ソフトウェア：DR-Visual Logic HOVIS

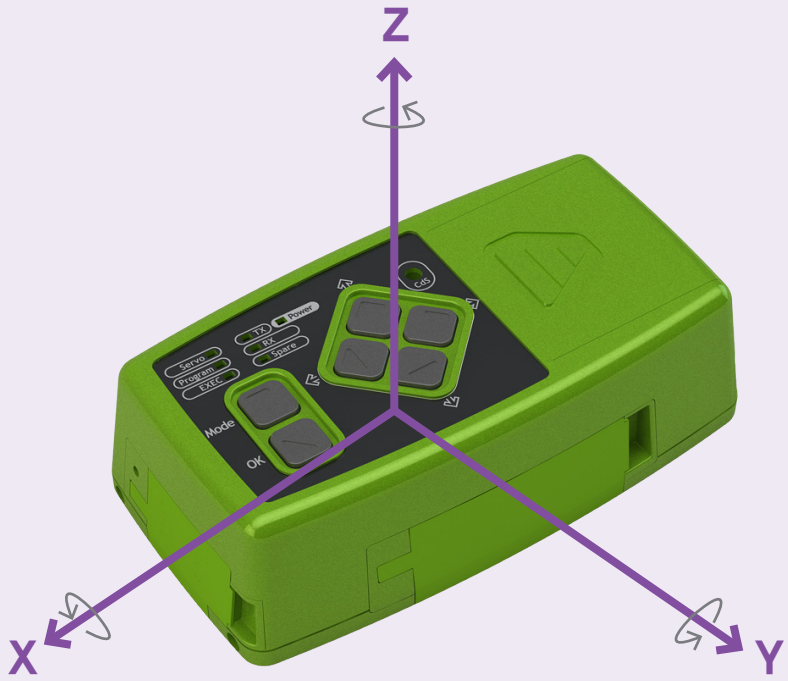
モジュール別プログラミング：センサー（Sensor）> Dynamics Sensor

Acceleration Sensorの例

例の説明

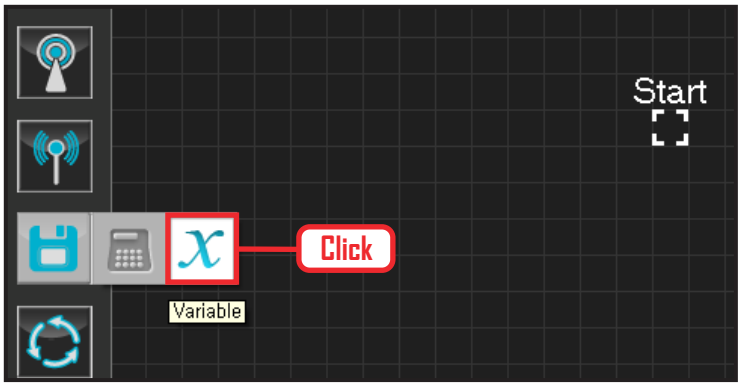
Acceleration（加速度）センサを用いてロボットが前に倒れた時と後ろに倒れた時を検出し立ち上がるプログラミングです。

Accelerationセンサはコントローラーの背面カバーを開けて入れられるモジュール形で、Gyroセンサモジュールと結合しています。



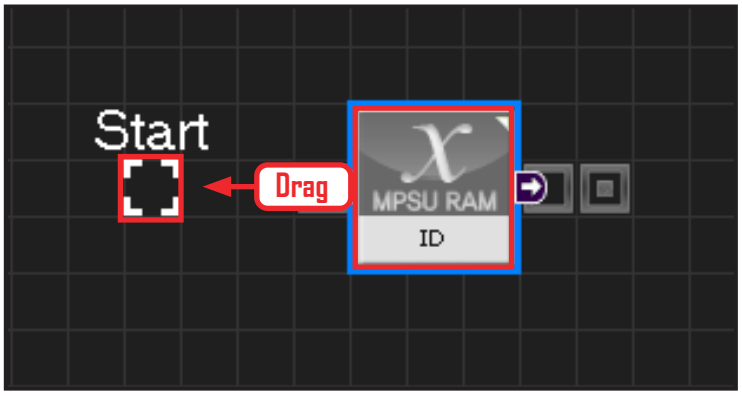
- ロボットが前に倒れた時、Z軸は“+”の加速度で、その値は約+256です。
- ロボットが後ろに倒れた時、Z軸は“−”の加速度で、その値は約−256です。

（256は約1gの重力の値を表します。）



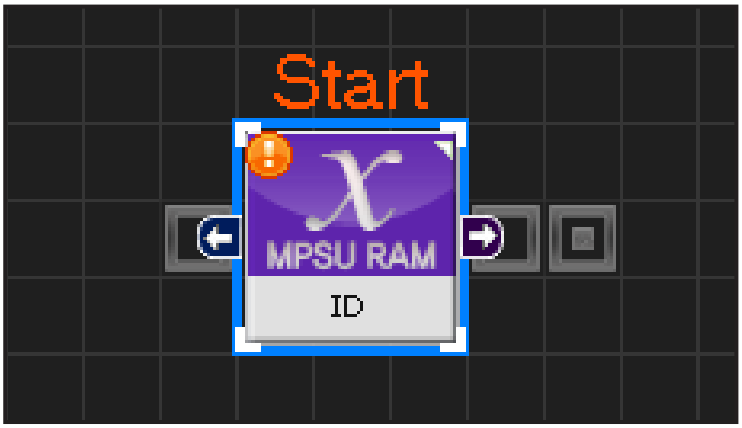
01 変数の指定

ロボットを動作させるということはロボットのサーボモーターを動作させるということです。サーボが自ら動くことができる状態の値を指定しなければなりません。
Data > Variableモジュールをクリックします。



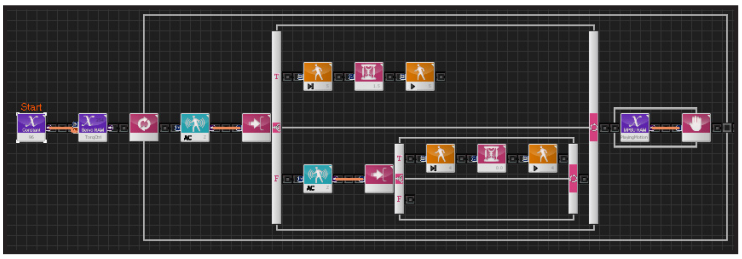
02 開始

モジュールの左側の連結線をStartPointにドラッグして正確にドッキングさせます。



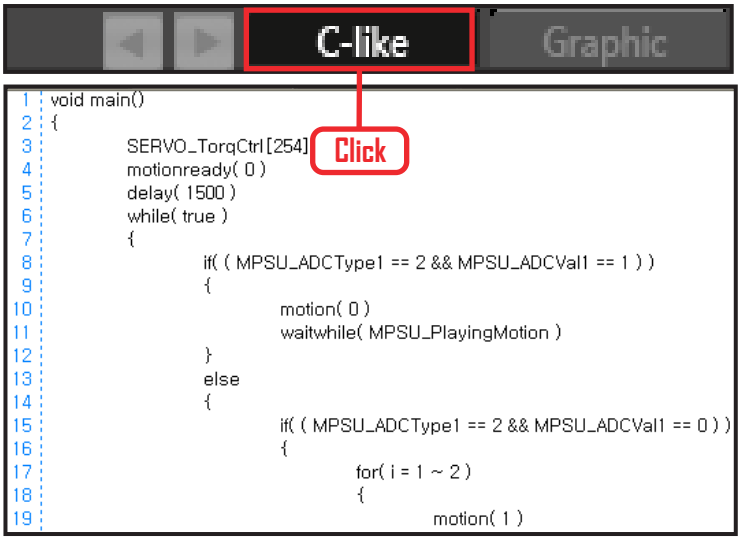
03 プログラミングの開始

モジュールとStart Pointが正確にドッキングすると、左側のように有効化されたカラーイメージのモジュールに変わります。
これはプログラミングが開始されたという意味です。



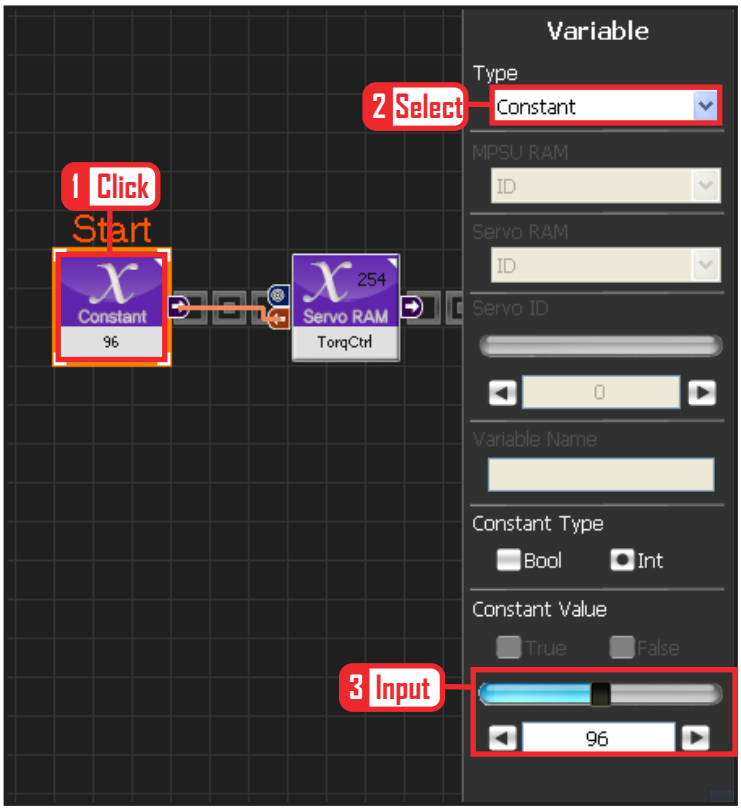
04 完全プログラミング

加速度センサを利用して倒れたロボットを立ち上げる完全なプログラムです。



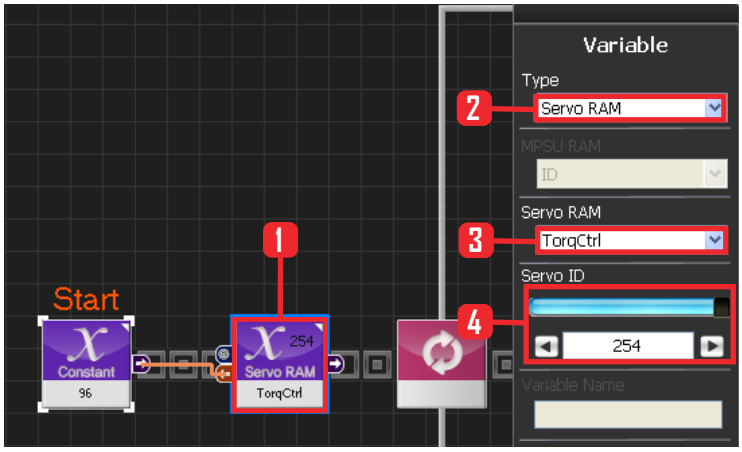
05 C-Like

右上のGraphicタブで、C-likeタブをクリックすると、左側のようなTaskプログラミングの画面が表示されます。
加速度センサを利用して完全なプログラミングをする画面です。
Cと似ている文法構造を持っているのでC文法の先行学習の効果もあります。
各モジュールごとにクリックすると、カーソルが今のプログラムの位置を示しながらモジュールごとにTextにどのように変換するかを確認することができます。



06 定数の設定

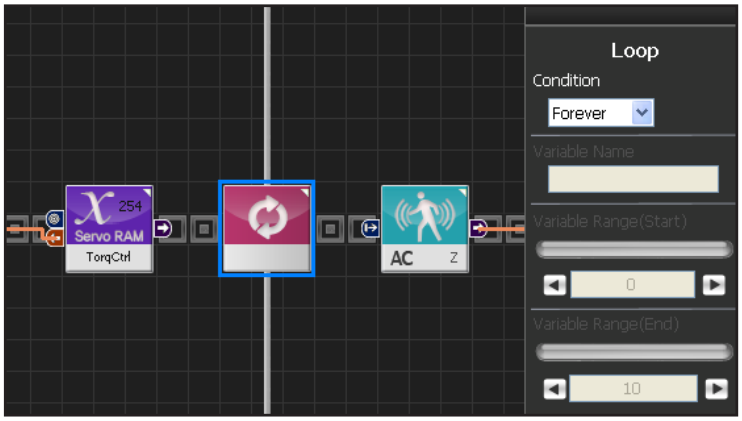
サーボモーターを自分で動かすことができる状態にする方法です。
Variable TypeでConstantを選択します。
プロパティー中のConstant　Value値を96に設定します。サーボのTorqControlレジスタに96(0x60)という値が入ると、サーボを動かすことができる状態になります。
その値はOutputコネクタを通して後ろのモジュールのトルク値に渡されます。



07 すべてのサーボに適用

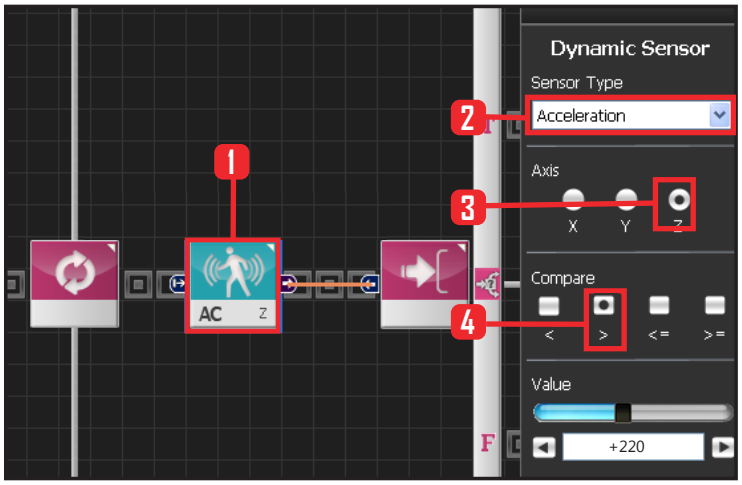
前にもらった定数値96をすべてのサーボに適用する方法です。

Variable > Type：Servo RAMを選択します。
Servo RAM：TorqCtrlを選択します。
Servo ID：254を選択します。 254は接続されているすべてのサーボに適用するという意味です。



08 反復

Forever無限反復を設定します。

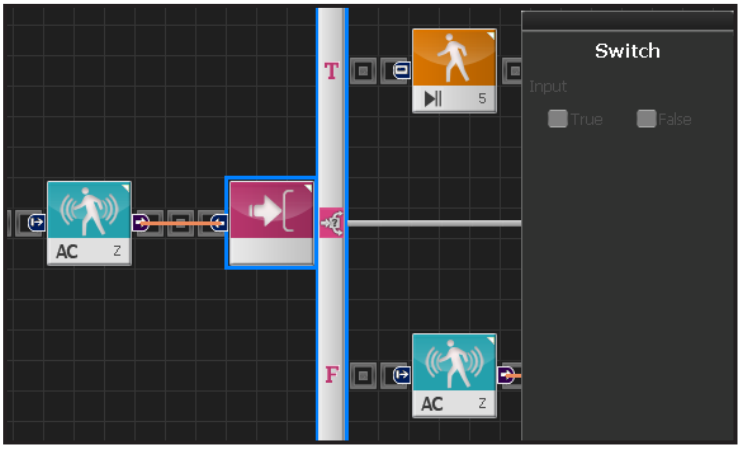


09 加速度設定（前に倒れた時）

ロボットが立っているときの加速度は0です。
ロボットが前に倒れたときの加速度は+256、後ろに倒れたときの加速度は-256です。
したがって+256の値に近いときはロボットが前に倒れたと判断します。その基準値を+220に設定します。

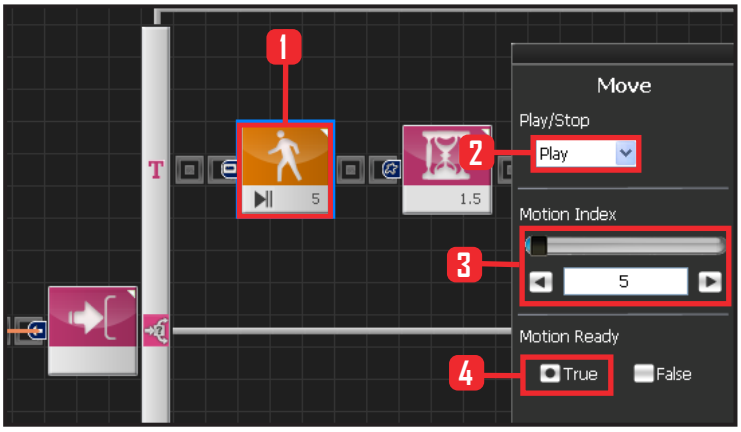
+220より小さい値の場合、ロボットが前に倒れたと判断します。

Sensor > Dynamic Sensorモジュールを選択します。
Sensor　Type：Accelerationを選択します。加速度センサです。
Axis：Z軸に設定します。
Compass：>に設定します。
Value：+220に設定します。



10 If 分岐文

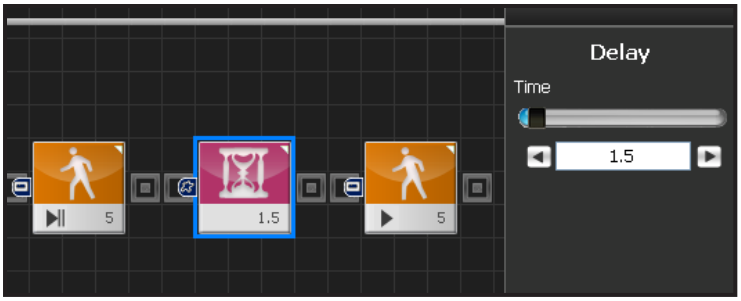
Trueの時は後ろに立ち上がるモーションを実行します。Falseの時は次の条件文に移動するプログラミングです。



11 後ろに立ち上がるモーション

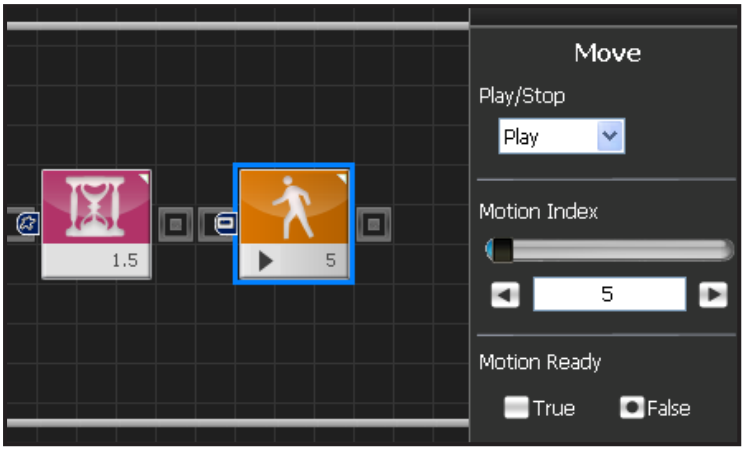
ロボットが前に倒れたときにはロボットを後ろに立ち上がらせる動作を挿入する必要があります。
5番モーションが後ろに立ち上がるモーションです。

Motion > Moveモジュールを選択します。
Play/ Stop:Playを選択します。
Motion Index:5番を選択します。 5番は後ろに立ち上がるモーションです。
Motion Ready:Trueを選択します。
モーションの準備過程です。



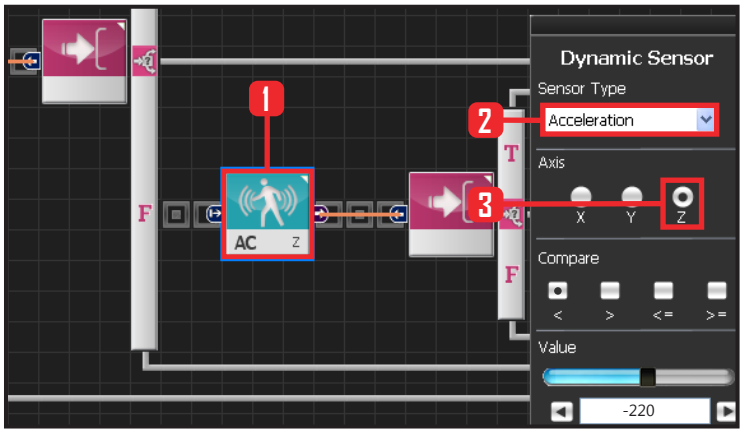
12 Delay

Motion Ready動作が終了する前にモーションが進行することを防止するためにDelay値を1.5秒に設定します。



13 後ろに立ち上がるモーションの実行

Motion　ReadyでFalseを選択すると、後ろに立ち上がるモーションが実行されます。

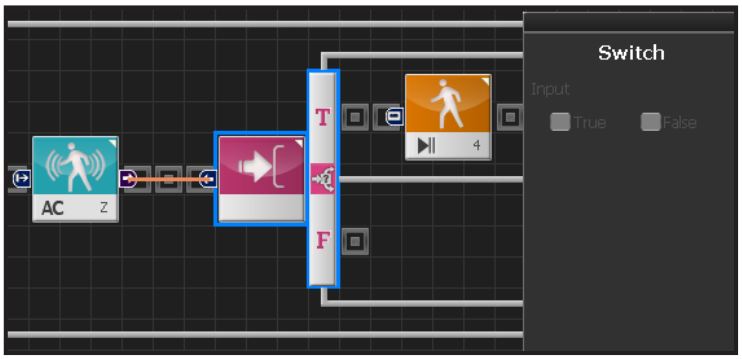


14 加速度設定（後ろに倒れた時）

ロボットが立っているときの加速度は0です。
ロボットが前に倒れたときの加速度は+256、後ろに倒れたときの加速度は-256です。
したがって-256の値に近いときはロボットが後ろに倒れたと判断します。その基準値を-220に設定します。

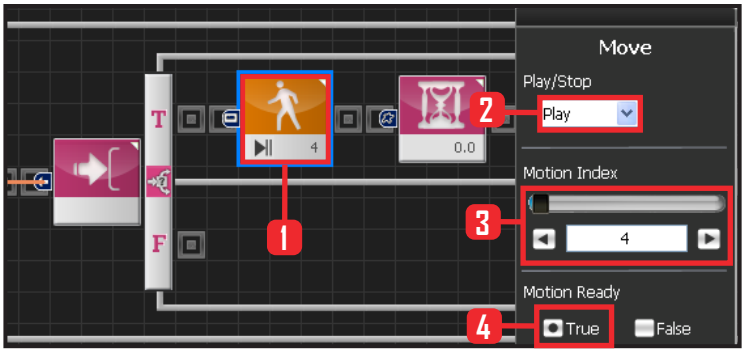
-220より大きい値の場合、ロボットが後ろに倒れたと判断します。

Sensor＞Dynamic Sensorモジュールを選択します。
Sensor　Type：Accelerationを選択します。加速度センサです。
Axis：Z軸に設定します。
Compass：＜に設定します。
Value：-220に設定します。



15 If 分岐文

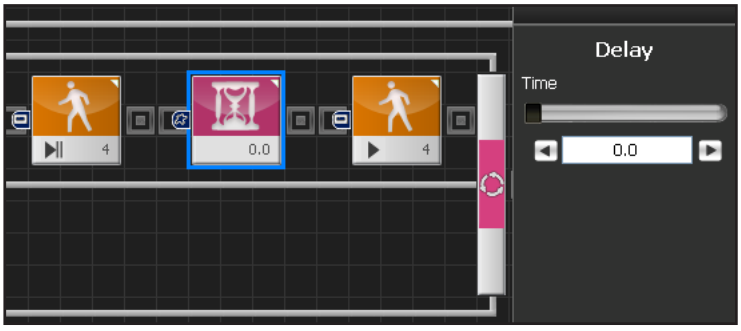
Trueの時は前に立ち上がるモーションを実行するプログラミングです。



16 モーション動作の準備

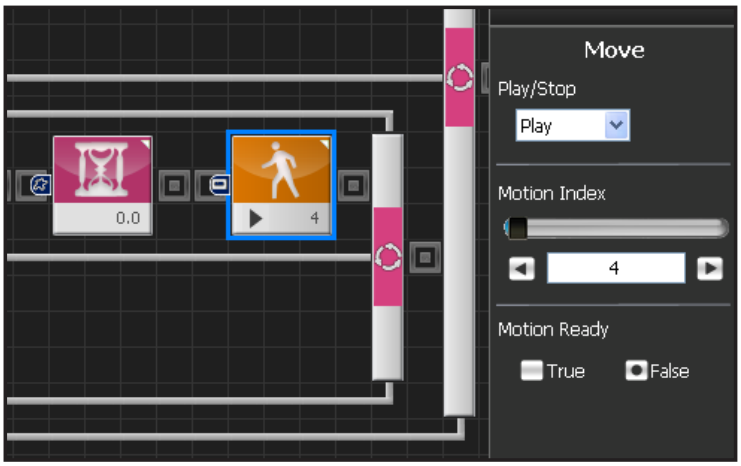
モーション動作のための準備過程です。ロボットのモーションが急に変動した場合、ロボットに無理がかかります。したがって、現在実行しようとするロボットの最初のモーションからゆっくりと移動させます。
Motion ReadyがTrueの場合、モーションの最初のシーンを準備し、Falseの場合、モーションが動作します。

Motion > Moveモジュールを選択します。
Play/ Stop:Playを選択します。
Motion Index:0番を選択します。 0番は前進しながら歩くモーションです。
Motion Ready:Trueを選択します。
モーションを準備する過程です。



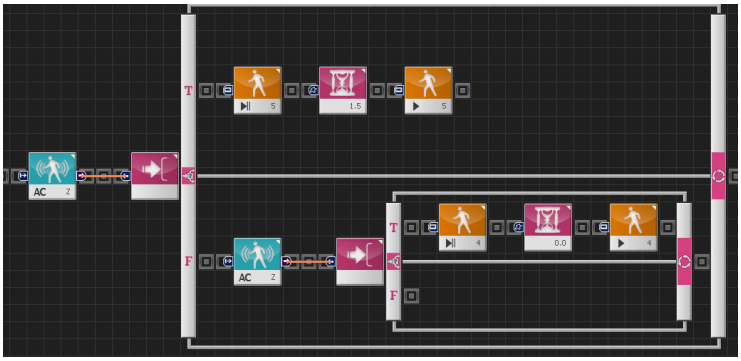
17 Delay

Motion Ready動作が終了する前にモーションが進行することを防止するためにDelay値を1.5秒に設定します。



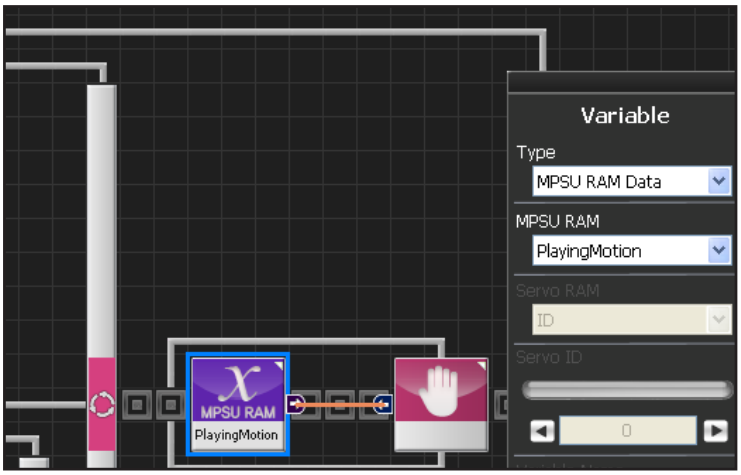
18 前に立ち上がるモーションの実行

Motion ReadyをFalseに選択すると、前に立ち上がるモーションが実行されます。



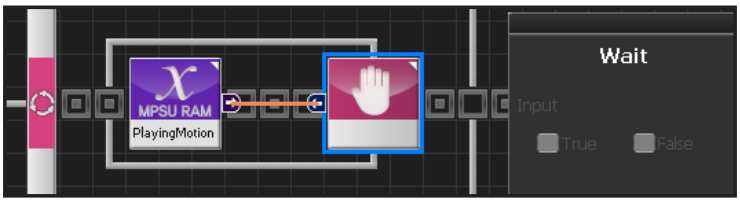
19 倒れたロボットを立ち上がらせる

Z軸の加速度値に基づいてロボットの転倒を判断し、倒れた場合に応じて、それぞれの動作を実行するプログラムです。



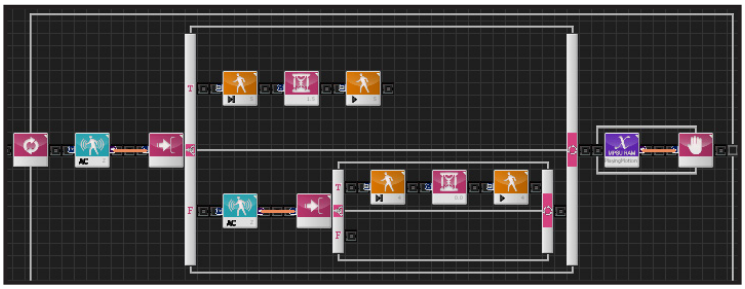
20 モーション動作の確認

Loopは継続的な繰り返しを意味します。Move命令を下してから、実際のモーションが実行され、完了するまでは時間がかかるので、loop内にMoveモジュールの一つだけを入れて実行すると、モーションが既に実行中央にもかかわらずloopを続けてモーションを実行する命令を繰り返すことになります。
この場合、Moveモジュールを実行した回数と実際のモーションを実行する回数が異なります。
したがって実行したモーションが終わるまで待ってから再びloopの最初に戻す方がより確実です。
Variable > MPSU RAM Dataに入ると、Playing Motion があります。Playing Motionはロボットのモーションを実行しているかを確認する変数です。そのPlaying MotionにWaitをかければ、ロボットの動作が終わるまでLoopを待ちます。
Data > Variableモジュールを選択します。
Type：MPSU RAM Dataを選択します。
MPSU RAM：Playing Motionを選択します。
Outputコネクタの値の後ろのWaitモジュールに接続します。
モーションが終わるまで待つことを意味します。



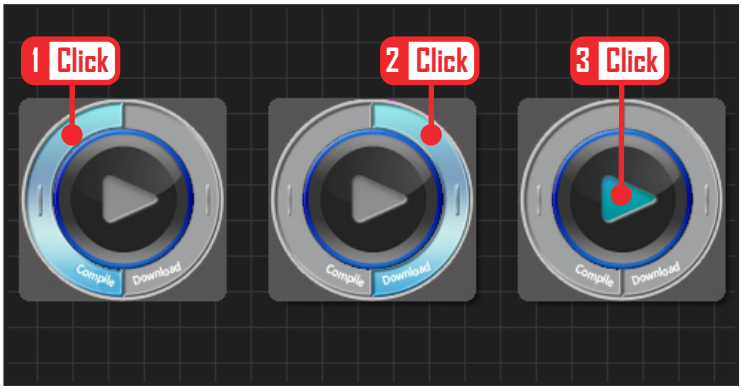
21 Wait

モーションが終わるまで待ちます。
モーションが終われば また最初に戻ってモーションを繰り返します。



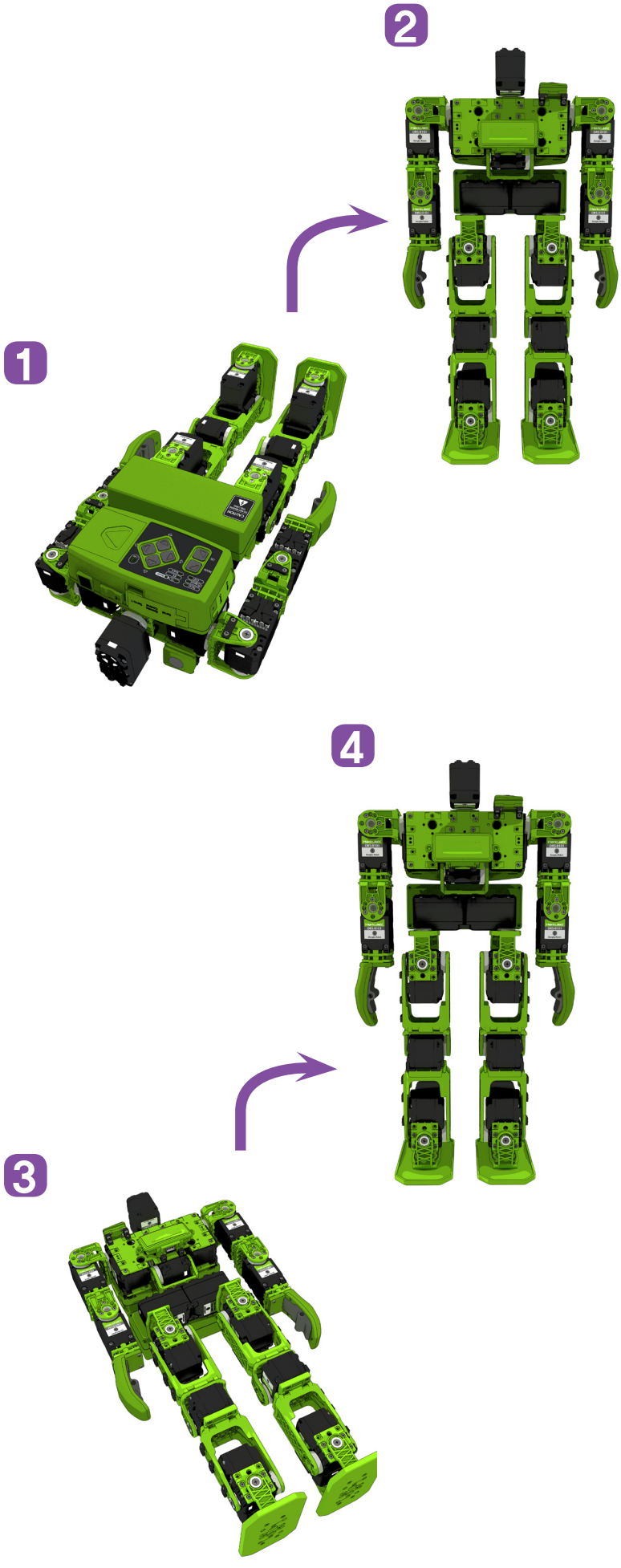
22 全体のプログラム

ロボットが前に倒れた時と後ろに倒れた時を判断して、それぞれ後ろに立ち上がるモーション、前に立ち上がるモーションを実行するプログラムです。



23 コンパイル、ダウンロード、実行

左をクリックしてコンパイルします。エラーがなければ右クリックして、ロボットにダウンロードします。ダウンロードが完了すると、中央の矢印の実行ボタンを押して、ロボットで実行させます。



24 ロボット動作

ロボットが前に倒れている場合は後ろに立ち上がり、ロボットが後ろに倒れている場合は前に立ち上がります。