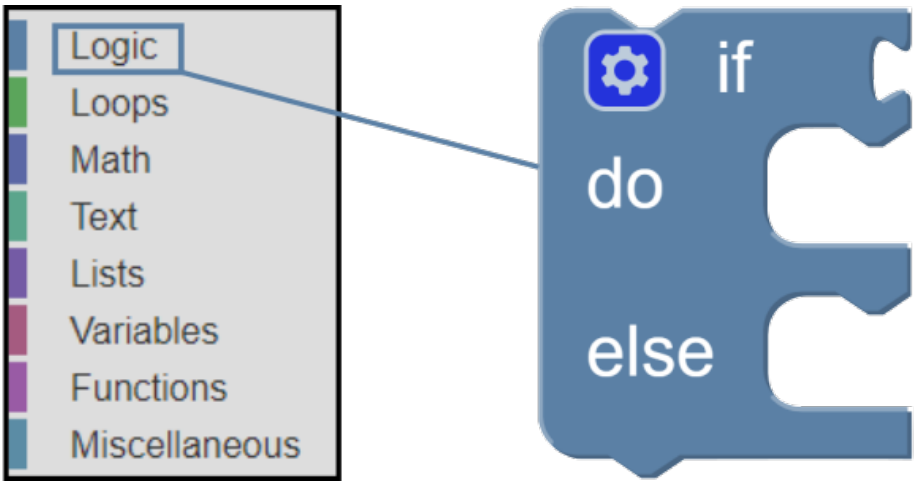


> Antes de programar com um Sensor de Toque ou outro dispositivo digital, é importante entender o que é um dispositivo digital e quais são as aplicações comuns para esses dispositivos. Visite a página de Sensores Digitais para obter mais informações.

Programando um dispositivo digital

As informações provenientes de dispositivos digitais vêm em dois estados, também conhecidos como estados binários. A maneira mais comum de utilizar essas informações é por meio de uma instrução condicional, como uma declaração if/else.

Do Menu Lógico em Blocos, selecione o bloco if/else.



Arraste esse bloco e coloque-o abaixo do bloco de comentário **Put run blocks here**

Selecione um bloco de **State** no menu de Dispositivos Digitais e adicione-o ao bloco if/do/else, conforme mostrado na imagem abaixo.

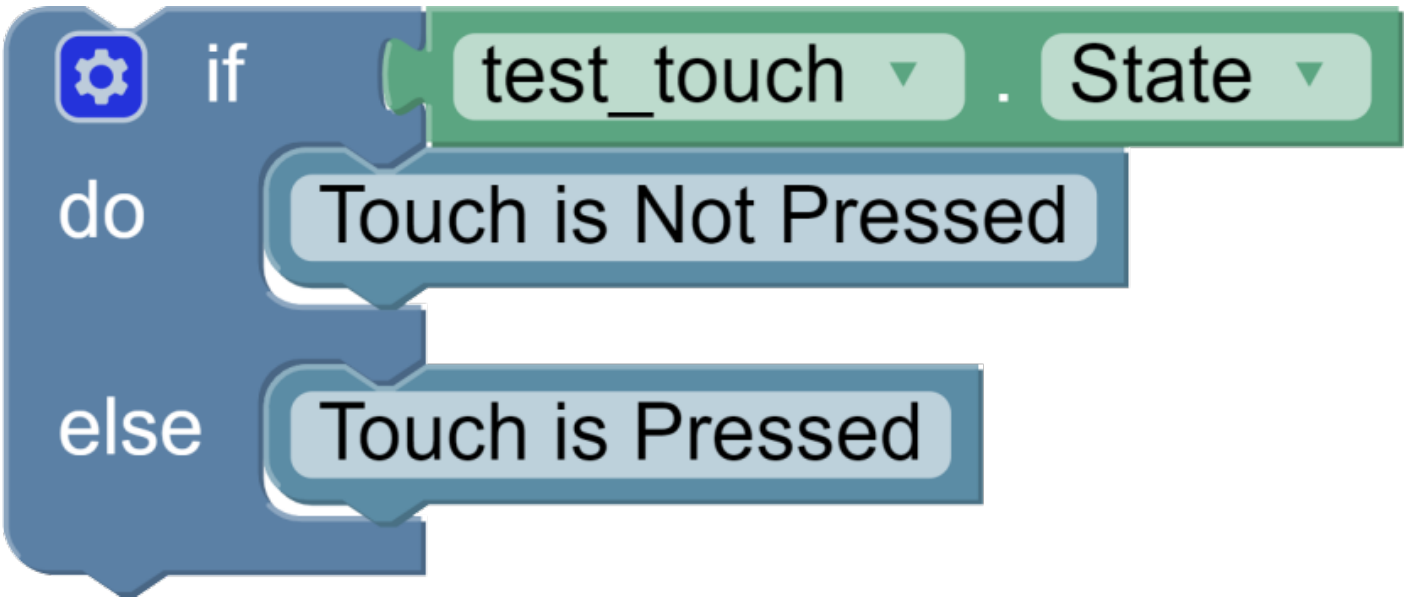
O bloco de State armazena a informação binária FALSO/VERDADEIRO do sensor de toque e atua como a condição para o bloco if/else. Se o State for verdadeiro, qualquer código colocado na parte de "fazer" do bloco será ativado. Se o State for falso, qualquer coisa colocada na parte de "senão" do bloco será ativada.

O estado FALSO/VERDADEIRO de um Sensor de Toque REV corresponde a se o botão no Sensor de Toque está pressionado ou não. Quando o botão não está pressionado, o estado do Sensor de Toque é Verdadeiro. Quando o botão é pressionado, o estado do Sensor de Toque é Falso.

Para ajudar a lembrar como os estados físicos e digitais do sensor correspondem nas próximas seções, vamos usar alguns comentários.

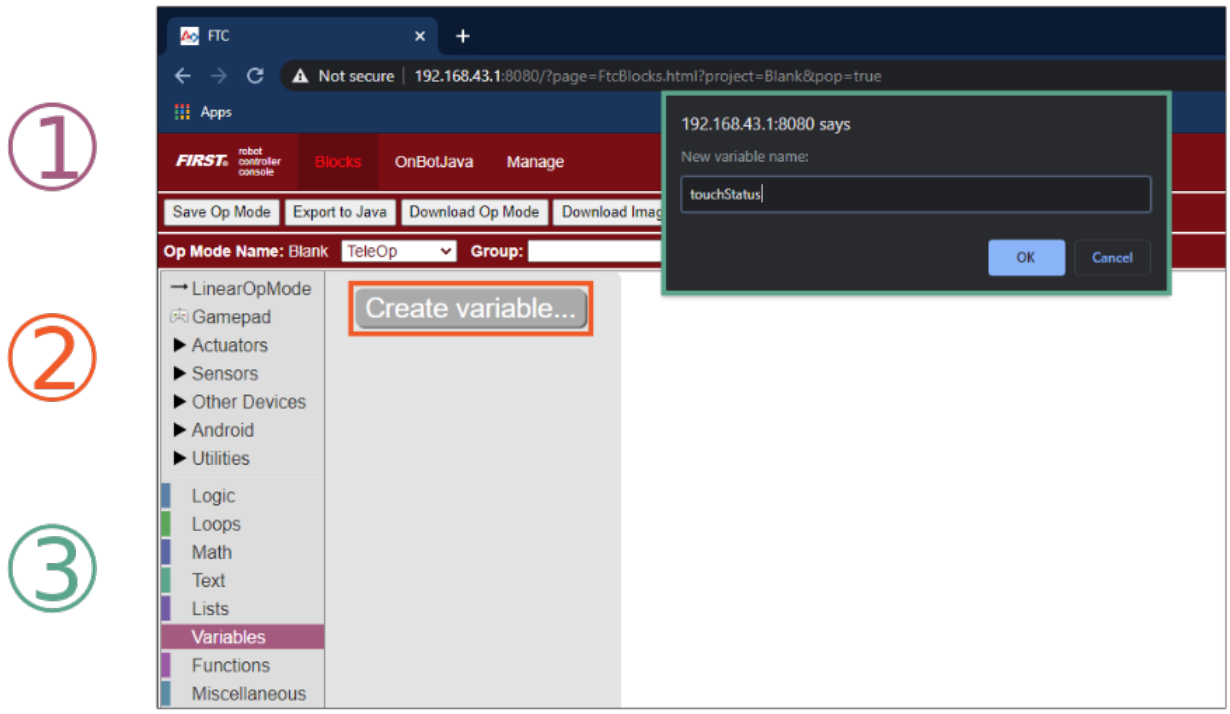
Blocos de comentário podem ser encontrados no menu Diversos.

Após achar esse bloco, monte um sistema conforme a imagem abaixo:



O próximo passo no processo é usar a telemetria para exibir o status do Sensor de Toque no dispositivo da Estação do Motorista. Para fazer isso, vamos criar uma variável de string chamada touchStatus.

> Uma String é uma sequÇecia de caracteres

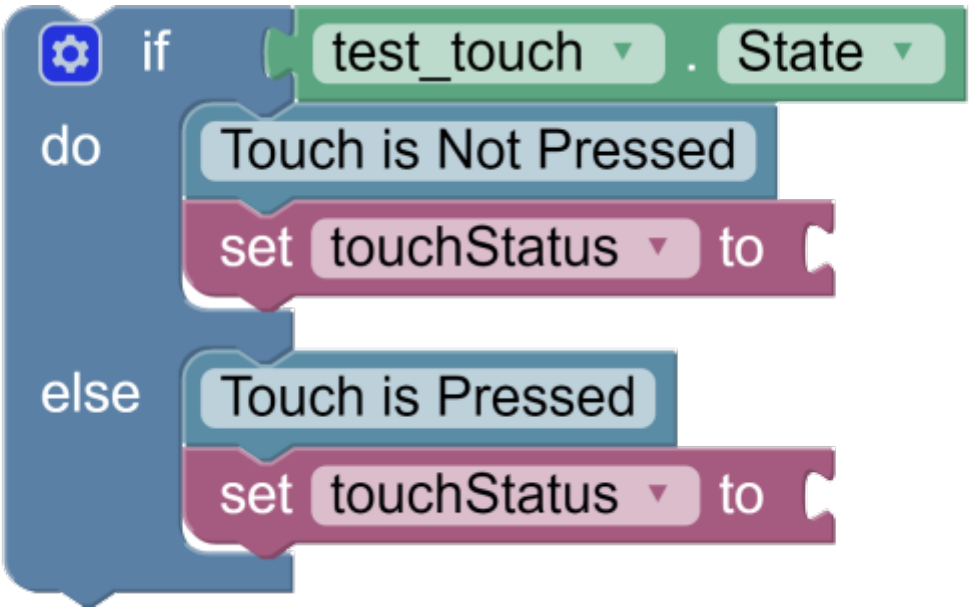


1. Clique no menu *Variables*. Isso abrirá uma janela lateral.
2. Selecione o bloco *Create variable*...
3. Um prompt do Controlador de Robôs da FIRST aparecerá pedindo um nome para a variável. Nomeie a variável como touchStatus. Clique em OK.

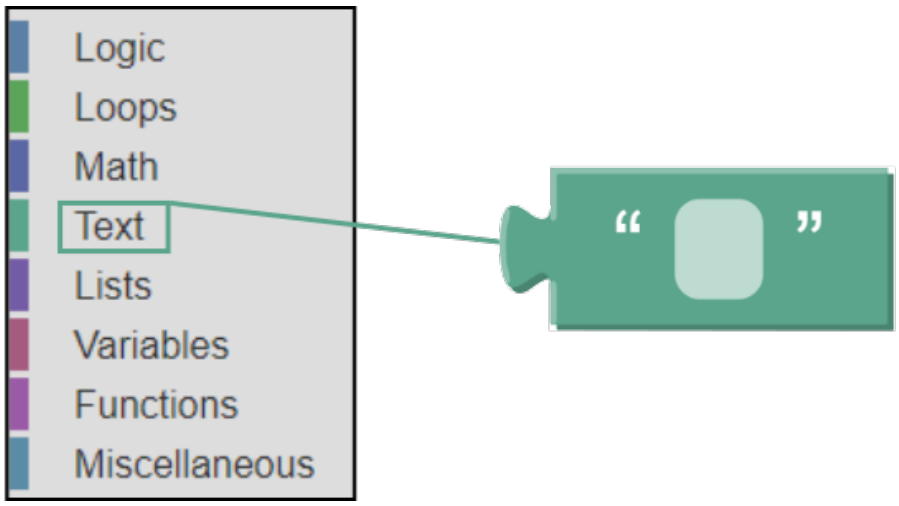
Esse processo criou uma variável chamada touchStatus. Atualmente, touchStatus está indefinido; para defini-lo, o bloco precisa ser usado. Este bloco pode ser encontrado no menu Variáveis, agora que a variável foi criada.

Arraste esse bloco e coloque-o abaixo do comentário *Touch is not pressed*.

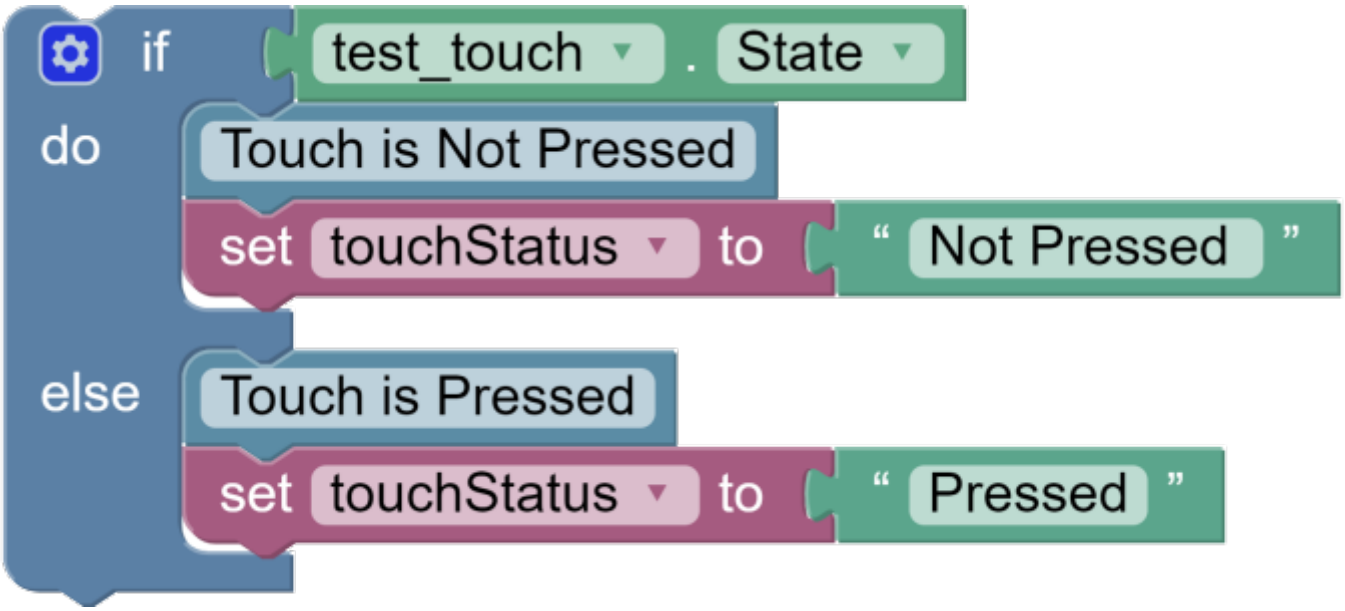
Adicione outro abaixo do *Touch is pressed*.



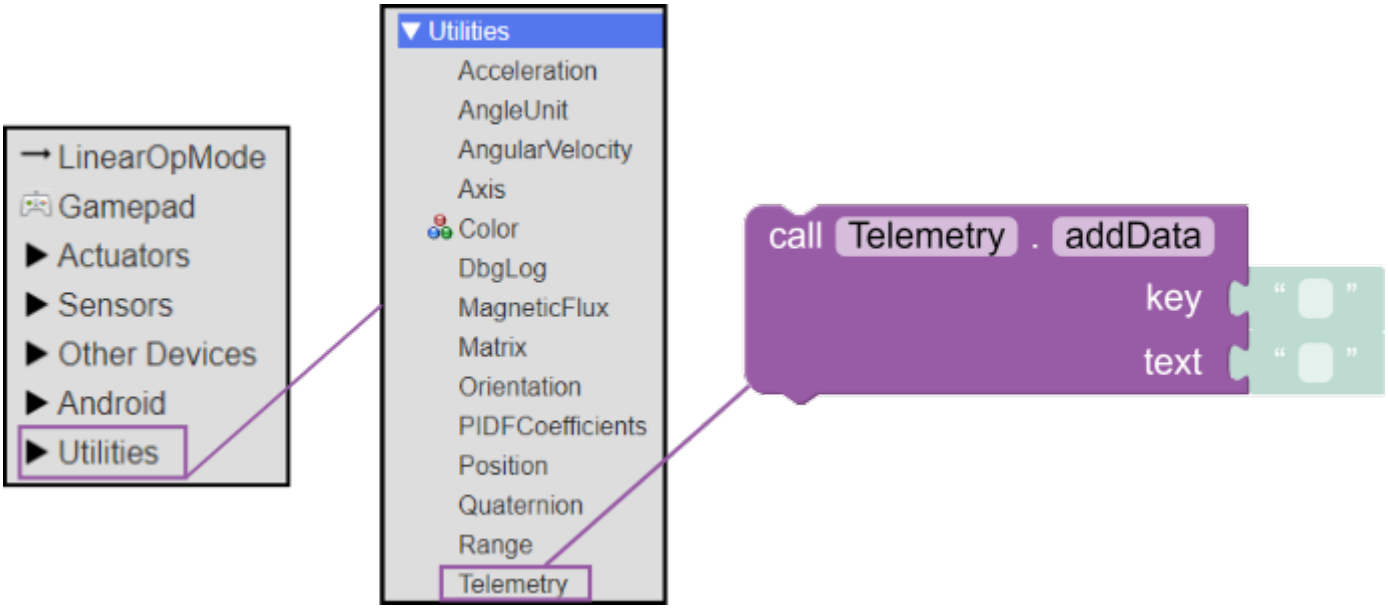
O bloco touchStatus permite que você altere o valor da variável. Dependendo de qual estado ele estiver, touchStatus será definido para outra String. Para isso selecione o bloco de String do menu Text. Como a imagem abaixo mostra:



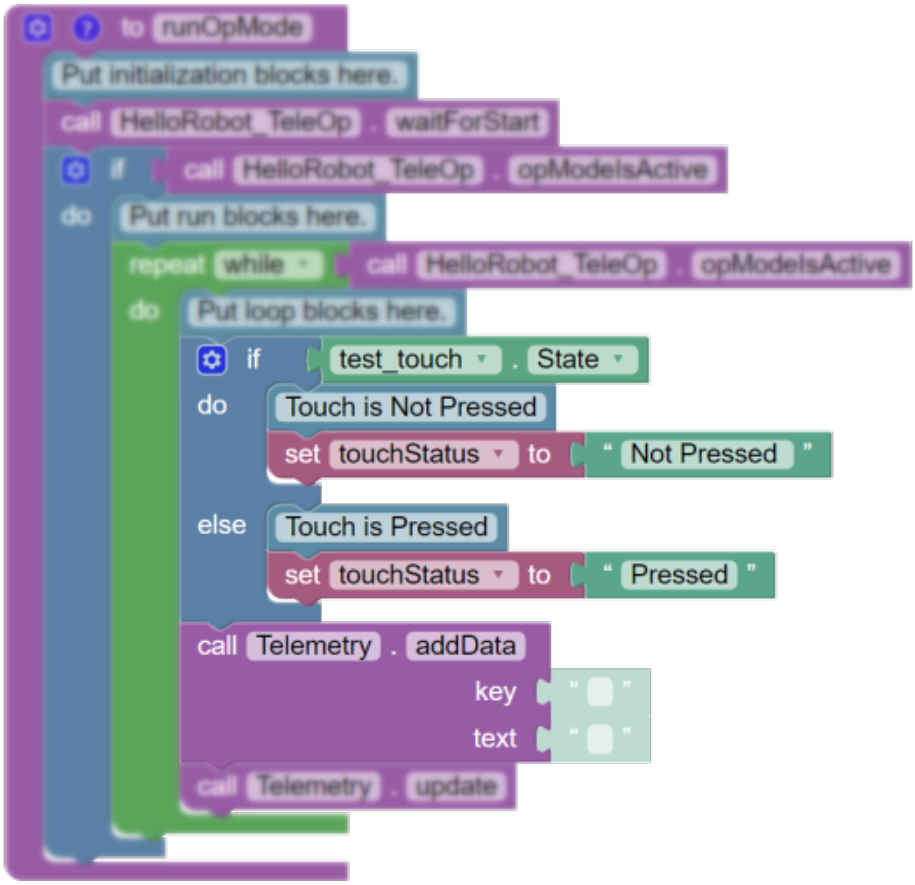
Adicione um bloco de string ao *touchStatus*. Preencha o bloco com uma mensagem de status que relate o estado do sensor. Como "*Not pressed*" e "*Pressed*".



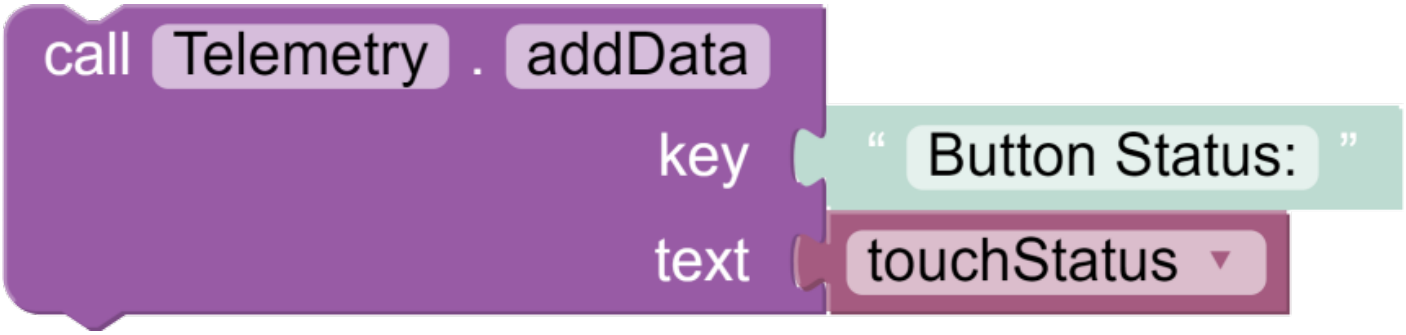
Para exibir essas informações na Driver Station, é necessário utilizar a telemetria. Para acessar os blocos de telemetria, selecione o menu suspenso de Utilitários. O menu de utilitários está em ordem alfabética, então a telemetria está localizada mais para o final das opções do menu suspenso. Selecione o bloco abaixo:



Arraste esse bloco e coloque-o abaixo do bloco if/else.



Arraste o bloco de *touchStatus* para a área de texto da função *addData* e mude o parâmetro *key* para "Button"

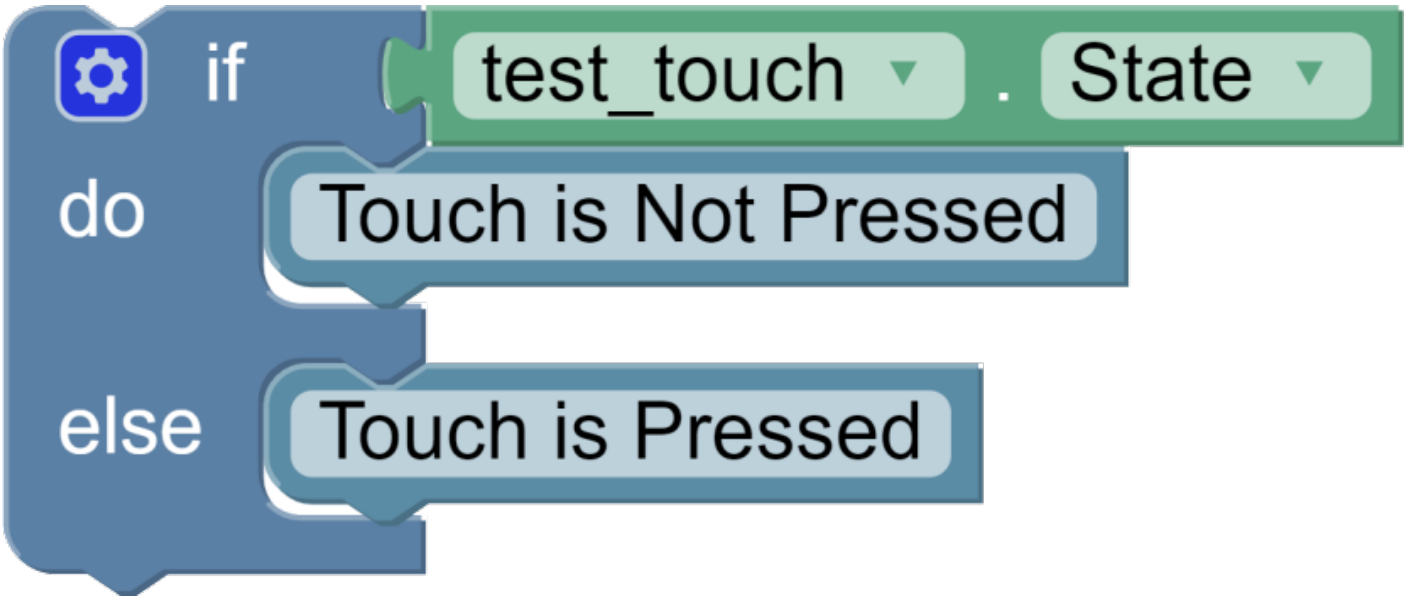


Enquanto esse programa estiver operando o estado do botão será mostrado na telemetria.

Dispositivos digitais como um interruptor de limite

Um dos usos mais comuns para um dispositivo digital, como um sensor de toque, é utilizá-lo como um interruptor de limite. A intenção de um interruptor de limite é interromper o funcionamento de um mecanismo, como um braço ou elevador, antes que ultrapasse suas limitações físicas. Nessa aplicação, a energia precisa ser cortada do motor quando o limite é atingido.

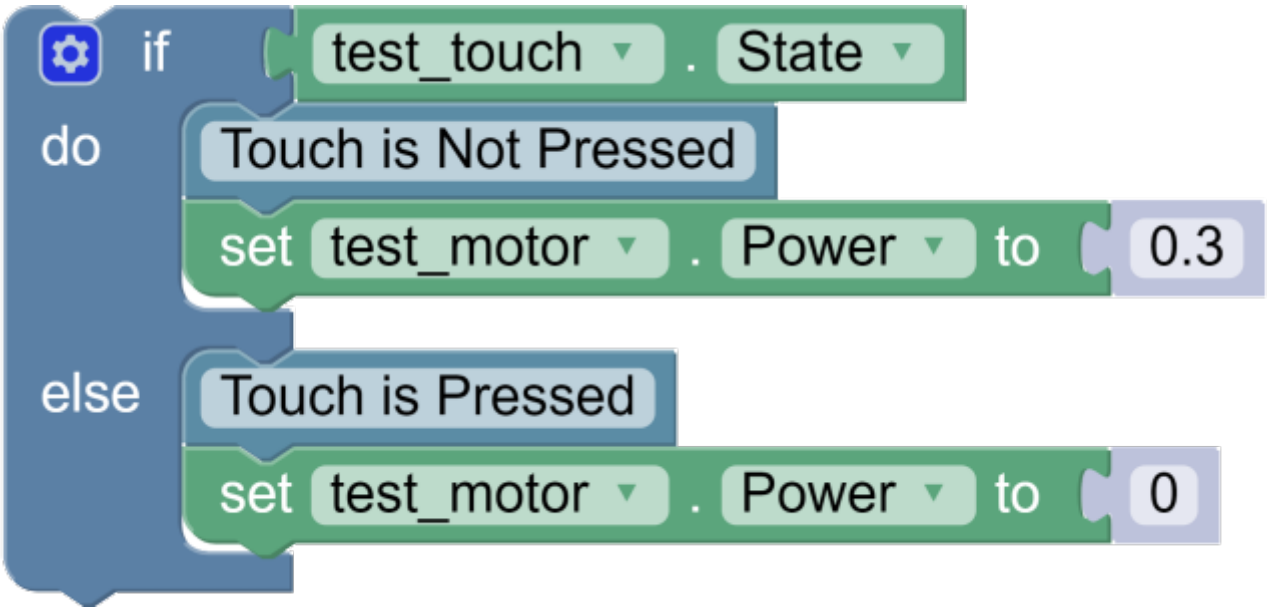
O conceito de um interruptor de limite envolve muitas etapas semelhantes à seção anterior sobre a programação de um dispositivo digital. Por essa razão, vamos retomar a partir do conjunto de blocos a seguir:



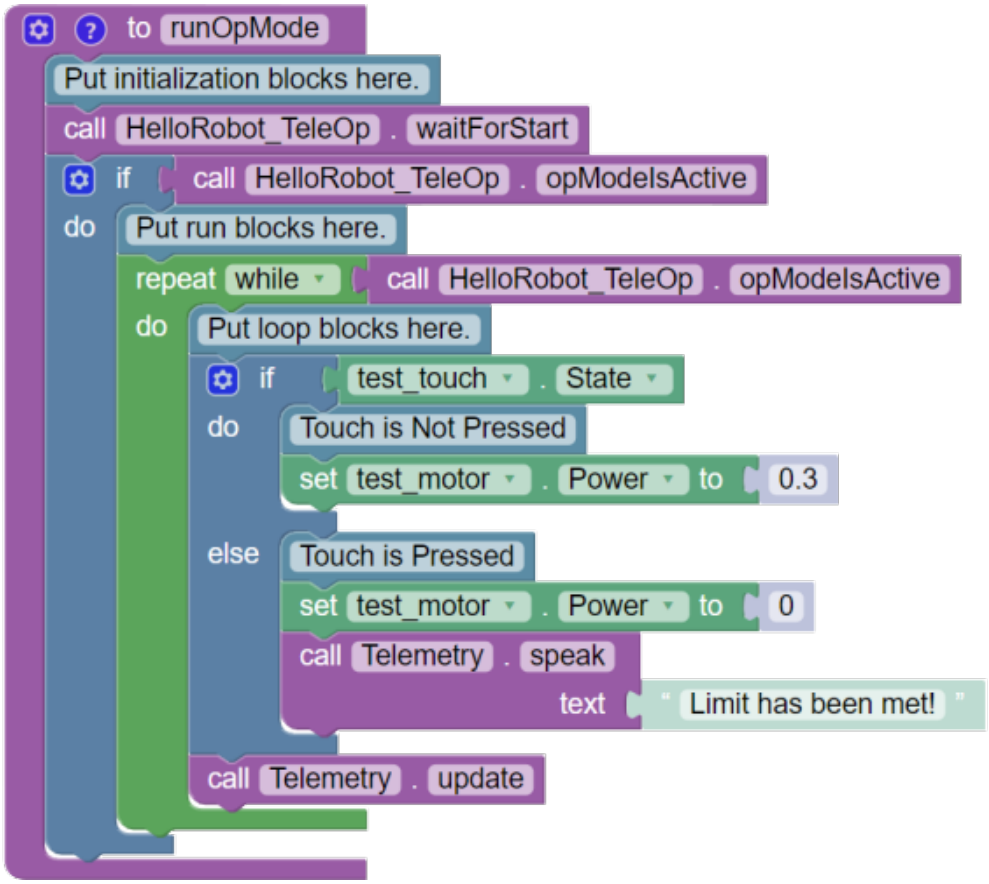
O bloco if/else estabelece um ambiente condicional para o interruptor de limite. Se o sensor de toque não estiver pressionado, o motor pode funcionar; no entanto, se estiver pressionado, o motor não pode funcionar. Para adicionar isso ao código, o bloco Power precisa ser utilizado.

> Para obter informações sobre onde encontrar blocos específicos para motores, por favor, reveja a seção sobre motores.

Adicione um bloco de Power abaixo do comentário "Touch is not Pressed". Altere o valor de Power para 0.3. Adicione outro bloco de Power abaixo do comentário "Touch is Pressed". Altere o valor de Power para 0.



Este bloco if/else introduz os conceitos básicos de um interruptor de limite. Como na maioria dos sensores, é bom ter telemetria que atualiza a Estação do Motorista sobre o status do sensor. Considere o código da seção anterior ou o seguinte código como ideias potenciais para telemetria.



Revisão #4
Criado 18 dezembro 2023 11:57:57 por Enzo Coutinho
Atualizado 18 dezembro 2023 16:15:48 por Enzo Coutinho