

gumonji/Q アクション・キューリファレンス

gumonji/Q で用意されているアクション（動作）とキュー（観測）の定義を示す。アクションは感嘆符で、キューは疑問符で始まるものとする。

各アクション・キューに関して、指定するパラメータのリストを示す。パラメータ名はコロンで始まり、in と out は入出力の別を表す。in のパラメータには指定した型の値を与える。out のパラメータにはパターン変数を与えると結果が代入されて返される。

（注意）ロボットは、初期状態ではセルロミンを持っていないため、セルロミンを消費するアクションを実行したい場合は、缶を装備させるなどしてセルロミンを集めてからアクションを実行させること。

1. アクション定義

1.1. 一般アクション

一体のエージェントの動作を指定する、一般アクションの一覧

!stop

指定した時間だけ動作を停止する

:time	in	整数	待機する時間を指定 (gumonji の分単位)	指定しない場合は 1
-------	----	----	--------------------------	------------

※ゾーンの時間が止まっていると動作を停止し続ける

!walk

移動先を指定して歩く

:x :y	in	整数	移動先の絶対座標を指定	指定しない場合は現在地
:dx :dy	in	整数	移動先の相対座標を指定	指定しない場合は 0

※!walk による移動は、ゾーンサーバ上では瞬時に指定された場所へ移動し、3D ブラウザが移動を遅らせて表示することで移動に時間がかかっているように見せかけている。従って、!walk のアクションのみでは移動にかかる時間を確保することができないため、移動時間を確保するためには?wait を用いる必要がある。なお、移動時間の確保の仕方については、今後のバージョンアップにより仕様変更を行う可能性がある。

!approach

対象を指定して歩いて近づく

:id	in	文字列	対象の ID を指定	
:dx :dy	in	整数	対象との相対座標を指定	指定しない場合は 0

!turn

方角を指定して方向転換

:dir	in	整数	方角を指定 (北:0 東:1 南:2 西:3)	
------	----	----	-------------------------	--

!speak

対象と発言内容を指定して話す (ロボットのみ)

:id	in	整数	対象の ID を指定	
:message	in	文字列	発言内容を指定	

!send

対象のエージェントを指定してメッセージを送る（ロボットのみ）

:name	in	文字列	対象エージェントの名前を指定	
:message	in	文字列	メッセージの内容を指定	

!express

表情を指定する（ロボットのみ）

:face	in	文字列	表情を指定	※表 1 参照
-------	----	-----	-------	---------

※現在対応していません

!perform

アクションを実行する

:action	in	文字列	状態を指定	※表 2 および表 3 参照
---------	----	-----	-------	----------------

※現在一部のアクションのみしか対応していません

!look

対象を指定して状態を調べる

:id	in	整数	対象の ID を指定	
:keys	in	文字列	対象から受け取る変数名のリストを指定	※表 4 参照
:values	out	文字列	指定した変数値のアソシエーションリストを取得	

※取得した変数の値は、(get-value-str 変数名 アソシエーションリストへの参照) で得られる。

!observe

地点を指定して地面の状態を調べる

:x :y	in	整数	絶対座標を指定	指定しない場合は現在地
:dx :dy	in	整数	相対座標を指定	指定しない場合は 0
:keys	in	文字列	対象から受け取る変数名のリストを指定	※表 5 参照
:values	out	文字列	指定した変数値のアソシエーションリストを取得	

!name

対象を指定して名前を付ける

:id	in	整数	対象の ID を指定	対象は動物
:name	in	文字列	名前を指定	

!denominate

対象を指定してグモ学名を付ける

:id	in	整数	対象の ID を指定	対象は動物または植物
:name	in	文字列	グモ学名を指定	

!rotate

対象と回転角を指定して回す

:id	in	整数	対象の ID を指定	対象は植物またはアイテム
-----	----	----	------------	--------------

:angle	in	整数	角度を指定	
--------	----	----	-------	--

!harvest

対象を指定して種を収穫する（ロボットのみ）

:id	in	整数	対象の ID を指定	対象は実を付ける植物
-----	----	----	------------	------------

!attach

バッヂを付ける（ロボットのみ）

:badge	in	整数	バッヂの種類を指定	※表 6 参照
--------	----	----	-----------	---------

!get-item-id

指定した種類のアイテムの中で、index 番目のアイテムの ID を得る

アイテムの種類を指定しない場合は、アイテム欄内での index 番目の位置にあるアイテムの ID を得る

:index	in	整数	アイテムの位置を指定	指定しない場合は 1 ※1~48
:type	in	文字列	アイテムの種類を指定	指定しない場合はすべて ※(葉っぱ, うまのタマゴなど)
:id	out	整数	アイテムの ID を取得	

!equip

アイテムを指定して装備する

:id	in	整数	アイテムの ID を指定	
:type	in	文字列	種類を指定	※(缶, シャベルなど)

※ID と種類の両方を指定した場合は、ID を優先。

!put

アイテムを指定した地点に置く

:id	in	整数	アイテムの ID を指定	
:x :y	in	整数	絶対座標を指定	指定しない場合は現在地
:dx :dy	in	整数	相対座標を指定	指定しない場合は 0

!merge

アイテムを指定して同種類のアイテムをまとめる

:id	in	整数	アイテムの ID を指定	
-----	----	----	--------------	--

!overlay

指定した同種類のアイテム 2 つを重ねる

:dest	in	整数	重ねる先のアイテムの ID を指定	
:from	in	整数	重ねる元のアイテムの ID を指定	

!divide

指定したアイテムを半分に分割する

:id	in	整数	アイテムの ID を指定	
-----	----	----	--------------	--

!price

指定したアイテムに指定した値段を付ける

:id	in	整数	アイテムの ID を指定	
:price	in	整数	値段を指定	

!takeout

指定したアイテム（缶・バケツ・タンク）から指定した量のアイテムを取り出す

:id	in	整数	アイテムの ID を指定	
:amount	in	整数	取り出す量を指定	

!store

指定したアイテムを缶・バケツ・タンクに収納する

:id	in	整数	アイテムの ID を指定	
-----	----	----	--------------	--

!pickup

指定したアイテムを拾う

:id	in	整数	アイテムの ID を指定	
-----	----	----	--------------	--

!bury

指定したアイテムを埋める

:id	in	整数	アイテムの ID を指定	
-----	----	----	--------------	--

!plant

指定した種を植える

:id	in	整数	アイテムの ID を指定	
-----	----	----	--------------	--

!incubate

指定した卵を孵化させる

:id	in	整数	アイテムの ID を指定	
-----	----	----	--------------	--

!write

指定したアイテムにコメントを書く

:id	in	整数	アイテムの ID を指定	
:message	in	文字列	コメントの内容を指定	

!read

指定したアイテムのコメントを読む

:id	in	整数	アイテムの ID を指定	
:message	out	文字列	コメントの内容を取得	

!delete

指定したアイテムを削除する

:id	in	整数	アイテムの ID を指定	
-----	----	----	--------------	--

!dig

地点・範囲・深さを指定して地面を掘る

:x :y	in	整数	絶対座標を指定	指定しない場合は現在地
:dx :dy	in	整数	相対座標を指定	指定しない場合は 0
:range	in	整数	範囲を指定	1辺の長さが $(range+1)*2$ の正方形 ※0~2
:height	in	整数	高さ(mm)(差分を指定)	

!raise

地点・範囲・高さを指定して地面を盛る

:x :y	in	整数	絶対座標を指定	指定しない場合は現在地
:dx :dy	in	整数	相対座標を指定	指定しない場合は 0
:range	in	整数	範囲を指定	1辺の長さが $(range+1)*2$ の正方形 ※0~2
:height	in	整数	高さ(mm)(差分を指定)	

!flatten

地点・範囲を指定して、地面をエージェントがいる地点の高さにそろえる

:x :y	in	整数	絶対座標を指定	指定しない場合は現在地
:dx :dy	in	整数	相対座標を指定	指定しない場合は 0
:range	in	整数	範囲を指定	1辺の長さが $(range+1)*2$ の正方形 ※0~2

!smooth

地点・範囲を指定して地面をなめらかにする

:x :y	in	整数	絶対座標を指定	指定しない場合は現在地
:dx :dy	in	整数	相対座標を指定	指定しない場合は 0
:range	in	整数	範囲を指定	1辺の長さが $(range+1)*2$ の正方形 ※0~2

!harden

地点・範囲を指定して地面を固める

:x :y	in	整数	絶対座標を指定	指定しない場合は現在地
:dx :dy	in	整数	相対座標を指定	指定しない場合は 0
:range	in	整数	範囲を指定	1辺の長さが $(range+1)*2$ の正方形 ※0~2

!scoop

地点・範囲・量を指定して水をすくう

:x :y	in	整数	絶対座標を指定	指定しない場合は現在地
:dx :dy	in	整数	相対座標を指定	指定しない場合は 0
:range	in	整数	範囲を指定	1辺の長さが $(range+1)*2$ の正方形

				※0~8
:amount	in	整数	各セルからすくう水の量を指定 (mol)	指定しない場合は 1000

!spill

地点・範囲・量を指定して水を撒く

:x :y	in	整数	絶対座標を指定	指定しない場合は現在地
:dx :dy	in	整数	相対座標を指定	指定しない場合は 0
:range	in	整数	範囲を指定	1辺の長さが $(range+1)*2$ の正方形 ※0~8
:amount	in	整数	各セルに撒く水の量を指定 (mol)	指定しない場合は 1000

!fell

対象を指定して植物を刈る

:id	in	整数	植物の ID を指定	
-----	----	----	------------	--

!hunt

対象を指定して動物を狩る

:id	in	整数	動物の ID を指定	
-----	----	----	------------	--

!chemical

指定したアイテムを作る

:item	in	文字列	アイテムの名前を指定	※ケミカルで作成できるアイテム名(布、木の机など)
-------	----	-----	------------	---------------------------

!play-note

音を鳴らす

:height	in	整数		
:patch	in	整数	音の種類を指定	
:note	in	整数	音の高さを指定	
:length	in	整数	音の長さを指定	

1.2. 全体アクション

全てのエージェントの動作を一度に指定する、アクションの一覧

!stop-all

指定した時間だけ動作を停止する

:time	in	整数	待機する時間を指定 (gumonji の分単位)	指定しない場合は 1
-------	----	----	--------------------------	------------

※ゾーンの時間が止まっていると動作を停止し続ける

!walk-all

移動先を指定して歩く

:x :y	in	整数	移動先の絶対座標を指定	指定しない場合は現在地
:dx :dy	in	整数	移動先の相対座標を指定	指定しない場合は 0

!approach-all

対象を指定して歩いて近づく

:id	in	文字列	対象の ID を指定	
:dx :dy	in	整数	対象との相対座標を指定	指定しない場合は 0

!turn-all

方角を指定して方向転換

:dir	in	整数	方角を指定(北:0 東:1 南:2 西:3)	
------	----	----	------------------------	--

!speak-all

対象と発言内容を指定して話す (ロボットのみ)

:id	in	整数	対象の ID を指定	
:message	in	文字列	発言内容を指定	

!express-all

表情を指定する (ロボットのみ)

:face	in	文字列	表情を指定	※表 1 参照
-------	----	-----	-------	---------

※現在対応していません

!perform-all

アクションを実行する

:action	in	文字列	状態を指定	※表 2 および表 3 参照
---------	----	-----	-------	----------------

※現在一部のアクションのみしか対応していません

!attach-all

バッヂを付ける (ロボットのみ)

:badge	in	整数	バッヂの種類を指定	※表 6 参照
--------	----	----	-----------	---------

!equip-all

アイテムを指定して装備する

:type	in	文字列	種類を指定	※(缶, シャベルなど)
-------	----	-----	-------	--------------

1.3. アクションの優先度

すべてのアクションにおいて、優先度を設定できる。優先度は、複数のアクションを同時に実行しようとした際に、どちらのアクションを先に実行するかを規定するものである。優先度の数値が高いほど、実行が優先される。なお、優先度を指定しない場合は、3 を指定したものと同等とする。

:priority	in	整数	優先度を指定	1~5
-----------	----	----	--------	-----

1.4. アクションの実行時間

すべてのアクションにおいて、実行時間を設定できる。実行時間は、アクションの実行後、次のアクションの実行を開始するまでの時間を指定するものである。実行時間を設定することで、移動や会話などのアクションの

後に、適当な間を置くことができる。

:time	in	整数	待機する時間を指定(gumonji の分単位)	指定しない場合は各アクション固有の待機時間が適応される
-------	----	----	-------------------------	-----------------------------

2. キュー定義

?hear-message

指定した内容に一致する発言が聞こえると、発言した対象の ID を得る

:message	in	文字列	発言内容を指定	
:id	out	整数	発言したキャラクターまたはロボットの ID を取得	
:dist	in	整数	聞こえる範囲(距離)を指定	指定しない場合は無限大

?hear-from

指定した対象の発言を聞くと、その発言内容を得る

:id	in	整数	キャラクターまたはロボットの ID を指定	
:message	out	文字列	発言内容を取得	
:dist	in	整数	聞こえる範囲(距離)を指定	指定しない場合は無限大

?receive-message

指定した内容に一致するメッセージを受け取ると、送り主の名前を得る

:message	in	文字列	メッセージ内容を指定	
:name	out	文字列	キャラクターまたはロボットの名前を取得	
:dist	in	整数	聞こえる範囲(距離)を指定	指定しない場合は無限大

?receive-from

指定した対象からメッセージを受け取ると、その内容を得る

:name	in	文字列	キャラクターまたはロボットの名前を指定	
:message	out	文字列	メッセージ内容を取得	
:dist	in	整数	聞こえる範囲(距離)を指定	指定しない場合は無限大

?see

指定した条件に一致する対象が指定した範囲内に存在すると、その ID を得る

:category	in	文字列	カテゴリを指定	指定しない場合はすべて ※表 7 参照
:type	in	文字列	種類を指定	指定しない場合はすべて ※(葉っぱ, うま, 丸い木など)
:name	in	文字列	名前・グモ学名を指定	指定しない場合はすべて
:color	in	文字列	色を指定	指定しない場合はすべて ※表 8 参照
:x :y	in	整数	対象地点の絶対座標を指定	指定しない場合は現在地
:dx :dy	in	整数	対象地点の相対座標を指定	指定しない場合は 0

:dist	in	整数	対象地点からの範囲(距離)を指定	指定しない場合は 0
:id	out	整数	対象を特定する ID を取得	

?wait

指定した時間が経過する

:time	in	整数	待機する時間を指定 (gumonji 上の分単位)	指定しない場合は 0 (必ず発火)
-------	----	----	---------------------------	-------------------

※ゾーンの時間が止まっていると発火しない

?is-equip

指定した条件に一致するアイテムを装備しているかを調べる

:id	in	整数	アイテムの ID を指定	
:type	in	文字列	種類を指定	※(缶, シャベルなど)

※種類または ID のいずれかの条件に一致するアイテムを装備していれば発火

3. エージェント定義

defagent

エージェントを作成する

:category	in	文字列	カテゴリを指定	robot または animal
:color	in	文字列	色を指定	※ロボット: 表 9 参照 ※動物: 指定の必要なし
:x :y	in	整数	座標を指定	指定しない場合はランダム

※カテゴリでロボットを指定した場合は、指定した座標にロボットが作成される。

- ・ ロボットが最初に持っているアイテムは、缶・シャベル・バケツ・いかだ・インクローラー・手斧・むしめがねの 7 個。
- ・ 持てるアイテムの最大数は、素材 16 個、完成品 16 個、服 16 個の計 48 個。

※動物の場合は、指定した名前が付けられている動物をエージェントにする。

4. Q のコマンド

Q の対話画面 (Q ウィンドウ) では、コマンドを入力することにより、シナリオの読み込みや実行を行うことができる他に、シナリオ実行中のエージェントの情報を得ることもできる。以下に、Q の対話画面で実行可能なコマンドを説明する。

4.1. ファイルの読み込み

シナリオなどを記述したファイルを読み込む。ファイルパスは、“qv2-src”からの相対パスで指定する。

(load “ファイルパス”)

4.2. シナリオの割り当て

以下のコマンドでエージェントにシナリオを割り当てることで、シナリオの実行が開始される。

シナリオ実行中に新たにシナリオを割り当てることや、1つのエージェントに複数のシナリオを割り当てることも可能。複数のシナリオを同時に与えた場合は、並行して実行される。

```
(assign-scenario 'シナリオ名' エージェント名 パラメータリスト)
```

4.3. シナリオの一時停止

以下のコマンドでエージェントに割り当てたシナリオの実行を一時停止する。

```
(stop-scenario 'シナリオ名' エージェント名)
```

以下のコマンドですべてのエージェントのシナリオ実行を一時停止する。

```
(stop-scenario-all)
```

4.4. シナリオの実行再開

以下のコマンドで一時停止したシナリオの実行を再開する。

```
(restart-scenario 'シナリオ名' エージェント名)
```

以下のコマンドですべてのエージェントの一時停止したシナリオの実行を再開する。

```
(restart-scenario-all)
```

4.5. シナリオの開放

以下のコマンドでエージェントに割り当てたシナリオの実行を終了する。

```
(release-scenario 'シナリオ名' エージェント名)
```

4.6. エージェントなどの位置を調べる

以下のコマンドでエージェント・動植物・アイテムの位置を調べることができる。

(get-position 対象)

※対象には、エージェントの名前（文字列）か、エージェント・動植物・アイテムの ID（整数）を指定する。

4.7. エージェントを削除する

以下のコマンドでエージェントを削除する。

(delete-agent “エージェント名”)

※エージェントを削除すると、対応するロボットまたは動物がゾーンから完全に削除される。

4.8. ゾーンのサイズを調べる

以下のコマンドでゾーンのサイズを取得する。

(get-zone-size)

※ゾーンのサイズは、(width height) という形のリストで取得する。

4.9. ログの表示切り替え

以下のコマンドで、ログの種類ごとに表示のオンオフを指定できる。

このコマンドは、シナリオファイルの中に記述することで、初期設定として指定できる。

また、シナリオ実行中に Q ウィンドウ上で使用することで、表示設定を切り替えることができる。

(set-log-print “種類” 設定)

※種類に指定できる値

warning	警告の表示
request	アクション・キューの実行依頼の表示
reply	アクション・キューの実行結果の表示
status	実行待機中のアクション・キューの状態の表示

※設定に指定する値は#t または#f であり、#t で表示をオンに、#f で表示をオフにする。

4.10. シナリオ中で使用できるコマンド

1. 別のシナリオを呼び出す

```
(シナリオ名 self パラメータリスト)
```

- 整数 a と b の間のランダムな整数を得る。(a と b は含まない)

```
(get-random-int a b)
```

- `!look` や `!observe` で得たアソシエーションリストから、指定したキーの値の文字列表現を得る。

```
(get-value-str "キー名" アソシエーションリスト)
```

- `!look` や `!observe` で得たアソシエーションリストから、指定したキーの値の整数値を得る。

```
(get-value-int "キー名" アソシエーションリスト)
```

- ゾーンのサイズを取得する。

ゾーンの横幅の整数値を得る。

```
(get-zone-width)
```

ゾーンの縦幅の整数値を得る。

```
(get-zone-height)
```

- デバッグ用コマンド

引数の値を Q ウィンドウ上に表示する。

```
(debug-output 引数)
```

4.11. その他

Q は、KAWA という Scheme 処理系の上で実行しているため、一般的な Scheme の式を使用することができる。詳しくは、KAWA のウェブサイト (<http://www.gnu.org/software/kawa/index.html>) や、Scheme に関する書籍を参照のこと。

5. 補足

(表 1) 表情リスト

default
shy
depression
steam
birds
sweat
heart
light
cheerful
stars
anger
flame
beam
kiss
exclamation
question
bigtears
smalltears
zzz
music

(表 2) 動物のアクションリスト

normal
pregnant
halfdead
poopsoon
hungry
sitdown
sleeping

(表 3) ロボットのアクションリスト

stand
walk
turn
lie
throw
wave
sit
jump
bow
swim

(表 4) !look パラメータ名リスト

category	文字列	カテゴリ名	
name	文字列	名前	
family	文字列	グモ学名	動物・植物・アイテム

type	文字列	種類	
color	文字列	色	
x	整数	横座標	
y	整数	縦座標	
dir	整数	方向	
state	整数	状態	動物・植物 ※注 1, 注 2
condition	整数	成長状態	動物・植物 ※注 3
age	整数	年齢	動物・植物
generation	整数	世代	動物・植物
overhead	整数	頭上キャラクターの ID	動物
item	整数	頭上アイテムの ID	動物
mol	整数	養分(物質質量)	動物・植物・アイテム
fertrecord	整数	最大養分	動物・植物
price	整数	値段	アイテム

※注 1 動物の状態

0	通常状態
1	妊娠状態
2	死にかけ(倒れ)
3	ハラヘリ
4	もうすぐウンコする
5	すわっている
6	眠っている

※注 2 植物の状態

0	通常
1	幼苗
2	花
3	種が 1 つ
4	種が 2 つ

※gumonji の木のみ仕様が異なり、3 では種を収穫できず、4 で種が 1 つとなります。

※注 3 成長状態

0	成長中
1	停滞中
2	衰弱中

(表 5) !observe パラメータ名リスト

height	整数	高さ(mm)
rock	整数	岩盤の高さ(mm)
water	整数	地下水の量(mol)
fert	整数	養分の量(mol)
editor	整数	最後に edit した人のユーザ ID
walker	整数	最後に歩いた人のユーザ ID

edited	整数	最後に edit した日
walked	整数	最後に歩いた日
plant	整数	植物の ID
item	整数	アイテムの ID
animal	整数	動物の ID

(表 6) バッジリスト

0	なし
1	初心者
2	太陽
3	月
4	星
5	花
6	雪
7	葉

(表 7) カテゴリリスト

robot
character
animal
plant
item

(表 8) 色リスト

cream
skyblue
black
muscat
red
orange
pink
pearlblue
blue
gray
green
white
yellow
lisa
chocolate
grape
beige

(表 9) ロボット色リスト

black
red
blue

gray
yellow
grape