SHARP

RoBoHoN
HVML 2.0 リファレンス

Version 2.0.0

Last update 2019/8/7

SHARP CORPORATION

SHARP

更新履歴

Version	Description	Date
2.0.0	0603_SR01MW_HVML2.0_Specification をベースに第２世代ロボホン向けのアップデートを追記	2019/8/7

SHARP

目次

更新履歴 2

目次 3

1. はじめに 8

1.1 本資料の目的 8
1.2 著作権 8
1.3 免責事項 8
1.4 表記関係について 8
1.5 用語の定義 8
1.6 参考資料 9

2. HVML の概説 10

2.1 HVML とは何か 10
2.2 HVML の基本形式 11

3. HVML の実行 12

4. タグ 15

4.1 タグ一覧 15
4.2 タグの階層図 15
4.3 A タグ 16
4.3.1 HREF 属性 16
4.3.2 TYPE 属性 16
4.3.3 使用例 17
4.3.4 注意事項 17
4.4 ACCOST タグ 19
4.4.1 WORD 属性 19
4.4.2 TOPIC_ID 属性 19
4.4.3 PRIORITY 属性 19
4.4.4 使用例 19
4.4.5 注意事項 19
4.5 ACTION タグ 20
4.5.1 INDEX 属性 20
4.5.2 使用例 20
4.5.3 注意事項 20
4.6 BEHAVIOR タグ 21
4.6.1 TYPE 属性 21

4.6.2 ID 属性 21
4.6.2.1 ID 一覧 21
4.6.3 使用例 24
4.7 BODY タグ 25
4.8 CASE タグ 25
4.8.1 ID 属性 25
4.8.2 LIMIT 属性 25
4.8.3 CLEARDAYS 属性 25
4.8.4 使用例 26
4.8.5 注意事項 26
4.9 CONDITION タグ 27
4.9.1 CASE_ID 属性 27
4.9.2 WEIGHT 属性 27
4.9.3 PRIORITY 属性 27
4.9.4 使用例 27
4.9.5 注意事項 28
4.10 CONTROL タグ 29
4.10.1 TARGET 属性 29
4.10.2 FUNCTION 属性 29
4.10.3 使用例 29
4.10.4 注意事項 29
4.11 DATA タグ 30
4.11.1 KEY 属性 30
4.11.2 VALUE 属性 30
4.11.3 使用例 30
4.12 DESCRIPTION タグ 31
4.12.1 使用例 31
4.13 EMOTION タグ 32
4.13.1 TYPE 属性 32
4.13.2 LEVEL 属性 32
4.13.3 使用例 32
4.13.4 注意事項 32
4.14 HEAD タグ 33
4.15 HVML タグ 33
4.15.1 VERSION 属性 33
4.16 MEMORY タグ 34
4.16.1 TYPE 属性 34
4.16.2 KEY 属性 34
4.16.3 VALUE 属性 34
4.16.4 OPERATION 属性 34

4.16.5 使用例 35
4.16.6 注意事項 36
4.17 NEXT タグ 36
4.17.1 HREF 属性 36
4.17.2 TYPE 属性 36
4.17.3 使用例 37
4.17.4 注意事項 37
4.18 PRODUCER タグ 38
4.18.1 使用例 38
4.18.2 注意事項 38
4.19 RULE タグ 39
4.19.1 使用例 39
4.19.2 注意事項 39
4.20 SCENE タグ 40
4.20.1 VALUE 属性 40
4.20.2 使用例 40
4.20.3 注意事項 40
4.21 SITUATION タグ 41
4.21.1 TRIGGER 属性 41
4.21.2 VALUE 属性 41
4.21.3 TOPIC_ID 属性 41
4.21.4 PRIORITY 属性 41
4.21.5 使用例 42
4.21.6 注意事項 42
4.22 SPEECH タグ 43
4.22.1 使用例 43
4.22.2 注意事項 43
4.23 TOPIC タグ 44
4.23.1 ID 属性 44
4.23.2 LISTEN 属性 44
4.23.3 LISTEN_MS 属性 44
4.23.4 DICT 属性 44
4.23.5 使用例 45
4.24 VERSION タグ 46
4.24.1 VALUE 属性 46
4.24.2 使用例 46
4.24.3 注意事項 46
4.25 WAIT タグ 47
4.25.1 MS 属性 47
4.25.2 使用例 47

4.25.3 注意事項 47
5. 演算子 48
5.1 演算子一覧 48
5.1.1 EQ/NEQ 48
5.1.2 GT 48
5.1.3 GE 48
5.1.4 LT 48
5.1.5 LE 48
5.1.6 AND 49
5.1.7 IN 49
5.1.8 OUTOF 49
5.1.9 INCLUDE 49
5.1.10 NEAR 49
5.1.11 +, -, *, / 49
5.2 演算子を利用できる箇所 50
6. 変数 51
6.1 変数のフォーマット 51
6.2 変数一覧 51
6.2.1 LVCSR:BASIC/LVCSR:KANA 53
6.2.2 LOCAL_REPLY:GLOBAL_REPLY_YES/LOCAL_REPLY:GLOBAL_REPLY_NO 53
6.2.3 LOCAL:WORD_APPLICATION/LOCAL:WORD_APPLICATION_FREEWORD 53
6.2.4 YEAR/MONTH/DAY 53
6.2.5 DATE 54
6.2.6 DAYOFWEEKJP 54
6.2.7 HOUR/MINUTE/SECOND 54
6.2.8 TIME 54
6.2.9 NOW 54
6.2.10 CURRENTDATE 54
6.2.11 CURRENTYEARDATE 54
6.2.12 CURRENTTIME 55
6.2.13 INDATERANGE 55
6.2.14 INTIMERANGE 55
6.2.15 ISWITHINTIME 55
6.2.16 DIFFDATE 55
6.2.17 SIZEOF 55
6.2.18 SELECT 56
6.2.19 YEAROF/MONTHOF/DAYOF/HOUROF/MINUTEOF/SECONDOF 56
6.2.20 RESOLVER:CONTACTS:ROBOT_NAME 56
6.2.21 RESOLVER:CONTACTS:USER:BIRTHDAY 56
6.2.22 RESOLVER:CONTACTS:USER:NAME_FOR_SPEECH 56
6.2.23 RAND 56
6.2.24 ENV:ROBOT_INIBOOT_DATE 57
6.2.25 RESOLVER:CHARGER_CONNECTED 57
6.2.26 RESOLVER:EARPHONE_CONNECTED 57

6.2.27 MEMORY_P:MANNER_MODE 57
6.2.28 RESOLVER:POSE 57
6.2.29 RESOLVER:SPEECH_OK 58
6.2.30 RESOLVER:MOTION_OK 58
6.2.31 RESOLVER:OK_ID 58
6.2.32 ENV:TELEPHONY_SUPPORT 58
6.2.33 ENV:BATTERY:LEVEL 58
6.2.34 ENV:PROJECTOR_SUPPORT 58
6.2.35 ENV:SITMODEL 59
6.2.36 ENV:ROBOHON_GENERATION 59
6.3 MEMORY 変数 60
6.4 変数の追加 60
6.5 変数型 61
6.5.1 日時型 61
6.5.2 リスト型 63
6.6 変数を使用できる箇所 63
6.7 禁止文字 63
6.8 変数を SITUATION に記載する際の注意点 64
7. 環境検知イベント 65
7.1 環境検知イベント一覧 65
7.2 使用例 65

1. はじめに

1.1 本資料の目的

本資料は、ロボホンの音声操作を実現するシナリオを記述するための言語である HVML に関する仕様書です。
対象読者はロボホンのアプリ開発者となります。
本資料記載の内容は特別な記載のない限り、ロボホンのビルド番号 03.01.00 以降のバージョンを対象にしています。
ロボホンのアプリ開発をご利用の際は「設定 – 端末情報 – ソフトウェア更新」より最新のバージョンにアップデートしてからご利用ください。

1.2 著作権

1.3 免責事項

当社は、本資料の内容が将来にわたり正常に作動すること、ならびに、常時利用できることを保証しません。また、本資料の内容が正常に作動しないことおよび本資料の内容が利用できないことにより開発者が損害を被った場合、当社は当該損害に関して一切責任を負いません。
本資料の内容は予告なしに変更される場合があります。

1.4 表記関係について

本資料に記載されている会社名、製品名などは、各社の登録商標または商標、商品名です。
会社名、製品名については、本文中では©、®、™マークなどは表示していません。

1.5 用語の定義

本資料で出てくる用語の意味を示します。

用語	意味
ロボホン	ロボホン(3G・LTE)、ロボホン(Wi-Fi)、ロボホンライトの総称
第 1 世代ロボホン	SR01MW/ SR02MW を指します
第 2 世代ロボホン	SR03M/SR04M/SR05M を指します
HVML	Hyper Voice Markup Language の略語
TTS	Text-to-Speech の略語。音声合成を利用してテキストを読み上げる機能です。
対話	ユーザの発話⇒ロボホンの応答のやり取り。HVML の実行開始からすべての HVML の topic 実行が終了される間を指します。

1.6 参考資料

	Title
[1]	RoBoHoN_アプリ開発スタートガイド
[2]	RoBoHoN_API リファレンス
[3]	RoBoHoN_アプリ開発ガイドライン

2. HVML の概説

2.1 HVML とは何か

HVML(Hyper Voice Markup Language)とは XML1.0 をベースとしたマークアップ言語であり、電子機器と人間との対話を表現したシナリオを実現することを目的として設計されています。
特徴として、構造表現に向いている XML をベースとしているため、音声対話のシナリオを「ユーザ発話」と「対応するアクション」の組み合わせで表現している点が挙げられます。
実際に人間同士がするような雑談のようなものに加え、ニュースの読み上げ、天気予報のお知らせなどが実現できます。
また機器を発話させるだけでなく、機器の動作や、アプリへのイベント通知などを記述することが可能です。

ロボホンとの対話(ロボホンの発話、および音声認識による操作)を実現するためには、HVML が記述されたファイルを登録し、その HVML を実行するという手順が必要になります。

2.2 HVML の基本形式

一つの HVML ファイルは head と body で構成されます。
主に head は HVML ファイル自体に付随する情報、HVML が選択されるために必要とする情報を記述し、body では実行する内容について記述されます。
また body 内は一つ以上の topic で構成されます。topic は実行内容の区切りのようなものです。

簡単な雑談 HVML シナリオのサンプルを記載します。

【雑談 HVML シナリオサンプル】

```
<hvml version="2.0">
  <head>
    <version value="1.0"/>
    <producer>jp.co.sharp.producer</producer>
    <scene value="jp.co.sharp.producer.sample"/>
    <situation topic_id="0001" trigger="user-word">${Lvcsr:Basic} eq こんにちは</situation>
  </head>
  <body>
    <topic id="0001" listen="false">
      <action index="1">
        <speech>こんにちは。今日もいい天気ですね。</speech>
      </action>
    </topic>
    <topic id="0002" listen="false">
      …
    </topic>
  </body>
</hvml>
```

HVML の記述に関する詳細については次章以降を参照してください。上記サンプルは、ユーザが機器に「こんにちは」と話しかけると機器側から「こんにちは。今日もいい天気ですね。」と TTS で応答するサンプルです。このように、ユーザとの対話を表現することが可能となります。
また、topic 実行後にユーザ発話を受け付けて、次の topic に遷移させることができます。この際 topic は別の HVML ファイルの topic に遷移させることも可能です。これによって、シナリオに沿ったユーザとの対話を実現することが可能となっています。

3. HVML の実行

ロボホンにおける HVML 実行時の一連の流れを【HVML 実行用サンプル】を元に説明します。

HVML の詳細については次章以降で記載するため、本章では概要のみ記載します。

【HVML 実行用サンプル】

```
<hvml version="2.0">
  <head>
    <version value="1.0"/>
    <producer>jp.co.sharp.producer</producer>
    <scene value="jp.co.sharp.producer.sample"/>
    <situation topic_id="0001" trigger="user-word" >${Lvcsr:Basic} eq こんにちは</situation>
  </head>
  <body>
    <topic id="0001" listen="true">
      <action index="1">
        <speech>こんにちは。今日もいい天気ですね。</speech>
      </action>
      <a href="#0002">
        <situation trigger="user-word">${Lvcsr:Basic} eq そうですね</situation>
      </a>
      <a href="#0003">
        <situation trigger="user-word">${Lvcsr:Basic} eq 雨ですよ</situation>
      </a>
      <a href="#0004" type="default"/>
      <next href="#0005" type="default"/>
    </topic>
    <topic id="0002" listen="false">
      <action index="1">
        <speech>こんにちは。今日もいい天気ですね。</speech>
      </action>
    </topic>
    …
    …
  </body>
</hvml>
```

a

b

c

1. HVML が実行されていない状態(idle 状態)で、以下のいずれかの契機により HVML の実行が開始されます。
   - ユーザによる発話
   - 環境検知イベント（ロボホンが振られた、など。詳しくは 7. 環境検知イベント参照）
   - accost イベントによる強制 HVML 実行（詳しくは 4.4 accost タグ参照）

【HVML 実行用サンプルからの抜粋(a)】

```
<head>
  <version value="1.0"/>
  <producer>jp.co.sharp.producer</producer>
  <scene value="jp.co.sharp.producer.sample"/>
  <situation topic_id="0001" trigger="user-word" >${Lvcsr:Basic} eq こんにちは</situation>
</head>
```

2. 上記が発生すると、登録されているすべての HVML の head 情報から、HVML の topic が選択されます。
   HVML が選択されるための条件は以下となります。
   - いずれかの scene タグの value に記載されている値が、現在シーンとして設定されている状態であること
   - ユーザ発話契機および環境検知イベント契機の場合、situation で条件に一致すること。条件が一致する situation が複数存在する場合は、priority が最も高い situation が選択される。priority が最も高い situation が複数ある場合はランダムで選択される。
   - accost イベント契機の場合、accost タグの word 属性の値とイベント名が一致すること

【HVML 実行用サンプルからの抜粋(b)】

```
<topic id="0001" listen="true">
  <action index="1">
    <speech>こんにちは。今日もいい天気ですね。</speech>
  </action>
    …
</topic>
```

3. topic が選択されると、topic 内の action が index 順に実行されます。action にはロボホンの発話、モーション再生などが含まれます。

【HVML 実行用サンプルからの抜粋(c)】

```
<a href="#0002">
  <situation trigger="user-word">${Lvcsr:Basic} eq そうですね</situation>
</a>
<a href="#0003">
  <situation trigger="user-word">${Lvcsr:Basic} eq 雨ですよ</situation>
</a>
<a href="#0004" type="default"/>
<next href="#0005" type="default"/>
```

4. action 終了後、topic の listen="false"の場合、next タグがあれば href に記載の topic へ、next タグが無ければ HVML の実行を終了します。listen="true"であれば、次のユーザ発話を待ち受けます。
   ユーザ発話を受けると、a タグおよび他の HVML を含む head から、HVML の topic が選択され、3 の手順に遷移します。
   複数条件が合致する場合、最も priority が高い situation の topic が選択されます。priority も同一の場合は、topic に記述されている a タグの記載順⇒HVML の head という優先順で選択されます。
   条件が合致する topic が無かった場合は、そのまま対話を終了します。

4. タグ

本章では HVML を構成する要素の一つである、タグについて記載します。

4.1 タグ一覧

タグ一覧を記載します。(アルファベット順)

- <a>
- <accost>
- <action>
- <behavior>
- <body>
- <case>
- <condition>
- <control>
- <data>
- <description>
- <emotion>
- <head>
- <hvml>
- <memory>
- <next>
- <producer>
- <rule>
- <scene>
- <situation>
- <speech>
- <tool_version>
- <topic>
- <version>
- <wait>

4.2 タグの階層図

タグの階層図を記載します。

```
<hvml> : <head> | <body>
    - <head> : <procuder> | <scene> | <situation> | <accost> | <version> | <description>
                | <tool_version>
    - <body> : <topic>
        - <topic> : <rule> | <case> | <action> | <a> | <next>
            - <rule> : <condition>
            - <case> : <action> | <a> | <next>
            - <action> : <speech> | <behavior> | <control> | <memory>
                - <speech> : <emotion> | <wait>
                - <control> : <data>
            - <a> : <situation>
```

4.3 a タグ

a はアンカー(anchor)の略です。指定した遷移先にジャンプします。一つの topic タグまたは case タグに対し、複数記述することもできます。topic の属性 listen="false"が指定されている場合は、その topic 中の a タグは無視されます。

表 4-1 概要

項目	説明
書式	~, <a href="…" ~/>
親要素	<topic>, <case>
子要素	<situation>
属性	href(必須), type

4.3.1 href 属性

遷移先を指定します。同一 HVML ファイル内の topic、別 HVML ファイルの topic を指定することができます。別ファイルの場合は「HVML ファイル名#topic id」のフォーマットで、同一ファイル内の topic 遷移の場合はファイル名を省略し、「#topic id」のフォーマットで記載してください。

【href 属性記載フォーマット】

```
href="jp_co_sharp_rb_sample_1.hvml#001"
```

4.3.2 type 属性

他のアンカーに合致しない場合の遷移を記述したい場合は、type="default"を指定します。

4.3.3　使用例

```
<hvml version="2.0">
  <head>
    …
  </head>
  <body>
    <topic id="0001" listen="true">
      <action index="1">
        <speech>今日はどこに行きますか？</speech>
      </action>
      <a href="#tokyo">
        <situation trigger="user-word">${Lvcsr:Basic} eq 東京</situation>
        <situation trigger="user-word">${Lvcsr:Basic} eq 東京都</situation>
      </a>
      <a href="#osaka">
        <situation trigger="user-word">${Lvcsr:Basic} eq 大阪</situation>
        <situation trigger="user-word">${Lvcsr:Basic} eq 大阪府</situation>
      </a>
      <a href="#retry" type="default" />
    </topic>
    <topic id="tokyo" listen="false">
      …
    </topic>
    …
  </body>
</hvml>
```

4.3.4　注意事項

href 属性が同じ situation が複数ある場合、一つの a タグ内に複数の situation を含めることも可能です。

例えば、

```
<a href="#tokyo">
  <situation trigger="user-word">${Lvcsr:Basic} eq 東京</situation>
</a>
<a href="#tokyo">
  <situation trigger="user-word">${Lvcsr:Basic} eq 東京都</situation>
</a>
```

は以下のように記述しても同じ動作になります。

```
<a href="#tokyo">
  <situation trigger="user-word">${Lvcsr:Basic} eq 東京</situation>
  <situation trigger="user-word">${Lvcsr:Basic} eq 東京都</situation>
</a>
```

4.4 accost タグ

アプリから HVML の topic を実行したい場合に使用します。アプリで word(accost 名称)を実行すると、定義されている topic が実行されます。アプリでの実行方法は参考資料[3]を参照ください。
situation タグとは異なり、要素の内容に条件式を記述することはできません。

表 4-2 概要

項目	説明
書式	<accost topic_id="…" word="…" ~/>
親要素	<head>
子要素	なし
属性	word(必須), topic_id(必須),priority

4.4.1 word 属性

accost 名称を記載します。
accost 名称は一意にする必要があります。package 名から始まる文字列を指定してください。

4.4.2 topic_id 属性

accost のイベントを受信した際に遷移する topic の id を記載します。

4.4.3 priority 属性

HVML 実行の優先度を指定します。
accost イベントを受けた時に、accost または situation で実行された topic が実行中の場合、実行時に指定された priority より高ければ、実行中の HVML を中断し、topic_id 属性に記載の topic が実行されます。
61～90 の間の数字を指定してください。数字が小さい方が優先されます。本属性を記載しない場合は 75 と扱われます。

4.4.4 使用例

```
<accost topic_id="1" word="jp.co.sharp.rb.sample.accost" priority="75"/>
```

4.4.5 注意事項

head に accost タグの記述がある場合でも、scene が一致しなければ topic は実行されません。
また、accost 名称（word 属性）は必ず一意になるようにしてください。同じ accost 名称の accost が複数存在する場合の動作は不定です。

4.5 action タグ

HVML 実行時に実施される操作に関する内容を記述します。一つの topic 内に複数記述することも可能です。
本タグは speech タグ、behavior タグ、control タグ、memory タグを子要素に持ちます。これらのタグは一つの action 内に併記することが可能です。ただし、memory タグ以外は同一のタグは一つの action 内に一つしか記述できません。

表 4-3 概要

項目	説明
書式	<action index="…">~</action>
親要素	<topic>,<case>
子要素	<speech>,<behavior>,<control>,<memory>
属性	index(必須)

4.5.1 index 属性

action の再生順序を示します。1 以上の数字を指定してください。数字が小さい順に再生されます。同じ数字があった場合の順番は保証されません。

4.5.2 使用例

```
<topic id="0001" listen="true">
  <action index="1">
    <speech>こんにちは</speech>
  </action>
  <action index="2">
    <speech>今日はいい天気ですね</speech>
  </action>
</topic>
```

4.5.3 注意事項

一つの action 内に記述されたタグの内容は、記述されている順によらず同時に実行されます。

4.6 behavior タグ

発話に合わせて行うモーションを指定します。

表 4-4 概要

項目	説明
書式	<behavior type="…" id="…"/>
親要素	<action>
子要素	なし
属性	type(必須),id(必須)

4.6.1 type 属性

“normal"を指定してください。

4.6.2 id 属性

モーションを指定します。

“assign","general"または個別のモーション ID のいずれかを指定してください。

モーションに関する詳細な説明は、参考資料[3]を参照ください。

4.6.2.1 id 一覧

表 4-5 モーション ID 一覧

モーション ID	モーションの説明	発話例	ビルド番号
060000	腕を軽く前に出す	好き！、楽しい！	
060001	バンザイ	やった！、バンザイ！	
060002	両腕体横で左右に振る	びっくりー！、すごーい！	
060003	両手上下で抗議	むかつく	
060004	うなだれ	悲しい、寂しい	
060005	両手もじもじ	えへへ	
060006	腰に手	えっへん	
060007	お辞儀	ありがとう	
060008	浅いお辞儀	お疲れ	
060009	右手をあげてふる	バイバイ	
06000a	頷いて右手上げる	うん、～だよね、～しよう	
06000b	首を振る	違うよ、ううん	
06000c	首をかしげる	分からない、～かな？	
06000d	両腕を体の横で前後に振る	帰る、行く、歩く、走る	

06000e	両手でハンドルをにぎにぎ	車、運転	
06000f	両腕を体の横でぶらぶら	ひま	
060010	頭を左右、両腕を前後に動かす	忙しい	
060011	書類や本を読む	会社、学校、勉強、読書	
060012	右手を口元に持っていく	食べる、ご飯に行く、ランチする	
060013	上を向いてジョッキを飲み干す	飲む、飲み会、合コン	
060015	片腕を枕のようにして首を傾ける	寝る	
060016	お腹のあたりをさする	お腹すいた、空腹	
060017	片手で、顔を仰ぐ	暑い	
060018	両手を体の横で震わせる	寒い、凍える	
060019	あくびをする	眠い	
06001a	電話を掛ける	電話、携帯電話	
06001b	おでこを指す	カメラ	
06001c	おでこを指す	プロジェクター	
06001d	自分を指す	ぼく、ロボホン	
06001e	相手を指す	君	
06001f	両手を体の前で振る	音楽、ダンス	
060021	1 回頷く	うん	
060022	2 回頷く	うんうん	
060023	両手を広げる	おめでとう！	
060025	2 回お辞儀	あけましておめでとうございます	
060026	両手を広げた後、右手を前	トリックオアトリート！	
060027	両手を広げる	メリークリスマス！	
060028	ハグする	大好き	
060029	両手を広げて頷く	なるほど、そうなんだ	
06002a	ぐったり	ツライ、しんどい、二日酔い	
06002b	両手を目にあてる	淋しい、メソメソ	
06002c	右手を上げる	おはよう！	
06002d	両手を広げる（怒）	大嫌い！、最低！	
06002e	右手で頭をかく	てへぺろ、あちゃー	
060030	両手を前で広げる	大きい、広い	
060031	両手を前で狭める	小さい、狭い	
060032	両手を体の横で広げる	長い	
060033	右手を上に上げる	高い	
060034	足元に片手を持ってくる	低い	
060035	ドリブルの仕草	サッカー	
060036	手でボールをつく	バスケ	
06003b	片手で投げキス	チュッ	
06003d	両手を体の横で、下を向いて首を振る	やれやれ	

06003e	肩をたたく感じ	まあまあ、落ち着いて	
06003f	俯いて悲しく手を振る	バイバイ	
060040	乾杯	かんぱい	
060041	そっぽをむく	ふんっ	
060043	慌てる	あたふた	
060044	背中をみる	背中、後ろ	
060046	両手でお腹周りをたたく	おなかいっぱい	
060047	きょろきょろする	どこにいる？、どこ	
060048	目を指す	目	
060049	鼻を指す	鼻	
06004a	腰を指す	腰	
06004d	横を向いて、手を広げて、足を曲げて、前後に体重移動している感じを表現	スノボ	
06004f	両手を胸の前でドラミング	ゴリラ	
060050	両手を胸の前に上げて、左右にゆらゆら	音楽	
060051	片手を上げて、片手で弓を弾く動き	バイオリン	
060052	両手を胸前で左右バラバラに上下に動かす	太鼓、ティンパニー	
060053	歯磨きする仕草	歯磨き	
060057	右腕を上げて、体をそる	シャワー	
060059	息が上がった感じ	疲れ、腰がいたい	
06005a	腰を曲げて後ろで手を組む	おんぶ	
06005b	片手を上にあげて頭をこつんと打つ感じ	まいった、しまった	
06005c	手を口に当てて少し斜め下を見る	うふふ	
06005d	右腕を前に出して上下に動かす	じゃんけん	
06005e	マイクをもって歌っているような動き	カラオケ、歌う	
06005f	両手胸の前に持ってきて右手を口元にもってくる	ラーメン、うどん、丼	
060062	両手を横に広げて前から後ろへ動かす	飛行機	
060064	ポインターもって白板コツコツ	プレゼン	
06007a	両腕頭で腰を落とす	スクワット	
06007b	カンフーのポーズ	カンフー	

4.6.3　使用例

```
<topic id="0001" listen="true">
  <action index="1">
      <speech>バイバイ</speech>
      <behavior type="normal" id="0x060009"/>
  </action>
</topic>
```

4.7 body タグ

HVML で実行する内容を記述します。body は一つ以上の topic から構成されます。

表 4-6 概要

項目	説明
書式	<body>~</body>
親要素	<hvml>
子要素	<topic>
属性	なし

4.8 case タグ

rule タグに対応する操作を記述します。topic 内で rule タグの記述に従って case が選択・実行されます。選択されなかった case は実行されません。

表 4-7 概要

項目	説明
書式	<case id="…" ~>~</case>
親要素	<topic>
子要素	<action>,<a>,<next>
属性	id(必須),limit,cleardays

4.8.1 id 属性

rule で選択される case を指定するための id です。同一 topic 内で重複することが無いように指定してください。

4.8.2 limit 属性

回数制限を行いたい場合に指定します。1 以上の数値を指定してください。指定した回数実行されると、以降はその case は選択されなくなります。
cleardays 指定によって制限は解除されます。

4.8.3 cleardays 属性

回数制限を解除するまでの日数になります。1 以上の数値を指定してください。

4.8.4　使用例

```
<body>
    <topic id="1" listen="false">
        <rule>
            <condition case_id="1" priority="1">true</condition>
            <condition case_id="2" priority="2">true</condition>
        </rule>
        <case id="1" limit="3" cleardays="1">
            <action index="1">
                <speech>おかえり。今日も頑張ったね</speech>
            </action>
        </case>
        <case id="2">
            <action index="1">
                <speech>おかえり</speech>
            </action>
        </case>
    </topic>
</body>
```

id=1 の topic が実行された場合、
1～3 回目：case id=1「おかえり。今日も頑張ったね」が発話されます。
4 回目以降：case id=2「おかえり」が発話されます。
3 回目の「おかえり。今日も頑張ったね」の発話以降、1 日(24 時間)が経過するまでは「おかえり」が発話されます。1 日が経過した後の topic の実行では再び「おかえり。今日も頑張ったね」が発話されます。

4.8.5　注意事項

case に limit 回数制限がかかった場合、その case は選択されなくなります。topic 内で condition の条件に合致する case が他に無くシナリオが終了することにならないよう、limit 指定をしない case もあわせて記述するようにしてください。

4.9 condition タグ

topic 内で実行する処理(case)を決定するための条件を記載します。要素の内容に条件式を記述し、条件式の結果が真(true)であれば合致と判定されます。条件式の記載方法に関しては後述の 5.演算子、6.変数の章を参照してください。
常に真となる条件を記載したい場合、要素の内容に true と記載してください。

表 4-8 概要

項目	説明
書式	<condition case_id ="…" ~>~</condition>
親要素	<rule>
子要素	なし
属性	case_id(必須),weight,priority

4.9.1 case_id 属性

タグ要素の内容に記載された条件に合致した場合に遷移する case の id を指定します。
一つの rule 内で複数の condition に同一の case_id 値が指定しても問題ありません。

4.9.2 weight 属性

case が選択されるための比率を指定します。1 以上の数字を指定してください。指定が無い場合は 10 と扱われます。

4.9.3 priority 属性

case を選択する際の優先度を指定します。1～99 の数字を指定してください。指定が無い場合は 10 と扱われます。また、数字が小さい方が優先されます。

4.9.4 使用例

```
<rule>
    <condition case_id="1" weight="3">${Hour} ge 20</condition>
    <condition case_id="2" weight="1">${Hour} ge 22</condition>
    <condition case_id="3" weight="1">${Hour} ge 22</condition>
    <condition case_id="4" priority="20">true</condition>
 </rule>
```

20～22 時　：　case 1 が実行されます
22~24 時　：　60%の確率で case1、20%の確率で case2、20%の確率で case3 が実行されます。
0~20 時　：　case4 が実行されます。

4.9.5　注意事項

条件が合致する condition が一つであった場合、case_id に記載された id の case に遷移します。複数合致した場合は、priority が最も高い condition の case に遷移します。さらに priority も同じであった場合、weight に記載された比率に従ってランダムで case が選択されます。
condition の記載順は選択される確率に影響しません。
条件が合致する condition が一つも無い場合は、その時点で対話が終了してしまいます。条件式 true の condition を一つは入れる等、対話終了することが無いよう記述してください。
また、case に回数制限がかかっている場合(limit 属性が指定されており、属性値の回数分 case が実行済である場合)は、制限がかかっている case 以外が case_id に指定されている condition から選択されます。回数制限の詳細は 4.8 case タグの章を参照ください。

4.10 control タグ

アプリとの連携の際に利用します。

本タグを含む action が実行されるタイミングで、指定されたアプリに対してイベントが発行されます。

表 4-9 概要

項目	説明
書式	<control target ="…" function="…">~</control> ,<control target="…" function="…"/>
親要素	<action>
子要素	<data>
属性	target(必須),function(必須)

4.10.1 target 属性

HVML を登録したアプリの package 名、または"home"を指定してください。

"home"はホーム用 HVML でアプリを起動するために使用します。詳細は参考資料[3]を参照してください。

4.10.2 function 属性

発行するイベント名を記載します。

4.10.3 使用例

```
<topic id="0010" listen="false">
  <action index="1">
    <control target="jp.co.sharp.android.rb.sample" function="sample_exec">
      <data key="id" value="test"/>
    </control>
  </action>
</topic>
```

4.10.4 注意事項

子要素として data タグを記述することで、target に対してより詳細な情報を連携できます。

アプリへの通知の詳細に関しては参考資料[2]および[3]を参照ください。

4.11 data タグ

control タグの子要素として使用し、Key-Value のペアで target で指定したアプリにパラメータを引き渡すことができます。アプリ側での受け取り方法は参考資料[3]を参照ください。

表 4-10 概要

項目	説明
書式	<data key ="…" value="…"/>
親要素	<control>
子要素	なし
属性	key(必須),value(必須)

4.11.1 key 属性

任意の文字列を指定します。一つの親要素に対して一意になるよう指定してください。

4.11.2 value 属性

key に対応する値を指定します。

4.11.3 使用例

```
<topic id="0010" listen="false">
  <action index="1">
    <control target="jp.co.sharp.android.rb.sample" function="sample_exec">
      <data key="id1" value="data1"/>
      <data key="id2" value="data2"/>
    </control>
  </action>
</topic>
```

4.12 description タグ

HVML ファイル自体の説明を記述する用途で利用してください。本タグは HVML 実行時の動作に影響しません。

表 4-11 概要

項目	説明
書式	<description>~</description>
親要素	<head>
子要素	なし
属性	なし

4.12.1 使用例

```
<hvml version="2.0">
  <head>
    <description>テスト用シナリオ</description>
    …
  </head>
  <body>
    …
  </body>
</hvml>
```

4.13 emotion タグ

ロボホンが嬉しそうにしゃべる/悲しそうにしゃべるといった、発話する際の感情(TTS のパラメータ)を設定します。本タグを指定していない発話は、type="happiness"、level="1"が設定されます。

表 4-12 概要

項目	説明
書式	<emotion type ="…" ~>~~</emotion>
親要素	<speech>
子要素	なし
属性	type(必須),level

4.13.1 type 属性

感情の種別を指定します。以下の 3 種類から指定可能です。

- happiness ： 嬉しそうに発話します。
- sadness ： 悲しそうに発話します。
- anger ： 怒った声で発話します。

4.13.2 level 属性

type で指定した感情の度合いを指定します。1～4 の 4 段階で指定可能で、数字が大きい方が感情値が高くなります。指定なしの場合は 1 と扱われます。

4.13.3 使用例

```
<topic id="0010" listen="false">
  <action index="1">
    <speech><emotion type="happiness" level="2">おはよう</emotion></speech>
  </action>
</topic>
```

4.13.4 注意事項

一つの speech タグ内で複数の emotion タグを挿入することは可能ですが、入れ子とならないようにしてください。

4.14 head タグ

HVML ファイル自体に付随する情報、HVML が選択されるために必要とする情報を記載します。

表 4-13 概要

項目	説明
書式	<head>~</head>
親要素	<hvml>
子要素	<producer>,<scene>,<situation>,<accost>,<version>
属性	なし

4.15 hvml タグ

hvml ファイルにおけるルート要素です。

表 4-14 概要

項目	説明
書式	<hvml version ="…">~</hvml>
親要素	なし
子要素	<head>,<body>
属性	version(必須)

4.15.1 version 属性

"2.0"を指定してください。

4.16 memory タグ

情報を記憶させたい場合に使用します。本タグで保存した情報は、後の対話で変数として記載することで利用することができます。変数についての詳細は後述します。

表 4-15 概要

項目	説明
書式	<memory key ="…" type="…" ~/>
親要素	<action>
子要素	なし
属性	type(必須),key(必須),value,operation

4.16.1 type 属性

“temporary","permanent"のいずれかを指定してください。temporary を指定した場合、対話(HVML の実行)が終了した時点で情報は破棄されます。対話の終了については 3.HVML の実行の章を参照ください。 permanent 指定では、明示的に削除するまで情報は保持されます。

4.16.2 key 属性

key を指定します。key は保存した情報を呼び出す際に使用します。type="permanent"の場合は衝突を避けるため、package 名から始まる文字列を指定してください。

【key 名記載フォーマット】

```
key="jp.co.sharp.rb.sample.sampleKey"
```

4.16.3 value 属性

記憶する値を指定します。operation="set"の場合は本属性は必須です。本パラメータには演算子を使用することが可能です。演算子を使用することで使用例に記載するようなカウンタとして利用することが可能です。演算子を利用する場合、「()」で囲んで記述してください。
変数および演算子を記述する以外では 6.7 禁止文字の章に記載の文字は使用しないでください。

【変数と演算子の記述例】

```
<memory type="temporary" key="loop_count" value="(${memory_t:loop_count} + 1)"/>
```

4.16.4 operation 属性

操作種別を指定します。set,delete のいずれかを指定してください。

・　set　：　上書き登録

・　delete　：　削除

指定なしの場合は set として扱われます。

4.16.5 使用例

```
<topic id="0001" listen="false">
  <action index="1">
    <memory type="permanent" key="jp.co.sharp.rb.sample.sampleKey" value="1"/>
    <memory type="temporary" key="loop_count" value="0"/>
  </action>
  <next href="#0010" type="default"/>
</topic>
<topic id="0010" listen="false">
  <rule>
    <condition case_id="1">${memory_t:loop_count} eq 0</condition>
    <condition case_id="2">${memory_t:loop_count} eq 1</condition>
  </rule>
  <case id="1">
    <action index="1">
      <speech>え、何？もう一回言ってね</speech>
    </action>
    <action index="2">
      <memory type="temporary" key="loop_count" value="(${memory_t:loop_count} + 1)"/>
    </action>
    <next href="#0020" type="default"/>
  </case>
  <case id="2">
    <action index="1">
      <speech>今度また教えてね</speech>
    </action>
    <next href="#0030" type="default"/>
  </case>
</topic>
```

上記例では、topic id="0001"で loop_count の memory に 0 をセットし、topic id="0010"の rule で loop_count によって処理を分けます。一回目は case id="1"が実行され、action index="2"で loop_count がカウントアップされるため、次の topic id="0010"では case id="2"が実行されます。

4.16.6 注意事項

memory タグは一つの action タグ内に複数記述できます。また、speech など他タグと合わせて記述することも可能です。また、memory で保存した情報は、同一 topic 内で利用することはできません。次 topic に遷移して以降利用できるようになります。

4.17 next タグ

topic 実行後、対話を終了せずに別の topic に遷移させたい場合に遷移先を記載します。

表 4-16 概要

項目	説明
書式	<next href ="…" type="…"/>
親要素	<topic>,<case>
子要素	なし
属性	href(必須),type(必須)

4.17.1 href 属性

遷移先の HVML ファイル名と topic id を指定します。遷移先が別ファイルの topic の場合は「HVML ファイル名#topic id」のフォーマットで、同一ファイル内の topic 遷移の場合はファイル名を省略し、「#topic id」のフォーマットで記載してください。

4.17.2 type 属性

type="default"を指定してください。

4.17.3 使用例

```
<hvml version="2.0">
  <head>
    …
  </head>
  <body>
    <topic id="0001" listen="false">
      <action index="1">
        <speech>こんにちは</speech>
      </action>
      <next href="#0002" type="default" />
    </topic>
    <topic id="0002" listen="false">
      <action index="1">
        <speech>今日もいい天気ですね</speech>
      </action>
    </topic>
    …
  </body>
</hvml>
```

4.17.4 注意事項

listen="false"が指定された topic では、action が実行された直後に href の指定先に遷移します。
listen="true"が指定された topic では、action 実行後、10 秒間ユーザからの発話が無ければ href の指定先に遷移します。

4.18 producer タグ

HVML ファイルの発行元を表します。要素の内容にＨＶＭＬファイルの登録元のアプリの package 名を記載してください。

表 4-17 概要

項目	説明
書式	<producer>~</producer>
親要素	<head>
子要素	なし
属性	なし

4.18.1 使用例

```
<hvml version="2.0">
  <head>
    <producer>jp.co.sharp.android.sampleProducer</producer>
    …
  </head>
    …
</hvml>
```

4.18.2 注意事項

HVML ファイルには必ず本タグを記述ください。また、一つのファイルに本タグを複数記述しないでください。

4.19 rule タグ

topic 実行時に、その時の条件によって実行するアクション等を変えたい場合に使用します。rule タグは必ず condition タグ(条件を記述します)、case タグ(実行するアクション等を記述します)とセットで使用してください。

表 4-18 概要

項目	説明
書式	<rule>~</rule>
親要素	<topic>
子要素	<condition>
属性	なし

4.19.1 使用例

```
<rule>
    <condition case_id="1" weight="3">${Hour} ge 20</condition>
    <condition case_id="2" weight="1">${Hour} ge 22</condition>
    <condition case_id="3" weight="1">${Hour} ge 22</condition>
    <condition case_id="4" priority="20">true</condition>
 </rule>
```

4.19.2 注意事項

子要素として、必ず一つ以上の condition タグを記述してください。

4.20 scene タグ

HVML が実行されるシーンを設定します。
シーンはロボホンの内部状態として保持されているパラメータであり、実行する HVML を絞り込む用途で使用します。
ホーム画面ではシーンとして"home"が設定されます。アプリ起動中は各アプリがシーンを設定します。アプリからのシーン指定方法は参考資料[3]を参照ください。またシーンは同時に複数設定することが可能です。
HVML 実行時、記載されている scene タグの value 値が現在設定されているシーンと一致している HVML が選択されます。HVML 実行時の詳細は 3.HVML の実行の章を参照ください。

シーンに合致しない HVML が選択されることはありませんが、他 HVML の a,next タグによる topic 遷移の場合は scene タグを参照しません。(明示的に HVML ファイル名および topic id を指定しているため、シーンの判定は行わず a,next タグの href 属性に記載の遷移先が実行されます。)
この場合も自アプリ外の HVML ファイルの実行はできません。

表 4-19 概要

項目	説明
書式	<scene value="…"/>
親要素	<head>
子要素	なし
属性	value(必須)

4.20.1 value 属性

ホーム画面で実行させる HVML に関しては、"home"を指定してください。
ホーム画面以外の HVML について、実行させるシーン名を指定してください。

4.20.2 使用例

```
<hvml version="2.0">
  <head>
    <scene value="jp.co.sharp.android.testtest"/>
    <scene value="jp.co.sharp.android.testtest.SettingActivity"/>
    …
  </head>
    …
</hvml>
```

4.20.3 注意事項

HVML ファイルには必ず本タグを記述ください。複数記述することも可能ですが、"home"を指定するファイルに関しては他の scene を設定しないでください

4.21 situation タグ

実行する topic を選択するための契機および条件を記述します。要素の内容には条件式を記述し、条件式の結果が真(true)であれば合致と判定されます。条件の詳細な記述方法に関しては後述の 5.演算子、6.変数の章を参照してください。
常に真となる条件を記載したい場合、要素の内容に true と記載してください。
属性に関しては親要素によって記述ルールが異なります。詳細は後述します。

表 4-20 概要

項目	説明
書式	<situation trigger ="…" ~>~~</situation>
親要素	<head>,<a>
子要素	なし
属性	trigger(必須),value,topic_id,priority

4.21.1 trigger 属性

"user-word","env-event"のいずれかを指定してください。

- env-event ： value 属性で指定された環境検知イベントを契機に条件式を評価します。
- user-word ： ユーザ発話を契機に条件式を評価します。

条件式を評価した結果が true であれば指定した topic を実行します。

4.21.2 value 属性

環境検知イベントの種別を記載してください。
環境検知イベントの種別と説明については 7.環境検知イベントの章を参照してください。
trigger="user-word"の場合は不要です。

4.21.3 topic_id 属性

head タグ中に situation タグを記述する場合は本属性を記載してください。
situation の条件に合致した場合に、遷移する topic の id を指定します。
a タグ中に situation を記述するケースでは本属性は不要で、遷移先は a タグの href 属性に従います。

4.21.4 priority 属性

HVML 動時の優先度を指定します。条件に合致する situation が複数ある場合は、priority に従って実行する topic が決

定されます。
a タグ内で priority を指定しない場合は、その時点の priority が引き継がれます。
scene="home"が指定されている HVML では 78～84、それ以外の HVML は 61～90 の値を指定してください。数字が小さい方が優先されます。
scene="home"が指定されている HVML では本属性は必須です。それ以外の HVML で本属性を記載しない場合は以下のようになります。

- head の situation タグ ： 75 と扱われます。
- a タグ内の situation タグ ： a タグが含まれる topic 実行時の priority(topic が実行された際の accost または situation に記載の priority)と同じ値と扱われます。

4.21.5 使用例

```
<hvml version="2.0">
  <head>
    <situation topic_id="0001" trigger="user-word" priority="78">
      誕生日 in ${Lvcsr:Basic} and 覚えて in ${Lvcsr:Basic}
    </situation>
  …
  </head>
    …
</hvml>
```

4.21.6 注意事項

本タグは一つの親要素内に複数記述することが可能です。

4.22 speech タグ

本タグの内容に文を記述することで、TTS を使用してロボホンに発話させることができます。

表 4-21 概要

項目	説明
書式	<speech>~</speech>
親要素	<action>
子要素	<emotion>,<wait>
属性	なし

4.22.1 使用例

```
<topic id="0001" listen="true">
  <action index="1">
      <speech>おはよう</speech>
      <behavior type="normal" id="0x040e01"/>
  </action>
</topic>
```

4.22.2 注意事項

本タグの内容に HVML で規定している以外のタグを挿入しないでください。

また一つの action に対して、本タグと behavior など他のタグと併記可能です。ただし、本タグ自体は一つの action に複数記述しないでください。

4.23 topic タグ

HVML の body 部を構成する要素で、HVML 内で実行したい処理をブロック化するためのタグです。id を指定することで、situation タグ、accost タグ等から実行することができます。

表 4-22 概要

項目	説明
書式	<topic id="…" listen="…" ~>~~</topic>
親要素	<body>
子要素	<rule>,<case>,<action>,<a>,<next>
属性	id(必須),listen(必須),listen_ms,dict

4.23.1 id 属性

situation,accost タグで遷移する topic を指定する際に利用します。
任意の文字列が指定可能ですが、HVML ファイル内で topic の id が重複しないようにしてください。

4.23.2 listen 属性

ユーザからの発話を受け付けるかどうかを指定します。"true","false"のいずれかを指定してください。
“true"を指定した場合、topic 内のすべての action を実行した後、発話を待ち受けます。ユーザ発話を受けると、a タグの記述に従って次の topic に遷移します。
“false"を指定した場合、action 実行後に next タグがあれば遷移し、なければ対話を終了します。

4.23.3 listen_ms 属性

listen="true"の場合に、ユーザ発話を受け付ける最大時間を msec 単位で指定します。本属性を記載しない場合は 10 秒(10000msec)が指定されます。
listen="false"が指定されている場合、本パラメータは無視されます。
本属性の指定時間以内にユーザ発話が無かった場合、next タグがあれば遷移し、なければ対話を終了します。

4.23.4 dict 属性

肯定/否定の聞き取りを行いたい場合に使用します。その場合は“Reply"を指定してください。
肯定/否定の聞き取りの詳細は参考資料[3]を参照ください。

4.23.5 使用例

```
<hvml version="2.0">
  <head>
    …
  </head>
  <body>
    <topic id="0001" listen="true">
      <action index="1">
        <speech>今日はいい天気ですか？</speech>
      </action>
      …
    </topic>
    <topic id="0002" listen="false">
      …
    </topic>
    …
  </body>
</hvml>
```

4.24 version タグ

HVML ファイル管理用のバージョン情報です。HVML の登録/更新時に利用します。

表 4-23 概要

項目	説明
書式	<version value="…"/>
親要素	<head>
子要素	なし
属性	value(必須)

4.24.1 value 属性

バージョン番号を記載します。HVML ファイルを登録する際、既に同一ファイル名の HVML が登録されていた場合、value 値が登録済みの HVML の値より大きい場合のみ HVML を上書き登録します。

4.24.2 使用例

```
<hvml version="2.0">
  <head>
    <version value="1.0"/>
    …
  </head>
    …
</hvml>
```

4.24.3 注意事項

value 値には正の整数または小数を記載してください。

【○正しい version の記載】

```
<version value="1.0"/>
<version value="2.01"/>
```

【×誤った version の記載】

```
<version value="1.0.1"/>
```

4.25 wait タグ

ロボホンの発話中に間を持たせたい場合に利用します。

表 4-24 概要

項目	説明
書式	<wait ms="…"/>
親要素	<speech>
子要素	なし
属性	ms(必須)

4.25.1 ms 属性

時間を msec 単位で指定します。

4.25.2 使用例

```
<topic id="0010" listen="false">
  <action index="1">
    <speech><emotion type="happiness" level="2">おはよう。</emotion><wait ms="300"/><emotion type="happiness" level="2">今日もいい天気だね</emotion></speech>
  </action>
</topic>
```

4.25.3 注意事項

一つの speech タグ内で複数の wait タグを挿入することも可能です。

5. 演算子

本章では HVML を構成する要素の一つである、演算子について記載します。値の型に関する詳細は 6.変数の章を参照ください。

5.1 演算子一覧

演算子一覧を記載します。

- eq (=)
- neq (≠)
- gt (>)
- ge (>=)
- lt (<)
- le (<=)
- and
- in
- outof
- include
- near
- +
- -
- *
- /

5.1.1 eq/neq

すべての型に対して使用可能です。
左辺と右辺が一致しているか、もしくは一致していないかを判別します。

5.1.2 gt

数値型もしくは日時型に対して使用可能です。
左辺が右辺より大きい場合に真となります。

5.1.3 ge

数値型もしくは日時型に対して使用可能です。
左辺が右辺以上の場合に真となります。

5.1.4 lt

数値型もしくは日時型に対して使用可能です。
左辺が右辺より小さい場合に真となります。

5.1.5 le

数値型もしくは日時型に対して使用可能です。
左辺が右辺以下の場合に真となります。

5.1.6 and

複数の演算結果を結合する際に使用します。and で繋がれた演算結果がすべて真である場合に、条件式全体として真となり、一つでも偽になるものがあれば偽と扱われます。

5.1.7 in

左辺は文字列型、右辺は文字列型またはリスト型を指定してください。
右辺が文字列型の場合、左辺の文字列が右辺の文字列中に含まれているなら真、
右辺がリスト型の場合、右辺の要素に左辺と一致するものがあれば真となります。

5.1.8 outof

左辺は文字列型、右辺は文字列型またはリスト型を指定してください。
右辺が文字列型の場合、左辺の文字列が右辺の文字列中に含まれていないなら真、
右辺がリスト型の場合、右辺の要素に左辺と一致するものがなければ真となります。

5.1.9 include

左辺は文字列型、右辺はリスト型または文字列型を指定してください。
右辺がリスト型の場合、右辺の要素のいずれか一つでも左辺に含まれていれば真、
右辺が文字列型の場合、右辺の文字列が左辺の文字列中に含まれていれば真となります。

5.1.10 near

左辺、右辺共に文字列型を指定してください。
左辺と右辺の文字列の近似度がある一定の範囲内であれば（つまり２つの文字列が近ければ）真となります。
主にユーザ発話を条件式に記載する際に、ユーザ発話をある程度のゆらぎを持たせて合致させたい場合に使用します。
ただし最低でも４文字以上の文字列を比較する場合のみ利用してください。短い文字列の比較では「テンキ」と「デンキ」　など、１文字違うと全く違う意味になるうえ、近似度が低くなり事実上成立しない条件となります。

5.1.11 + , - , * , /

数値型の四則演算用の演算子になります。それぞれ左辺、右辺の加減乗除を行います。
+、-に関しては、日時型での利用も可能です。

5.2 演算子を利用できる箇所

HVML 中で変数を記述できる箇所を以下に記載します。

- situation タグの内容
- condition タグの内容
- data タグの value 属性
- memory タグの value 属性

6. 変数

本章では HVML を構成する要素の一つである、変数について記載します。
変数は日時情報や、音声認識の結果、センサー等で取得できる環境変数など、動的な内容を HVML ファイル内に記述する際に用います。また memory タグで記憶されている情報についても変数を用いて HVML 内で利用できます。
変数は実際に HVML を実行する際に、取得した値に置き換えた後にタグの解釈が行われます。

6.1 変数のフォーマット

変数は以下のような形式で記載してください。

【変数フォーマット】

```
${jp.co.sharp.rb.sample:variable}
```

上記例では、「jp.co.sharp.rb.sample:variable」が変数名になります。
また、変数の中には引数を指定できるものがあります。引数を持つ変数は以下の形式で記載してください。

【引数を持つ変数フォーマット】

```
${ jp.co.sharp.rb.sample:variable (arg1,arg2)}
```

「arg1」が第一引数、「arg2」が第二引数、…になります。

6.2 変数一覧

変数の一覧を記載します。

【ユーザ発話変数】

- Lvcsr:Basic
- Lvcsr:Kana
- Local_Reply:GLOBAL_REPLY_YES
- Local_Reply:GLOBAL_REPLY_NO
- Local:WORD_APPLICATION
- Local:WORD_APPLICATION_FREEWORD

【時制取得用変数】

- Year
- Month
- Day
- Date
- DayOfWeekJp

- Hour
- Minute
- Second
- Time
- Now
- CurrentDate
- CurrentYearDate
- CurrentTime

【時制判定用変数】

- InDateRange
- InTimeRange
- IsWithinTime
- DiffDate

【型変換用変数】

- SizeOf
- Select
- YearOf
- MonthOf
- DayOf
- HourOf
- MinuteOf
- SecondOf

【個人情報を扱う変数】

- resolver:contacts:robot_name
- resolver:contacts:user:birthday
- resolver:contacts:user:name_for_speech

個人情報とは、利用規約に記載の「2.センサー情報」に該当します。

【その他変数】

- Rand
- env:robot_iniboot_date
- resolver:charger_connected
- resolver:earphone_connected
- memory_p:manner_mode
- resolver:pose
- resolver:speech_ok
- resolver:motion_ok

- resolver:ok_id
- env:telephony_support
- env:projector_support
- env:sitmodel
- env:robohon_generation

6.2.1 Lvcsr:Basic/Lvcsr:Kana

戻り値：文字列型

直前のユーザ発話の認識結果が格納されます。
Lvcsr:Basic は発話内容をかな漢字の文字列で、Lvcsr:Kana は発話内容をカタカナで返します。
ネットワークエラー等により、認識結果が取得できなかった場合は、「ＶＯＩＣＥＰＦ＿ＥＲＲ＿ｘｘ」という文字列が格納されます（全角、xx 部分はエラー毎のエラー理由文字列です）。
具体的な認識エラー処理の記述方法は参考資料[3]を参照ください。

6.2.2 Local_Reply:GLOBAL_REPLY_YES/Local_Reply:GLOBAL_REPLY_NO

戻り値：文字列型

直前のユーザ発話の認識結果が肯定/否定の意図の場合に格納されます。
ただし、事前に dict="Reply"を指定しておく必要があります。
肯定/否定の聞き取りの詳細は参考資料[3]を参照ください。

6.2.3 Local:WORD_APPLICATION/Local:WORD_APPLICATION_FREEWORD

戻り値：文字列型

直前のユーザ発話文言が登録したアプリ起動文言に一致した場合、一致した文言が格納されます。
本変数はホーム用の HVML でのみ使用します。また、演算子は eq を使用してください。詳細は参考資料[3]を参照ください。

6.2.4 Year/Month/Day

戻り値：数値型

現在の年、月、日をそれぞれ整数値で返します。

6.2.5 Date

戻り値：日時型

現在の年月日を返します。

6.2.6 DayOfWeekJp

戻り値：文字列型

現在の曜日を返します。文字列は月/火…で「曜日」は含みません。

6.2.7 Hour/Minute/Second

戻り値：数値型

現在の時、分、秒をそれぞれ整数値で返します。

6.2.8 Time

戻り値：日時型

現在の時分秒を返します。

6.2.9 Now

戻り値：日時型

現在日時を返します。

6.2.10 CurrentDate

戻り値：数値型

年内通算日数。1 月 1 日を 0 として、0~365 の値を返します。

6.2.11 CurrentYearDate

戻り値：数値型

1970/01/01 からの通算日数。1970/01/01 を 0 として、経過日数を整数値で返します。

6.2.12 CurrentTime

戻り値：数値型

日内通算秒数。0~86399 の値を返します。

6.2.13 InDateRange

戻り値：真偽型

引数として、開始月、開始日、終了月、終了日を指定してください(すべて数値型)。現在月日がその間にある場合は true を返します。開始、終了どちらかのみ指定したい場合は、指定なしの月と日に-1 を指定してください。

6.2.14 InTimeRange

戻り値：真偽型

引数として、開始日内秒数、終了日内秒数を指定してください(すべて数値型)。現在時刻がその間にある場合は true を返します。開始、終了どちらかのみ指定したい場合は、指定なしの秒数に-1 を指定してください。

6.2.15 IsWithinTime

戻り値：真偽型

引数として、1970/01/01 からの通算日数、日内秒数、差分秒数を指定してください(すべて数値型)。指定日、秒、からの経過時間が差分秒数以内だったら true を返します。

6.2.16 DiffDate

戻り値：数値型

引数として、基準日時(日時型)、比較日時(日時型)を指定してください。基準日時に対して比較日時があと何日かを返します。

6.2.17 SizeOf

戻り値：数値型

引数としてリスト型を指定すると、リストの要素の個数を返します。

6.2.18 Select

戻り値：文字列型

第一引数にリスト型、第二引数に要素番号を指定することで、指定の要素の文字列のみを取得できます。

6.2.19 YearOf/MonthOf/DayOf/HourOf/MinuteOf/SecondOf

戻り値：数値型

それぞれ日時型の値を引数に指定すると、引数の年のみ/月のみ/日のみ/時のみ/分のみ/秒のみを返します。

6.2.20 resolver:contacts:robot_name

戻り値：文字列型

ロボホンの呼び方を返します。初期値は「ろぼほん」です。

6.2.21 resolver:contacts:user:birthday

戻り値：日時型

オーナーの誕生年月日を返します。誕生年月日が設定されていない場合は null を返します。

6.2.22 resolver:contacts:user:name_for_speech

戻り値：文字列型

オーナー名を返します。ロボホンがオーナーを呼ぶ際には本変数を利用してください。

6.2.23 Rand

戻り値：数値型

引数を 0~99 で指定可能で、引数を指定した場合は 0 以上引数未満の整数値をランダムで返します。

引数なしの場合は 0～99 の整数値をランダムで返します。

6.2.24 env:robot_iniboot_date

戻り値：日時型

初期設定を実施した日（ロボホンと出会った日）を返します。

6.2.25 resolver:charger_connected

戻り値：真偽型

充電/USB ケーブル接続している場合に true/それ以外は false を返します。

6.2.26 resolver:earphone_connected

戻り値：真偽型

イヤホン接続している場合に true/それ以外は false を返します。

6.2.27 memory_p:manner_mode

戻り値：文字列型

マナーモードの状況に応じて以下を返します。

- off　：　非マナーモード
- on_partial　：　マナーモード　(会話のみ OFF)
- on　：　マナーモード　(すべて OFF)

マナーモードに関する詳細は参考資料[3]を参照ください。

6.2.28 resolver:pose

戻り値：文字列型

ロボホンの姿勢
(stand,hand_stand,sit,hand_sit,mobile,hand_mobile,back,belly,hand_phone,projector,cradle,immobile)を返します。
各姿勢に関しては、参考資料[3]を参照ください。

6.2.29 resolver:speech_ok

戻り値：文字列型

ユーザの発話などに対して、ロボホンが返事する際の発話内容を返します。引数には${resolver:ok_id}を指定してください。
詳細は参考資料[3]を参照ください。

6.2.30 resolver:motion_ok

戻り値：文字列型

ユーザの発話などに対して、ロボホンが返事する際のモーション ID を返します。引数には${resolver:ok_id}を指定してください。
詳細は参考資料[3]を参照ください。

6.2.31 resolver:ok_id

戻り値：数値型

resolver:speech_ok と resolver:motion_ok の引数に指定する用途で利用してください。
詳細は参考資料[3]を参照ください。

6.2.32 env:telephony_support

戻り値：真偽型

ロボホン(3G・LTE)モデルの場合に true/それ以外は false を返します。※SIM の違いによる電話機能有無(通話用 SIM とデータ SIM)は判別できません。あくまでもロボホン自体の電話機能有無のみ判別できます。

6.2.33 env:battery:level

戻り値：数値型

ロボホンの電池残量の値を返します。0 – 100 (%)

6.2.34 env:projector_support

戻り値：真偽型

プロジェクター機能のあるロボホン(モデル番号：SR01MW/SR02MW)の場合に true/それ以外は false を返します。

6.2.35 env:sitmodel

戻り値：真偽型

ロボホンライト場合に true/それ以外は false を返します。

6.2.36 env:robohon_generation

戻り値：数値型

ロボホンの世代(1 or 2)を返します。
ロボホン第１世代：SR01MW/SR02MW
ロボホン第２世代：SR03M/SR04M/SR05M

6.3 memory 変数

保存した記憶の情報を変数として利用することができます。
以下のように記載してください。

【type="temporary"で保存した変数】

```
${memory_t: var}
```

【type="permanent"で保存した変数】

```
${memory_p:jp.co.sharp.rb.sample.var}
```

上記例の「var」「jp.co.sharp.rb.sample.var」はそれぞれ保存時の key 値を指定してください。
上記形式で記載した結果として、保存されている value 値が返ります。

記憶の保存は、hvml ファイル内に memory タグを記述する方法とアプリから直接保存する方法があります。詳細は参考資料[3]を参照してください。

6.4 変数の追加

前述の変数以外に、変数を定義し、HVML 実行時にアプリへの callback 通知で値を設定することができます。
変数名は「アプリパッケージ名:xx」としてください。

【追加変数フォーマット】

```
${jp.co.sharp.rb.sample:variable}
```

HVML が実行される際に、アプリに対し、VoiceUIListener#onVoiceUIResolveVariable()が通知されますので、値を設定すると HVML に反映されます。
詳細は参考資料[2]および[3]を参照してください。

6.5 変数型

変数の型は string(文字列)型、float(数値)型、boolean 型(真偽型、true/false)および、変数に値が入っていない/変数が存在しないことを示す"null"に分類されます。また、一部特殊な演算を可能とするための型を定義しています。詳細は次節以降を参照ください。
それぞれの値の型は HVML 内で自動的に判別されます。

6.5.1 日時型

日時専用のフォーマットで、以下の 3 タイプに分類されます。

- 日時全て ： YYYY/MM/DD_hh:mm:ss （例：2015/08/18_16:26:00, 2015/8/18_16:26:00）
- 日付のみ ： YYYY/MM/DD （例：2015/08/18, 2015/8/18）
- 時間のみ ： hh:mm:ss （例：16:26:00）

日時型同士で、+, -, eq, neq, gt, lt, ge, le による演算が可能です。
+,-を使う場合は、左辺に上記日付型を書き、右辺は以下のフォーマットで記述してください。(***X*** は数値)

- ***X***year
- ***X***mon
- ***X***day
- ***X***hour
- ***X***min
- ***X***sec

また、特定の箇所をワイルドカード指定することも可能です。ワイルドカードを指定する際は「*」と指定してください。
下記に簡単なサンプルを記載します。

【日時型を使った HVML サンプル】

```
<hvml version="2.0">
    …
  <body>
    <topic id="date" listen="false">
      <rule>
        <condition case_id="1" priority="1">${Now} eq 2016/01/01</condition>
        <condition case_id="2" priority="2">${Now} eq */1/1</condition>
        <condition case_id="3" priority="2">${Now} ge */7/1</condition>
      </rule>
      <case id="1">
        <action index="1">
          <speech>今日は２０１６年のお正月だよ</speech>
        </action>
      </case>
      <case id="2">
        <action index="1">
          <speech>今日はお正月だよ</speech>
        </action>
      </case>
      <case id="3">
        <action index="1">
          <speech>今年ももう後半だね</speech>
          <memory type="temporary" key="Date" value="(${Date} + 15day)"/>
        </action>
        <next href="#later" type="default"/>
      </case>
    </topic>
    <topic id="later" listen="false">
      <action index="1" >
        <speech>15 日後は${memory_t:Date}だよ</speech>
      </action>
    </topic>
  </body>
</hvml>
```

現在日が 2016 年 1 月 1 日なら case_id 1 が、2016 年以外の 1 月 1 日なら case_id 2 が、7 月 1 日～12 月 31 日なら case_id 3 が実行されます。

6.5.2 リスト型

複数の値を格納するための型で、[abc,ABC,xyz,XYZ]のような形式で記述できます。${SizeOf}変数でリスト内の要素数の取得や、${Select}変数にてリスト内の指定の要素のみを取得することができます。

6.6 変数を使用できる箇所

HVML 中で変数を記述できる箇所は以下に記述します。

- situation タグの内容
- speech タグの内容
- memory タグの value 属性
- behavior タグの type 属性/id 属性
- control タグの target 属性/function 属性
- data タグの value 属性
- condition タグの内容
- a タグの href 属性
- next タグの href 属性

6.7 禁止文字

以下の記号は条件判定文の解釈等で使用しているため、基本的にアプリ(java)から API で設定する変数の値に含めたり、memory タグの value に設定する値として使用しないでください。

- , (カンマ)
- (半角スペース)
- : (コロン)
- [(開きブラケット)
-] (閉じブラケット)
- ((開きバーレン)
-) (閉じバーレン)
- { (開きブレイス)
- } (閉じブレイス)
- ¥ (バックスラッシュ/円記号)

ただし以下のケースに限り使用することが可能です。

1. 配列(リスト型)としてカンマ、開きブラケット、閉じブラケットを使用するケース

【配列サンプル】

```
[a1,a2]
```

ただし、この際も半角スペースは使用できないので注意してください。

【×誤った配列サンプル】

```
[a1, a2]
```

2.　日時型としてコロンを使用するケース

【日時型サンプル】

```
2015/08/18_16:26:10
```

3.　memory タグの value 属性に「()」内の演算子や変数として記述するケース

詳細は 4.16 memory タグの章を参照ください。

6.8　変数を situation に記載する際の注意点

head の situation trigger="user-word"に条件式を記載する際は、以下の優先度順で記載してください。

ユーザ発話変数(Lvcsr:Basic/Lvcsr:Kana)　> ${Hour} > その他の変数(memory_p など)

【○正しい situation の記載】

```
<situation trigger="user-word" topic_id="0001">
こんにちは in ${Lvcsr:Basic} and ${Hour} ge 12 and {memory_p:jp.co.sharp.rb_hoge} eq 0
</situation>
```

【×誤った situation の記載】

```
<situation trigger="user-word" topic_id="0001">
{memory_p: jp.co.sharp.rb_hoge} eq 0 and こんにちは in ${Lvcsr:Basic} and {Hour} ge 12
</situation>
```

誤った記載例の通り実装しても動作はしますが、シナリオ選択時の速度に影響しますので注意してください。

7. 環境検知イベント

7.1 環境検知イベント一覧

本章では環境検知イベントについて記載します。

環境検知イベントはシナリオ実行の契機として利用するために、ロボホンに状態変化がある際に通知されるイベントです。

表 7-1 環境検知イベント一覧

イベント名	内容
put_down_standup	置かれた（垂直）。 ロボホンの胴体部分が垂直方向となっている必要があります。立ち状態、座り状態、充電台に乗っている状態でしか発生しません。
charge_start	充電開始された
charge_stop	充電停止された
low_batt	電池残量が低下した ロボホンでは電池残量 20%以下を低下状態としています。
stop_alarm	アラームが停止された
earphone_connect	イヤホン接続された
location	現在の位置情報が更新された場合に通知されます
put_down_back	水平に置いた
shake	振られた
upside_down	逆さにされた

7.2 使用例

```
<hvml version="2.0">
  <head>
      <situation topic_id="0001" trigger="env-event" value="shake" priority="75">true</situation>
  …
  </head>
    …
</hvml>
```