CHAPTER 9

レポート マクロ リファレンス

Prime Performance Manager のレポート インタフェースは、レポートで使用できるたくさんの定義済みレポート マクロを提供します。マクロの呼び出し方法には、次の 2 種類があります。

1. object.macro (*arg1*, *arg2*, *arg3*, etc.)
2. macro (*object*, *arg1*, *arg2*, *arg3*, etc.)

この章で参照するトピックでは、2 番目の方式の構文が提供されます。

引数が角カッコ（[]）で囲まれている場合、その引数は省略可能であることを意味します。

引数が特定の型であることが示されている場合（オブジェクトは文字列型）、その引数は次のもので置き換えることもできます。

- 同じ型を返すマクロ / アルゴリズム

または

- 数値型 / 文字列型の場合は数値 / 文字列値

マクロ

ADD

構文

ADD (object, arg1, arg2)

マクロの説明

オブジェクトが収集のインスタンスの場合、このマクロはオブジェクト コレクションに arg1 のデータが追加します。オブジェクトが Map のインスタンスの場合、このマクロはキー値（arg1-arg2）のペアをオブジェクトに追加します。失敗した場合は、ヌルを返します。

- オブジェクトは、オブジェクト インスタンスまたは収集インスタンスのいずれかです。
- arg1 は、キーまたは値のいずれかです。
- arg2 は値です。

例：

```
row.add("totalInPkts",totalInPkts);
```

BITS

構文

BITS (object, arg)

マクロの説明

arg で指定されたオブジェクトの位置が 1 の場合に true を返し、そうでない場合は false を返します。

- object は、ビット文字列です
- arg は、一致するビット位置の整数です

例：

```
status = cfmMdiMetricsValid.bits(0);
```

BOTTOMN

構文

BOTTOMN (object, arg1, arg2)

マクロの説明

昇順ソート キーでソートされた object の最下部 n（arg2）行を返す

- object はテーブル
- arg1 はソートする列
- arg2 は取得するレコード数（n）
- ProcessDBSummary セクションの Filter セクション内で使用される場合、rows は、一般に、ProcessDBSummary の実行が完了したときに設定されるオブジェクトとして使用可能な暗黙の変数です。

例：

```
rows = rows.bottomN("ProcessCPUUtil5mAvg", 5);
```

CONTAINS

構文

CONTAINS(arg1,arg2)

マクロの説明

このマクロは、arg2 が arg1 に含まれているかどうかを確認します。含まれている場合、true を返し、そうでない場合は false を返します。

- arg1 および arg2 はマップ、収集、または文字列である可能性があります。

例 :

```
if(cmStatusName.contains("online"),true,false);
```

COUNT

構文

COUNT(object)

マクロの説明

このマクロは、グループ化されたデータの行およびこれらの行のカウント数を更新します。カウント数を返します。

- オブジェクトはカウントされる必要のあるグループ化されたデータで構成されます。

例

```
cmStatusGroups.count();
```

CSVPOLL

構文

CSVPOLL (arg1, arg2, arg3)

マクロの説明

PollDefinition セクション内でのみ使用されます。生成された CSV ファイルから一括統計情報カウンタ値を取得することに特化して使用されます。同じタイプに属するカウンタの取得に役立ちます。現在および他のレポートで参照されるすべてのカウンタを、同じポール マクロ内の同じラベルを使用してポーリングする必要があることに注意してください。これは、一括統計情報ファイルの再解析を防ぐためです。

- arg1：カウンタのタイプを示します。
- arg2：一括統計情報ファイルからポーリングするためのカウンタ名のリスト（二重引用符で挟まれたカウンタ名のカンマ区切りリスト）。各カウンタ変数にカウンタ変数のデータ型のプレフィックスが付いているため、Prime Performance Manager で変数の解析方法が判断できます。
- arg3：スキーマ / タイプに対するインデックスまたはキー カウンタのリスト（二重引用符で挟まれたカウンタ名のカンマ区切りリスト）。各インデックス変数にカウンタ変数のデータ型のプレフィックスが付いているため、Prime Performance Manager で変数の解析方法が判断できます。

例

```
CsvPoll("gtpu",
             "integer:curr-sess,
```

```
integer:total-setup-sess,
integer:curr-gtpu0-sess,
long:curr-gtpu1-sess",
"string:vpnname,
integer:vpnid,
string:servname,
integer:servid");
```

DATEANDTIME

構文

DATEANDTIME ()

マクロの説明

SNMPv2-TC で定義された DateAndTime 形式または Java Calendar.getTimeInMillis() で定義されたミリ秒での時間値を返します。詳細な入力情報：

```
 Size list:1: 8
                2: 11
Display hint:2d-1d-1d,1d:1d:1d.1d,1a1d:1d
Description:A date-time specification.
        field  octets  contents                  range
        ----  ------  --------                   -----
         1      1-2   year                        0..65536
         2       3    month                      1..12
         3       4    day                          1..31
         4       5    hour                         0..23
         5       6    minutes                     0..59
         6       7    seconds                     0..60   (use 60 for leap-second)
         7       8    deci-seconds               0..9
         8       9    direction from UTC       '+' / '-'
         9      10    hours from UTC            0..11
        10      11    minutes from UTC        0..59
```

（注）　現地時間しかわからない場合、時間帯情報（フィールド 8 ～ 10）はありません。

例

1992 年 5 月 26 日火曜日の 1:30:15 PM EDT は 1992-5-26,13:30:15.0,-4:0 と表示されます。

DELTA

構文

DELTA (*object*)

マクロの説明

このマクロは、現在と前回のポーリング値の間の差分を返します（currentPolling - previousPolling）。ポーリングが実行されるたびに、ポーリング値は「前回」のポーリング値として使用されるために維持され、次回ポーリングが実行されます。またこのマクロでは、最初のポーリングが起動して（前回の値なし）そして回数がオーバーフローした場合など発生する可能性のある状況も考慮されています。

- object は数値タイプで、ラップが有効だと検出した場合にこのパラメータはラップを確認し、差分を計算する前に調整します。
- ProcessPollResult セクション内でのみ使用されます。

例

```
AuthenTransactionSuccesses     = casAuthenTransactionSuccesses.delta();
```

DELTANEXT

構文

DELTANEXT (*object*, arg1)

マクロの説明

このマクロは、前のポーリングの代わりにデータベース内の次の行との間で DELTA マクロを呼び出すことができます。このマクロは、arg1 の行の値が変化するまで差分を計算し続けます。

- *object* は数値型です。arg1 は文字列型であり、PollDefinition セクション内で前に作成された変数のインデックスの名前です（オブジェクトと同じテーブル内に存在していなければならない）。

例

```
duration = rttMonIcmpJitterStatsStartTimeId..deltaNext("rttMonCtrlAdminIndex") / 100;
```

DEVICECUSTOMNAME

構文

DEVICECUSTOMNAME()

マクロの説明

デバイスのカスタム名を返します。

例

```
deviceCustomName();
```

DEVICEID

構文

DEVICEID()

マクロの説明

デバイスの一意な ID を返します。

例

```
deviceID();
```

DEVICEIPADDRESS

構文

DEVICEIPADDRESS()

マクロの説明

現在のコンテキストのデバイスの IP アドレスを返します。

例

次のエントリは、デバイスの IP アドレスが「102.168」を含むかどうかを判断します。

```
If(Contains(DeviceIpAddress(), "102.168"), true, false);
```

DEVICELOCATION

構文

DEVICELOCATION()

マクロの説明

デバイス位置の属性を返します。

例

次のエントリは、デバイスのロケーション属性が「labx」を含むかどうかを判断します。

```
If(Contains(DeviceLocation(), "labx"), true, false);
```

DEVICENAME

構文

DEVICENAME()

マクロの説明

デバイス名を返します。

例

次のエントリは、デバイス名属性が「ppm7000a」を含むかどうかを判断します。

```
If(Contains(DeviceName(), "ppm7000a"), true, false);
```

DEVICESOFTWAREVERSION

構文

DEVICESOFTWAREVERSION()

マクロの説明

デバイスのソフトウェア バージョンを返します。

例

次のエントリは、現在のデバイスのソフトウェア バージョンが「1.4.0」を含むかどうかを判断します。

```
If(Contains(DeviceSoftwareVersion(), "1.0.3"), true, false);
```

DEVICESYNCNAME

構文

DEVICESYNCNAME()

マクロの説明

デバイス同期名を返します。

例

次のエントリは、デバイス同期名が「ppm7000a」を含むかどうかを判断します。

```
If(Contains(DeviceSyncName(), "ppm7000a"), true, false);
```

DEVICESYSNAME

構文

DEVICESYSNAME()

マクロの説明

デバイスのシステム名を返します。

例

次のエントリは、デバイスのシステム名が「ppm7000a」を含むかどうかを判断します。

```
If(Contains(DeviceSysName(), "ppm7000a"), true, false);
```

DEVICETYPE

構文

DEVICETYPE (object)

マクロの説明

ノードのデバイス タイプがあった場合にそれを返します。それ以外の場合は「unknown」を返します。

- object は sysObjectID です。
- SystemCapabilities.xml および UserCapability.xml ファイルで使用されます。

例 :

```
If((DeviceType() == "vmwESX"), Environment(" SNMPENV_MAX_VARBIND = 10;"), false);
```

DOUBLEVALUE

構文

DOUBLEVALUE (object)

マクロの説明

object を倍精度に変換して返します。失敗すると、ヌルを返します。

例

```
satSigNoiseRatio  = saSatSigCndisp.doubleValue();
```

ENDSWITH

構文

ENDSWITH (object, arg1)

マクロの説明

このマクロは、文字列 *object* の末尾が文字列 arg1 の場合に true を返します。これは、Java の endsWith 文字列機能に似ています。

- object は文字列タイプ
- arg1 は文字列です。

例

```
memoryPoolInfo = memoryPoolInfo.filter(not(cempMemPoolName.endsWith("image")));
```

ENTITYINFO

構文

ENTITYINFO()

マクロの説明

このマクロは、ENTITY-MIB のデータを取得するために使用されます。これがキャッシュ内にある場合、キャッシュ内のデータが返されます。そうでない場合は、維持されているデータを確認して、見つかってそのエンティティが最後のポーリングから変化していない場合にその維持されているデータが返されます。まだ利用できるデータがない場合、デバイスはそのエンティティ データをポーリングします。

例

```
entPhysicalTable = EntityInfo();
```

EXPONENT

構文

EXPONENT (arg1, arg2)

マクロの説明

このマクロは、指数値を返します。失敗した場合は、ヌルを返します。

- arg1 は基数で arg2 は指数です。

例

```
exponentValue = Exponent("e", 2);
```

FILTER

構文

FILTER (object, arg1)

マクロの説明

このマクロは、条件に合格しないアイテム（false を返すアイテム）を削除した、オブジェクトのサブセットを返します。

- object はテーブルです。
- arg1 は条件（ブール型の結果を返す）です。

例

```
cgnStatisticsTable = cgnStatisticsTable.filter(true);
```

FLOWPOLL

構文

FLOWPOLL (arg1, arg2, arg3)

マクロの説明

NetFlow 統計情報を受信 NetFlow ストリームから取得するために PollDefinition のセクションだけで使用します。

- arg1：NetFlow ストリームから取得される非キー フィールドのリスト（二重引用符で挟まれたフィールド名のカンマ区切りリスト）。
- arg2：NetFlow ストリームから取得されるキー フィールドのリスト（二重引用符で挟まれたフィールド名のカンマ区切りリスト）。キー フィールドには任意に選択できるコロンで区切られたデフォルト値を設定できます。
- arg3：テンプレート ID フィールドのリスト（二重引用符で挟まれたテンプレート番号のカンマ区切りリスト）。キー フィールドでフローを一意に指定するのにキー フィールドが十分でない場合、追加フィルタとして動作するテンプレート ID を指定できます。

例：

```
flowPoll("octets,
                    pkts",
                   "input,
                    srcaddr,
                    flowDirection:0");

flowPoll("egressVRFID,
                     postNATSourceIPv4Address,
                     postNAPTSourceTransportPort",
                    "ingressVRFID,
                     sourceIPv4Address,
                     sourceTransportPort,
                     protocolIdentifier",
                     "256,257");
```

FOR

構文

FOR (arg1,arg2, arg3, arg4)

マクロの説明

このマクロが使用する引数：Java の「for」でのループ初期値、ループ制御、およびループ増分、そして各プロセッサの戻り値を示します。戻り値が一時停止した場合、Java の for ループが一時停止します。

- arg1 はループ初期値です。
- arg2 はループ制御です。
- arg3 はループ増分です。
- arg4 はプロセッサを示す値です。

例

```
for(index1=0,value @gt 0,index1 = index1 + 1, value = value / 10.0);.
```

FOREACH

構文

FOREACH (object, arg1, arg2)

マクロの説明

このマクロはオブジェクトの各レコード全体で繰り返しをするために Java「for」を使用します。arg1 とともに各 object が戻り値を指定するために使用します。戻り値が一時停止した場合、Java の「for」ループが一時停止します。

- object はリストです
- arg1 は文字列です。
- arg2 はプロセッサを示す値です。
- ヌルを返します。

例

```
forEach(cnpdAllStatsTable,row,totalInPkts     = totalInPkts +
row.get("cnpdAllStatsInPkts"));
```

FORMATTIME

構文

FORMATTIME (object)

マクロの説明

object を HH:MM:SS 形式の文字列に変換し、この文字列を返します。障害が発生した場合、「00:00:00」が返されます。

- object は秒数です。

例

```
UpTime = FormatTime(cpuUpTime/100);
```

GET

構文

GETALL (object, arg1)

マクロの説明

このマクロは、arg1 で指定された要素を返します。

- object はテーブルです。
- arg1 は文字列です（キー）。

例

```
row.get("cnpdAllStatsInPkts"));
```

GETALL

構文

GETALL (object, arg1)

マクロの説明

このマクロは、arg1 のカラムの値をすべて返します。失敗した場合は、ヌルを返します。

- object はテーブル
- arg1 は文字列です（キー）。

例

```
cgnInstanceTable.getAll("serviceTypeNat")
```

GETAVAILABILITYINFO

構文

GETAVAILABILITYINFO()

マクロの説明

このマクロは、ノードから可用性情報（システムが起動しているかどうか）を取得します（MWTMCURRTIME、sysUpTime、sysName、および可用性）。Poll セクションのレポートで、追加の引数 alwaysExecute="true" が必要です。

- PollDefinition セクション内でのみ使用されます。

例

```
deviceVariables = getAvailabilityInfo();
```

GETDSCP

構文

GETDSCP (arg1)

arg1 として Type of Service (ToS) フィールドに入力する DiffServ コード ポイント（DSCP）のテキスト表現を返すためには ProcessPollResult セクションだけを使用します。このマクロは、名前の検索のために ToS フィールドへの入力の上位 6 ビットを使用します。このマクロは、一般に数値の代わりに DSCP 名を表示する NetFlow レポートのために使用されます。表 9-1 に、DSCP 名および十進数と ToS の値を表示します。

表 9-1　　DSCP の 10 進数および ToS の値

DSCP 名	10 進値	ToS 値
AF11[1]	10	40
AF12	12	48
AF13	14	56
AF21	18	72
AF22	20	80
AF23	22	88

表 9-1　DSCP の 10 進数および ToS の値 （続き）

DSCP 名	10 進値	ToS 値
AF31	26	104
AF32	28	112
AF33	30	120
AF41	34	136
AF42	36	144
AF43	38	152
CS1[2]	8	32
CS2	16	64
CS3	24	96
CS4	32	128
CS5	40	160
CS6	48	192
CS7	56	224
EF[3]	46	184
デフォルト	0	0

1. AF = 確認転送
2. CS = クラス セレクタ
3. EF = 緊急転送

例：

```
GetDSCP(96) returns "CS3"
```

GETECN

構文

GETECN (arg1)

arg1 として入力された Type of Service (ToS) フィールドにエラー輻輳通知（ECN）のテキスト表現を返すためには ProcessPollResult セクションだけを使用します。このマクロは、名前の検索のために入力された ToS フィールドの下位 2 ビットを使用します。通常は、数値の代わりに ECN 名を表示するために NetFlow レポートが使われます。表 9-2 に ToS 値および ECN 名を示します。

表 9-2　ToS 値および ECN 名

ToS 値	ECN 名	ECN 説明
00	Not-ECT	ECN-Capable 転送ではない
01	ECT(0)	ECN-Capable 転送
10	ECT(1)	ECN-Capable 転送
11	CE	Congestion Experienced

例：

```
GetECN(11) returns "CE"
```

GETHOSTADDRESS

構文

GETHOSTADDRESS (object)

マクロの説明

このマクロはホスト アドレスを文字列形式で返します。失敗した場合は、ヌルを返します。

- object は IP アドレスです。
- ProcessPollResult セクション内でのみ使用されます。

例

```
RserverIpAddress  = serverIpAddress.getHostAddress();
```

GETHOSTNAME

構文

GETHOSTNAME(object, arg)

マクロの説明

このマクロはホスト名を文字列形式で返します。ホスト名は任意指定の arg パラメータで定義されている名前解決方法を使用して解決されます。この arg パラメータが指定されていない場合、マクロは Reports.properties ファイルの RESOLVE_HOST_NAMES で定義されている名前解決方法を使用します。RESOLVE_HOST_NAMES が Reports.properties にない場合、このマクロはホスト名への IP を解決するために DNS を使用します。名前解決方法のやり方に関係なく、このマクロが IP アドレスを解決できない場合、IP アドレス自体をホスト名として返します。

- object は IP アドレスです。
- arg は、名前解決方法を定義するために使用される任意指定のパラメータです。使用する方法：DNS または PPM。使用できる値は次のいずれかです。
  - dns
  - ppm
  - dns,ppm
  - ppm,dns

何も指定しない場合で、RESOLVE_HOST_NAMES の設定が Reports.properties ファイルにない場合、デフォルトは dns です。

このマクロは ProcessPollResult のセクションでのみ使用されます。

次に、例を示します。

```
targetAddress1.getHostName()
targetAddress2.getHostName("ppm")
targetAddress2.getHostName("ppm, dns")
```

GETPINGINFO

構文

GETPINGINFO()

マクロの説明

このマクロは、ノード オブジェクトを取得するためにデバイス コンテキストを使用します。ノード オブジェクトは ping 結果のリスト（マッピング）を返します。

例

```
deviceVariables = getPingInfo();
```

GETPREFIX

構文

GETPREFIX (arg1, arg2)

IP アドレスおよびをサブネット マスクを入力として使用して IP アドレス プレフィックスを返すために ProcessPollResult のセクションだけで使用されます。

- arg1：完全な形式の IP アドレス。
- arg2：サブネット マスク ビット。

例：

```
GetPrefix(10.1.1.10, 24) returns an InetAddress 10.1.1.0.
```

文字列形式の値に戻すにはプレフィックスを渡す GetHostAddress() マクロを使用できます。

```
prefixObj = GetPrefix(addr, mask);
prefixString = prefixObj.getHostAddress();
```

GETSERVERBY PORT

構文

GETSERVBYPORT (arg1, arg2)

特定のポート番号およびプロトコル名のサービス名を返すために ProcessPollResult のセクションでだけで使用されます。サービス名の検索のためにゲートウェイおよび装置上の etc ディレクトリでサービス ファイルを使用します。

- arg1：ポート番号
- arg2：プロトコル名

通常は、サービスまたはアプリケーション名を表示するには NetFlow レポートとともに使用します。

例：

```
GetServByPort(80, TCP) returns HTTP.
```

GETSYSTEMPROPERTY

構文

GETSYSTEMPROPERTY(arg1, arg2)

マクロの説明

このマクロは、文字列の形式でシステム プロパティを読み込むために arg1 を使用します。これに失敗すると、arg2 はデフォルトの戻り値として使用されます。

- arg1 はプロパティ キーです。
- arg2 はデフォルト値です。

例

```
IfNameFormat = GetSystemProperty("IFNAME_FORMAT", "both");
```

GROUP

構文

GROUP(object, arg1, arg2)

マクロの説明

このマクロは、表の行をグループ化します。このマクロはカウント マクロとともに使用すると有効です。これはグループ化されたデータを返します。

- object はグループ化を必要とする行です。
- arg1 および arg2 は 1 つの行でグループ化する必要のある列名です。
- cmts.xml のレポートに使用します。

例

```
cmStatusGroups = cdxCmtsCmStatusExtTable.group("cdxCmtsCmStatusValue,
docsIfCmtsCmStatusIndex");
```

HASCAPABILITY

構文

HASCAPABILITY(arg1)

マクロの説明

このマクロは、パラメータとしてリストされている機能の名前が、指定されたノードのオブジェクトに適用できるかどうかを確認します。

- arg1 は機能の名前です。

例

```
if(hasCapability("MPLS_L3_VPN");
```

HASINTERFACE

構文

HASINTERFACE(arg1)

マクロの説明

このマクロは、特定のインターフェイスの存在に基づいた機能の設定を有効にするために使用することを目的としています。

- arg1 は条件に一致するデバイス インターフェイスがあるかどうかを確認するために使用される条件文です。
- SystemCapability.xml ファイル内でのみ使用されます。

例

```
hasInterface(ifDescr.startsWith("GigabitE"));
```

HASMATCHINGENTRIES

構文

HASMATCHINGENTRIES (arg1,arg2)

マクロの説明

このマクロは、フィルタに一致するエントリがテーブルにあるかどうかを確認するために使用されます。

- arg1 はポーリング結果テーブル
- arg2 は一致させるフィルタ

例

```
Y1731_MIB_IOS    = Not(IOS_XR) @and hasMatchingEntries(
                            poll("rttMonCtrlAdminIndex,
                                  rttMonCtrlAdminOwner,
                                  rttMonCtrlAdminTag,
                                  rttMonCtrlAdminRttType,
                                  rttMonCtrlAdminFrequency",
                                 "rttMonCtrlAdminTable"),
                            (rttMonCtrlAdminRttType @eq 23) @or (rttMonCtrlAdminRttType
@eq 24));
```

HASSENSOR

構文

HASSENSOR (arg1,arg2)

マクロの説明

文字列「transmit」、「receive」、「tx」、「rx」は、エンティティ テーブル内の共通の文字列であり、必ずしも光センサー固有ではないため、このマクロはこれらの文字列が十分ではないことを確認しています。また entityphysicalvendortype フィールドも検査する必要があります。

- arg1 はエンティティ テーブル
- arg2 は一致させるフィルタ

例

```
     CISCO_ENTITY_SENSOR_MIB_RX = CISCO_ENTITY_SENSOR_MIB @and
                                  (Not(CPT_NGXP) @and
hasSensor(entPhysicalVendorType.contains("1.3.6.1.4.1.9.12.3.1.8.46")));

    CISCO_ENTITY_SENSOR_MIB_TX = CISCO_ENTITY_SENSOR_MIB @and
                                  (Not(CPT_NGXP) @and
hasSensor(entPhysicalVendorType.contains("1.3.6.1.4.1.9.12.3.1.8.47")));
    CISCO_ENTITY_SENSOR_MIB_TEMP = CISCO_ENTITY_SENSOR_MIB @and
                                  (Not(CPT_NGXP) @and
hasSensor(entPhysicalVendorType.contains("1.3.6.1.4.1.9.12.3.1.8.50")));
    CISCO_ENTITY_SENSOR_MIB_CURRENT = CISCO_ENTITY_SENSOR_MIB @and
                                  (Not(CPT_NGXP) @and
hasSensor(entPhysicalVendorType.contains("1.3.6.1.4.1.9.12.3.1.8.48")));
    CISCO_ENTITY_SENSOR_MIB_VOLTAGE = CISCO_ENTITY_SENSOR_MIB @and
                                  (Not(CPT_NGXP) @and
hasSensor(entPhysicalVendorType.contains("1.3.6.1.4.1.9.12.3.1.8.49")));
```

HASVAR

構文

HASVAR (arg1)

マクロの説明

このマクロは、パラメータ リストの MIB 変数が指定されたノードにあるかどうかを確認します。

- arg1 は MIB のテーブルまたはインスタンス変数です
- SystemCapability.xml ファイル内でのみ使用されます。

例

```
TCP_MIB  = hasVar("tcpInSegs");
```

HEX2STRING

構文

HEX2STRING(object, arg1)

マクロの説明

このマクロは、16 進数から文字列への変換をサポートするために作成されます。デリミタは、16 進数値を分けるために入力パラメータとしてサポートされます。

- object は、16 進数値を保持しています。
- arg1 はデリミタです。

例

```
fcIfWwn = fcIfWwn.hex2String(":");
```

HYPERVISORPOLL

構文

HYPERVISORPOLL(arg1, arg2, arg3, arg4, arg5)

マクロの説明

このマクロは、ハイパーバイザのコレクタをサポートするために作成されます。このマクロは、ハイパーバイザ ポーリングの結果を返します。

arg0 は機能名

arg1 は入力パラメータ

arg2 は値のキー

arg3 はスタティックまたはダイナミックを示す

arg4 はキャッシュ キー

例

```
hypervisorPoll("ListVMAvailability", "",  "", false);
```

HYPERVISORPOLLPERSIST

構文

HYPERVISORPOLLPERSIST()

マクロの説明

このマクロは、ハイパーバイザ機能を取得し、それを保持するために作成されます。

例

```
hypervisorPollPersist().
```

IF

構文

IF (object, arg1, [arg2])

マクロの説明

2 つの引数が使用されると、object が真ならば arg1 を返すか実行し、object が偽ならばヌルを返します。3 つの引数が使用されると、object が真ならば arg1 を返すか実行し、object が偽ならば arg2 を返すか実行します。

object はブール型です。

例

```
 if(Linux,LinuxDevices());
```

IFDESCR

構文

IFDESCR ([object])

マクロの説明

このマクロは、インターフェイスの説明を返します。先に検索することで、再ポーリングを防ぎます。このマクロが返すオプションの引数は、この操作のインデックス キーです（デフォルトは ifIndex）。

ProcessPollResult セクション内でのみ使用されます。

例

```
interfaceDescr   = IfDescr(interfaceIndex);
```

IFINFO

構文

IFINFO ()

マクロの説明

このマクロは、現在のデバイスの Iftable および ifXtable SNMP データを含む行のリストを返します。

例

```
interfaceTable = IfInfo();
```

IFSPEED

構文

IFSPEED ([object])

マクロの説明

このマクロはインターフェイスの設定速度を返します。

ProcessPollResult セクション内でのみ使用されます。

オプションの引数は、この操作のインデックス キーです（デフォルトは ifIndex）。

例

```
txSpeed = IfSpeed();
```

IFSPEEDRECEIVE

構文

IFSPEEDRECEIVE ([object])

マクロの説明

このマクロは、インターフェイスの受信速度を返します。

- オプションの引数は、この操作のインデックス キーです（デフォルトは ifIndex）。

例

```
ifSpeedReceive();
```

IFTABLE

構文

IFTABLE ( )

マクロの説明

このマクロは、現在のデバイスの Iftable SNMP データを含む行のリストを返します。

例

```
interfaceTable = IfTable();
```

INRANGE

構文

INRANGE(arg1, arg2)

マクロの説明

このマクロは、IPSLA インデックスがプローブ リスト rttMplsVpnMonCtrlProbeList に含まれているかどうかを調べる必要がある MPLS OAM レポートに対して範囲一致をサポートするために作成されます。次の例は、このマクロによってサポートされます。

（a）1,5,3 のようにカンマで区切られた個別の ID。

（b）1-10,12-34 のようにカンマで区切られた複数の範囲のハイフンを含む範囲形式。

（c）1,2,4-10,12,15,19-25 のような上記の 2 形式の混在。

成功した場合は true を返し、それ以外の場合は false を返します。

- arg1 はデータ範囲を示す文字列です。
- arg2 は文字列形式の数です。

例

```
InRange(rttMplsVpnMonCtrlTable.rttMplsVpnMonCtrlProbeList,
rttMonStatsCaptureTable.rttMonCtrlAdminIndex))
```

INTERVALDURATION

構文

INTERVALDURATION ()

マクロの説明

このマクロは、指定されたレポートの時間間隔を返します（秒単位）。レポートでは通常、15 分、1 時間、1 日などになります。

- WebReport セクションと CSV セクションでのみ使用されます。

例、

```
IntervalDuration();
```

INTVALUE

構文

INTVALUE (object)

マクロの説明

オブジェクトを整数に変換します。エラーが発生すると、ヌルを返します。

例

```
cpwVcMplsLocalLdpID.intValue(4,2);
```

INVENTORYPERSIST

構文

INVENTORYPERSIST ()

マクロの説明

他のマクロでの使用に対する実行コンフィギュレーションおよびエンティティの最後の変更時刻の値を保持します。

例

```
InventoryPersist();
```

IOSVERSION

構文

IOSVERSION (object)

マクロの説明

object が「sysDescr」の場合（通常の指定）、指定された文字列から IOS バージョンを解析します。

- SystemCapability.xml ファイル内でのみ使用されます（一度しか使用されません）。

例

```
iosVer = iosVersion(deviceVersion);
```

IPADDRESS

構文

IPADDRESS (object, [arg1, arg2])

マクロの説明

1 つの引数のみが使用されると、IP アドレスに変換されたオブジェクトを返します。2 つの引数のみが使用されると、IP アドレスに変換されたオブジェクト（「arg1」のバイトから開始）を返します。3 つの引数が使用されると、IP アドレスに変換されたオブジェクト（「arg1」のバイトから開始し、「arg2」のバイトで終了）を返します。オフセットは、Java の substring 関数に似ています。

- object はオクテット文字列のアドレスです。arg1 と arg2 はオフセットです。
- ProcessPollResult セクション内でのみ使用されます。

例

```
cmStatusIpAddress     = docsIfCmtsCmStatusInetAddress.ipAddress();
```

ISHYPERVISOR

構文

ISHYPERVISOR ( )

マクロの説明

このマクロは、デバイスがハイパーバイザかどうかを判断するために作成されます。

例

```
if( isHypervisor(), AllHypervisors(), AllDevices());
```

ISNULL

構文

ISNULL (object, [arg1])

マクロの説明

1 つの引数が使用されると、object がヌルならば真を返し、そうでなければ偽を返します。2 つの引数が使用されると、object がヌルでなければ object を返し、object がヌルならば arg1 を返します。

例

```
txOctets = ifHCOutOctets.isNull(ifOutOctets);
```

ISTABLEEMPTY

構文

ISTABLEEMPTY (object)

マクロの説明

このマクロは、パラメータ リストの各 MIB テーブルが空かどうかまたはノードにないかどうかを確認します。object が空の場合、真を返します。

- object は、MIB テーブル名です。
- SystemCapability.xml ファイルと PollDefinition セクションでのみ使用されます。

例

```
isTableEmpty("dot3HCStatsTable").
```

ISTABLENOTEMPTY

構文

ISTABLENOTEMPTY (object)

マクロの説明

このマクロは、パラメータ リストの各 MIB テーブルが空でないかどうかを確認します。オブジェクトが空ではない場合に true を返します。

- object はテーブルです。

例

```
isTableNotEmpty("mplsInSegmentTable")
```

JOIN

構文

JOIN (object, arg1, arg2)

マクロの説明

このマクロは、*object* と arg1 を結合して得られたテーブルを返します。*object* の行と arg1 の行は、条件（arg2）が true になる場合に結合されます。

- object と arg1 はテーブルです。arg2 は一致条件を指定します。
- PollDefinition セクション内でのみ使用されます。

例

```
stats = stats.join(interfaceTable, (stats.ifIndex == interfaceTable.ifIndex));
```

JMXPOLL

構文

JMXPOLL(BeanName [Path [**s** | **c** | **d**]

マクロの説明

サーバから Java Management Extensions（JMX）属性を取得します。マクロは次の 2 通りの方法で起動できます。

- JMXPoll（BeanName）：属性を指定しないで Bean 名のみをポーリングします。
- JMXPoll（BeanName）：ポーリングする属性および属性の深さを指定して Bean 名のみをポーリングします。オプション：
  - BeanName：Bean の完全な名前。
  - Path：属性とそのサブ属性（ある場合）の名前。# 区切り記号が属性のレベル間、つまり属性とサブ属性の間に使用されます。
  - s：path での単純な属性を取得します。例：文字列、長文字列、文字列配列データ、長配列データ。
  - c：path からすべての子属性を取得します。例：複合データ、表形式データ。
  - c：リーフまでのすべての子ノードを path から取得します。例：複合データ、表形式データ。例：複合データ配列データ、複合および表形式データの組み合わせ。

例：

```
JMXPoll("java.lang:type=Runtime","SystemProperties#java.ext.dirs ","s");
where 'java.lang:type=Runtime' is the beanName.
         'SystemProperties' is the attribute.
         'java.ext.dirs' is the sub-attribute.
```

JVM bean には多くのデータ タイプがあります。次は JMXPoll コールの例の概要です。

- シンプルなデータ タイプ

  ```
  JMXPoll("java.lang:type=Compilation","Name","s");
  ```

- 文字列配列データ

  ```
  JMXPoll("java.util.logging:type=Logging","LoggerNames","s");
  ```

- 複合データ

  ```
  JMXPoll("java.lang:type=MemoryPool","CollectionUsage","c");
  ```

- 表形式データ

  ```
  JMXPoll("java.lang:type=Runtime","SystemProperties","d");
  ```

- 整数配列データ

```
JMXPoll("java.lang:type=Threading","AllThreadIds","s");
```

- 複合データ配列データ

```
JMXPoll("com.sun.management:type=HotSpotDiagnostic","DiagnosticOptions","d");
```

- 表 + 複合データ配置データ

```
JMXPoll("java.lang:type=GarbageCollector","LastGcInfo","d");
```

LEFTJOIN

構文

LEFTJOIN (object, arg1, arg2)

マクロの説明

このマクロは、object と arg1 を結合して得られたテーブルを返します。object の行と arg1 の行は、条件（arg2）が true になる場合に結合されます。ただし、object 内の各行は、arg1 で指定されたオブジェクト内に一致する行がなくても、結果として得られるテーブルに引き続き保持されます。

- object と arg1 はテーブルです。arg2 は一致条件です。

例

```
casStats.leftJoin(casConf, ((casConf.casIndex == casStats.casIndex    @and
(casConf.casProtocol == casStats.casProtocol))));
```

LENGTH

構文

LENGTH (object)

マクロの説明

このマクロは、object 内の文字の数を返します。これは、Java の length 文字列機能に似ています。

- object は文字列型です。

例

```
if(((Length(cpwVcMplsPeerLdpID) == 6))
```

LONGVALUE

構文

LONGVALUE (object)

マクロの説明

オブジェクトを long 型に変換します。エラーが発生すると、ヌルを返します。

例

```
LongValue(GetSystemProperty("IPSLA_FTP_FILE_SIZE", 1048576)), size);
```

MATCHES

構文

MATCHES (object, arg1)

マクロの説明

このマクロは、arg1 の正規表現パターンが object 内で検出されると、true を返します。

- object は文字列です。arg1 は、Java の正規表現パターンです。

例

```
ifDescr.matches("\w"));
```

MACADDRESS

構文

MACADDRESS (object)

マクロの説明

このマクロは、SNMPv2-TC で定義された指定済み MACAddress 形式を返します。

- object は、MACAddress の Byte Array 形式です。

例

```
flapMacAddr = ccsCmFlapMacAddr.macAddress().
```

NOT

構文

NOT (arg)

マクロの説明

object の反対を返します。

- arg はブール型です。

例 :

```
not( ifDescr.startsWith("unrouted vlan") );
```

PARSESTRING

構文

PARSESTRING (arg1, arg2, arg3)

マクロの説明

arg1 をトークンに解析し、arg2 で識別されるトークンを返します

- arg1 は文字列定数または変数です。
- arg2 はトークン内の arg1 を区切るために使用する区切り文字です。
- arg3 は返すトークンを指定する整数の値です。

例：

```
chassisName = ParseString(adaptorDesc, "/", 2);
ParseString("sys/chassis-1/blade-2/board/memarray-1", "/", 2) returns "chassis-1".
```

POLL

構文

POLL (arg1, arg2)

マクロの説明

スカラ値のグループまたは同じインデックスを共有するテーブル変数のみをポーリングします。

- PollDefinition セクション内でのみ使用されます。
- arg1 はポーリングする変数のリスト（二重引用符で挟まれた変数のカンマ区切りリスト）
- arg2 は再ポーリングを防ぐために他のポーリング内で参照できるようにするために使用されるラベルです。

（注）　現在および他のレポートで参照されるすべての変数を、同じポール マクロ内で同じラベルを使用してポーリングする必要があります。これは、変数が欠落していることを防ぐためです。ポーリング結果がラベルに関連付けられると、ポーリング マクロが同じラベルでコールされてもそのデバイスは再ポーリングされません。

例：

```
cpmCPUTotalTable = poll("cpmCPUTotalIndex,
                         cpmCPUTotalPhysicalIndex,
                         cpmCPUTotal5minRev,
                         cpmCPUTotal1minRev");
```

POLLNEXT

構文

POLLNEXT (arg1, arg2, arg3, arg4)

マクロの説明

まだ取得していないデータを取得します（使用可能な次のデータを取得します）。

PollDefinition セクション内でのみ使用されます。

- arg1 は、各行を一意に識別するインデックスです（二重引用符で囲まれ、カンマで区切られた変数のリスト）。
- arg2 は、ポーリングする変数のリストです（二重引用符で囲まれ、カンマで区切られた変数のリスト）。
- arg3 は、ポーリングするインデックス変数のリストです。
- arg4 はブール値です。最後の行が直近のポーリングの最後の行と同じ場合でも最後の行を返す場合に真にします。

例：

```
rttMonStatsCaptureTable =
```

```
pollNext("rttMonCtrlAdminIndex,
          rttMonStatsCapturePathIndex,
          rttMonStatsCaptureHopIndex,
          rttMonStatsCaptureDistIndex",
         "rttMonStatsCaptureStartTimeIndex",
         "rttMonCtrlAdminIndex,
          rttMonStatsCaptureStartTimeIndex,
          rttMonStatsCapturePathIndex,
          rttMonStatsCaptureHopIndex,
          rttMonStatsCaptureDistIndex,
          rttMonStatsCaptureCompletions,
          rttMonStatsCaptureSumCompletionTime",
         true);
```

POLLPERSIST

構文

POLLPERSIST (arg1)

マクロの説明

リスト内の変数をポーリングし、値を他のアルゴリズムで使用可能にするためにその値を（行マップ内に）保持します。

値をポーリングし、現在のコンテキストに格納します（他の機能チェックですぐに使用するために使用可能）。SystemCapability.xml ファイル内でのみ使用されます。

- arg1 は、ポーリングする変数のリストです。

例：

```
VARS = pollPersist("sysDescr,sysObjectID");
```

PRINT

構文

PRINT ([object])

マクロの説明

コンソール ログ内の object の文字列バージョンを出力します。

- 引数が指定されなければ、空白行をコンソール ログに出力します（デバッグ用）。
- 1 つの引数が使用されると、そのオブジェクトをコンソール ログに出力します（デバッグ用）。

例：

```
print(result);
```

PROCESSORLIST

構文

PROCESSORLIST (arg1)

マクロの説明

このマクロは、内部処理コードが参照するオブジェクトの「リスト」を作成することを目的としています。

o　arg1 は Java の Array 構文で指定した次のようなオブジェクトのリストです。

["object1", "object2", "object3"]

o　通常、Usage および Objects ディレクティブに対して保持されたグループ定義ファイル内で使用されます。

例

```
Usage = ProcessorList( [ "AGG_GGSN_APN_INS_MIS", "AGG_GGSN_APN_DHCP",
"AGG_GGSN_APN_INS_PDP"] );
```

PROTOCOLNAME

構文

PROTOCOLNAME (arg1)

arg1 としてプロトコル番号に提供されたプロトコルの名前を返すために ProcessPollResult のセクションでだけ使用されています。たとえば、17 に対して UDP が返されます。このマクロは、検索するプロトコル名に対するゲートウェイおよび装置の etc ディレクトリ内の netflow-config.xml ファイルで使用されます。このマクロは、一般に数字の代わりにレポート内のプロトコルを表示する NetFlow レポートのために使用されます。

例：

```
protocol        = ProtocolName(17);
```

RATE

構文

RATE (object, [arg1, arg2])

マクロの説明

1 つの引数が使用されると、オブジェクトに関する前回と今回のポーリング間での変化の速度を返します（時間遅延に sysUpTime を使用して計算を実行）。

3 つの引数が使用されると（2 つの引数を使用するオプションはない）、オブジェクトに関する前回と今回のポーリング間での変化の速度を返します。この場合、arg1 は時間に使用される値です。arg2 は正しいメトリックを得るための乗数または変換係数です。つまり、秒数またはミリ秒数にするために使用されます。

たとえば、変数 **sysUpTime** は、1/100 秒の単位で記録されるので、秒数にするには 100 の乗数が必要です。

例：

```
txBitRate = txOctets.rate() * 8;
```

SETALGORITHMS

構文

SETALGORITHMS (arg1, arg2)

マクロの説明

実行するアルゴリズムを定義します。

- arg1 はアルゴリズム名です。
- arg2 はマクロ ステートメントを含むリストかまた key=value のマップのペアのいずれかで、キーは値に定義されたステートメントに割り当てられたオブジェクトの名前です。

例 :

```
setAlgorithms("AllHypervisors", {HYPERVISOR_CAPABILITY = hypervisorPollPersist();});
```

SETCPUINFO

構文

SETCPUINFO ([object])

マクロの説明

CPU に関する情報（cpuDescr、cpuNum、cpuSlot）を設定します。CPU インデックスを上書きするには、object をその値に設定します。

例 :

```
setCpuInfo();
```

SETMEMORYPOOLINFO

構文

SETMEMORYPOOLINFO ()

マクロの説明

このマクロは、CISCO-ENHANCED-MEMORYPOOL-MIB および ENTITY-MIB のポーリング後に結合されたメモリ プール名の作成に使用されます。新しいメモリ プール名は「slotNumber-slotName-memoryPoolName」です。データがポーリングされると、データが変化していない場合は再利用のためにデータは保持され、他のマクロの使用のためにキャッシュ内にも保存されます。

例 :

```
memoryPoolInfo = SetMemoryPoolInfo();
```

SETTIMEVARINFO

構文

SETTIMEVARINFO (object, arg1, arg2)

マクロの説明

指定された引数に従って時間変数情報をコンテキストに設定します。

- object は使用する時間変数です。
- arg1 は変数を次回使用するかどうかを示すブール値
- arg2 は値を変更するまで使用するインデックスです。

例 :

```
setTimeVarInfo("sysUpTime", true);
```

STARTSWITH

構文

STARTSWITH (object, arg1)

マクロの説明

文字列オブジェクトが arg1 の文字列で始まる場合に真を返します。

- object は文字列型で、arg1 も文字列型です。
- Java の startsWith ストリング関数と同様です。

例：

```
status = ifDescr.startsWith("unrouted vlan");
```

SYSTIME

構文

SYSTIME()

マクロの説明

現在のシステム時刻を返します（ミリ秒単位)。

例：

```
now = systime();
```

TABLEINDICES

構文

TABLEINDICES (object)

マクロの説明

テーブル内の行を特定するために使用する値として object を設定します（二重引用符で囲まれ、カンマで区切られた変数のリスト)。

例：

```
tableIndices("serviceTypeNat");
```

TOLOWERSTRING

構文

TOLOWERSTRING (object)

マクロの説明

すべて小文字の文字列形式でオブジェクトを返す

例：

```
ifDescr6 = ifDescr6.ToLowerString(ifDescr6);
```

TOPN

構文

TOPN (object, arg1, arg2)

マクロの説明

降順ソート キーでソートされた object の最上部 n（arg2）行を返す

- PollDefinition セクションまたは ProcessDBSummary セクションで使用されます。
- object はテーブル
- arg1 はソートする列
- arg2 は取得するレコード数（n）
- ProcessDBSummary セクションの Filter セクション内で使用される場合、rows は、一般に、ProcessDBSummary の実行が完了したときに設定されるオブジェクトとして使用可能な暗黙の変数です。

例：

```
rows = rows.topN("ProcessCPUUtil5mAvg", 5);
```

TOSTRING

構文

TOSTRING (object)

マクロの説明

object を文字列形式で返します。

例：

```
probeIndex = "IPSLA " + ToString(rttMonCtrlAdminIndex);
```

TOUPPERSTRING

構文

TOPUPPERSTRING (object)

マクロの説明

すべて大文字の文字列形式でオブジェクトを返します。

例：

```
ifDescr6 = ifDescr6.ToUpperString(ifDescr6);
```

VIEWDESCENDANT

構文

VIEWDESCENDANT (arg1)

マクロの説明

ノードが指定されたビュー名の子ノードかどうかをテストします。子ノードの場合は true を返し、それ以外の場合は false を返します。

- arg1 は、次のの形式のビュー名です。
persistedViewlFileName.persistedViewlFileName.viewname.subviewname
- persistedViewFileName はユーザがアクセス可能かどうかによって異なります。有効になっていないと、ビュー名の最初の 2 レベルはビューが最初に作成されたデバイスです。有効の場合は、ユーザの ID です。
- 通常はデバイスの範囲を制限するためにグループ内およびしきい値定義で使用されます。

例：

```
Algorithm =
If(ViewDescendant("dhcp-64-102-86-126-cisco-com_._dhcp-64-102-86-126-cisco-com_._andy"),
true, false)
```

XMLPOLL

構文

XMLPOLL (arg1, arg2, arg3, arg4, arg5, arg6)

マクロの説明

レポート データに基づいた CLI を取得するために PollDefinition のセクションでのみ使用されます。

- arg1 は PAL のコールに対するパッケージ ID です。
- arg2 は PAL のコールに対するアクション名です。
- arg3 は PAL のコールに対する入力パラメータです。
- arg4 値は PAL のコールに対してポーリングされます。
- arg5 はテーブル インデックス値のキーです。
- arg6 はスタティックまたはダイナミックな設定で、結果がキャッシュされるかどうかを示します。
- arg7 はキャッシュ キーです。

例：

```
XmlPoll("y1731Stats",
"y1731Stats.y1731Config",
"1001",
"integer:rttMonCtrlAdminIndex,
octetstring:ipslaType,
octetstring:operationType,
octetstring:ipslaFrameType,
octetstring:cfmDomain,
integer:evc,
integer:cfmTargetMpid,
integer:cfmSourceMpid,
octetstring:targetMACaddress,
octetstring:sourceMACaddress,
integer:cos,
octetstring:clock",
"rttMonCtrlAdminIndex",
true,"xmlY1731ConfigTable");
```

XMLPOLLNEXT

構文

XMLPOLLNEXT (arg1, arg2, arg3, arg4, arg5, arg6, arg7, arg8)

マクロの説明

サブ キーに基づいた CLI レポート データを取得するために PollDefinition のセクションでのみ使用されます。

- arg1 は PAL のコールに対するパッケージ ID です。
- arg2 は PAL のコールに対するアクション名です。
- arg3 は PAL のコールに対する入力パラメータです。
- arg4 値は PAL のコールに対してポーリングされる必要があります。
- arg5 はテーブル インデックス値の主キーです。
- arg6 はテーブル インデックスの値のサブ キーです。
- arg7 は開始時間の調整用のキーです。
- arg8 はキャッシュ キーです。

例：

```
XmlPollNext("y1731Stats",
"y1731Stats.y1731Hist",
"1001",
"integer:rttMonCtrlAdminIndex,
datetime:startTime,
datetime:endTime,
gauge:operationInitiatedNum,
gauge:operationCompletedNum,
gauge:delayTwoWayNum,
gauge:delayTwoWayMin,
gauge:delayTwoWayAvg,
gauge:delayTwoWayMax,
gauge:delayVarianceTwoWayPosNum,
gauge:delayVarianceTwoWayMinPos,
gauge:delayVarianceTwoWayAvgPos,
gauge:delayVarianceTwoWayMaxPos,
gauge:delayVarianceTwoWayNegNum,
gauge:delayVarianceTwoWayMinNeg,
gauge:delayVarianceTwoWayAvgNeg,
gauge:delayVarianceTwoWayMaxNeg,
gauge:delayForwardNum,
gauge:delayForwardMin,
gauge:delayForwardAvg,
gauge:delayForwardMax,
gauge:delayVarianceForwardPosNum,
gauge:delayVarianceForwardMinPos,
gauge:delayVarianceForwardAvgPos,
gauge:delayVarianceForwardMaxPos,
gauge:delayVarianceForwardNegNum,
gauge:delayVarianceForwardMinNeg,
gauge:delayVarianceForwardAvgNeg,
gauge:delayVarianceForwardMaxNeg,
gauge:delayBackwardNum,
gauge:delayBackwardMin,
gauge:delayBackwardAvg,
gauge:delayBackwardMax,
gauge:delayVarianceBackwardPosNum,
```

```
gauge:delayVarianceBackwardMinPos,
gauge:delayVarianceBackwardAvgPos,
gauge:delayVarianceBackwardMaxPos,
gauge:delayVarianceBackwardNegNum,
gauge:delayVarianceBackwardMinNeg,
gauge:delayVarianceBackwardAvgNeg,
gauge:delayVarianceBackwardMaxNeg,
gauge:lossForwardNum,
gauge:lossForwardAvailableNum,
gauge:lossForwardUnavailableNum,
gauge:lossForwardTxFrms,
gauge:lossForwardRxFrms,
percent:lossForwardMinFLR,
float:lossForwardAvgFLR,
percent:lossForwardMaxFLR,
float:lossForwardCumFLR,
gauge:lossBackwardNum,
gauge:lossBackwardAvailableNum,
gauge:lossBackwardUnavailableNum,
gauge:lossBackwardTxFrms,
gauge:lossBackwardRxFrms,
percent:lossBackwardMinFLR,
float:lossBackwardAvgFLR,
percent:lossBackwardMaxFLR,
float:lossBackwardCumFLR",
"rttMonCtrlAdminIndex",
"endTime",
"startTime",
"xmlY1731StatsTable");
```

XMLPOLLPERSIST

構文

XMLPOLLPERSIST (arg1)

このマクロは、CLI のポーリングによってシステム機能の検出のために設計されています。このマクロは、CLI ポーリング機能（除外）を検出するために SystemCapability.xml だけを使用します。

- arg1 は etc/palRuntime/conf/DeviceCapability.xml で定義されている機能名です（追加）。

例：

```
BGP4_SUMMARY = xmlPollPersist("bgpIpv4Summary");
```