IBM Endpoint Manager
バージョン 9.2

アクション・ガイド

IBM Endpoint Manager
バージョン 9.2

アクション・ガイド

IBM

お願い

本書および本書で紹介する製品をご使用になる前に、 75 ページの『特記事項』に記載されている情報をお読みください。

本書は、IBM Endpoint Manager バージョン 9 リリース 2 モディフィケーション・レベル 0、および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： IBM Endpoint Manager
Version 9.2
Action Guide

発行： 日本アイ・ビー・エム株式会社

担当： トランスレーション・サービス・センター

目次

第 1 章 アクション言語の紹介

Fixlet によってコンピューター上の潜在的な問題が検出されると、IBM Endpoint Manager シェル・コマンド (アクション・スクリプト) を使用した修正が提案されます。スクリプトを作成する方法は他にもありますが、最も効率的なのは **IBM Endpoint Manager Action Language** を使用する方法です。これを使用すれば、関連エンジンとの統合が強固になるからです。

多くのアクション・コマンドには、パラメーターを使用できるコマンドや、パラメーターが必須のコマンドがあります。これらのパラメーターは、ハードコーディングされた (静的な) 値または式にすることができます。これらの値や式は IBM Endpoint Manager の関連エンジンによって評価され、挿入されます。これらは**置換変数**と呼ばれます。この置換変数により、対象範囲が詳細に指定された、非常に柔軟なスクリプトを作成することができます。アクションをトリガーした正確な関連式をアクション・スクリプトで使用することができます。これにより、問題と修正を完全に一致させることができます。すべてのコマンドは、その処理前に引数での置換を実行できますが、一部例外もあります。

この本書では、すべての IBM Endpoint Manager アクション・コマンドを、具体的な例を挙げて説明します。各アクション・コマンドのトピック末尾にはバージョン番号を示しています (バージョン 7.2 以降など)。これは、該当するコマンドに対応した最小バージョンを表しています。また、一部のアクションは「Windows のみ」と記されており、UNIX または Macintosh システムでは実行できません。

アクション・スクリプトの作成

カスタム・アクションを作成して、標準のコンテンツではカバーされない問題を、ネットワーク全体にわたって修正または処理することができます。プロセスの説明自体は単純なものですが、自由に使用できるアクションと対象となる手法にはさまざまなものがあります。カスタム・アクションを作成するには、以下の手順を実行します。

1. IBM Endpoint Manager コンソールに「マスター オペレータ」としてログオンします。
2. 「**ツール**」>「**カスタム アクションの実行**」を選択します。
3. 「**アクションの実行**」ダイアログがポップアップ表示されます。このダイアログの上部は、カスタム・アクションの**名前**を指定する場所です。このフィールドはソートやフィルタリングが可能であるため、効果的な命名規則を使用することで、レポートを最大限に活用できます。
4. 「名前」フィールドの下には「**プリセット**」プルダウン・メニューがあります。これを使用すると、プリセットされたカスタマイズ済みアクションを選択できるため、時間を節約して正確性を確保することができます。また、後で使用するために、現在の入力内容をプリセットとして保存することもできます。「プリセット」インターフェースには、以下のフィールドとボタンがあります。
   a. **プリセット:** プルダウン・メニューからプリセットを選択します。

b. **個人用プリセットのみ表示:** プリセットのリストをフィルタリングして個人用のプリセットのみ表示するには、このチェック・ボックスを選択します。

c. **プリセットを保存:** 後で使用できるように、現在の一連のアクション・オプションを保存します。このボタンは、このダイアログのいずれかの場所でいずれかのオプションを変更しない限り、アクティブになりません。このボタンをクリックすると、プリセットの名前を入力するダイアログがポップアップ表示されます。下にあるチェック・ボックスでは、プリセットを保存する方法としてパブリックまたはプライベートのいずれかを選択できます。

d. **プリセットの削除:** 対象のプリセットを選択可能リストから削除します。このボタンを選択すると確認ダイアログが表示され、このコマンドの実行をキャンセルすることもできます。

5. 「プリセット」の下には、以下に示すいくつかのタブがあります。

a. **対象:** 表示されているリストから対象を選択するか、プロパティーまたは特定のコンピューター・リストを使用してアクションの対象を指定します。

b. **実行:** 適用オプションと制約 (繰り返し適用や障害復旧など) を指定します。

c. **ユーザー:** ユーザーが存在する場合または存在しない場合の、このアクションの動作を決定します。

d. **メッセージ:** アクションに先行して表示されるメッセージや、アクションと同時に表示されるメッセージを指定します。

e. **提案:** アクション提案を作成します。これにより、ユーザーはアクションを適用するかどうかを選択できます。

f. **ポスト・アクション:** アクションを完了するために必要なアクションを記述します (再起動やシャットダウンなど)。

g. **適用条件:** 元のアクション関連度をオーバーライドできます。

h. **成功条件:** アクションが成功したかどうかの判別に使用できる特定の基準を作成します。

i. **アクション・スクリプト:** このタブでは、アクション・スクリプトを作成または変更できます。

6. 「**アクション・スクリプト**」タブをクリックし、スクリプトを入力します。本書では、使用可能なアクション・コマンドについて説明し、複数の例を示します。

1. アクション・スクリプトの対象を詳細に調整するには、「**適用条件**」タブをクリックします。関連言語について詳しくは、「***IBM Endpoint Manager Relevance 言語リファレンス***」および「***IBM Endpoint Manager Inspector Guides***」を参照してください。
2. 「**実行**」、「**ユーザー**」、「**メッセージ**」、「**提案**」または「**ポスト アクション**」の各タブをクリックして、アクションを詳細にカスタマイズします。
3. カスタム・アクションをデプロイする準備ができたら、「**OK**」をクリックします。
4. カスタム・アクションは、選択された (または対象となっている) すべてのコンピューターに適用されます。これらのアクションは、指定した制約とスケジュールに基づいて適用されます。

「**タスクの作成**」または「**Fixlets の作成**」からも、アクションを作成することができます。これらのトピックの詳細については、IBM Endpoint Manager コンソールに関する資料を参照してください。

プリフェッチ・ブロックについて

プリフェッチ・ブロックは、アクション・スクリプト内の最初のエントリーである必要があります (コメントとブランク行は除く)。プリフェッチ・ブロックには、後続のアクション実行の準備に必要なすべてのダウンロード・プリフェッチ・ロジックが含まれています。これにより、アクションを理解しやすくなります。プリフェッチ・ブロックで使用できる方式には、次のようなものがあります。

リテラル・ダウンロード
これは通常の静的ダウンロードで、現在も引き続き使用することができます。

条件付きダウンロード
TRUE 条件パス内のコマンドのみ実行されます。

変数置換
この方式には、関連度置換を使用して収集するファイルを決定するダウンロードが含まれます。

カスタム・ロジック
この方式は、プラグインを利用して、ダウンロード・マニフェストを作成します。

従来のダウンロード・アクションで使用されていた事前解析アルゴリズムとは異なり、プリフェッチ・ブロック・ダウンロードはトップダウン・アプローチとして考えることができます。この場合、最初にプリフェッチ・ブロックが処理され、残りのアクションを続行する前にこのブロックが正常に完了する必要があります。これにより、管理性、柔軟性、機能性が向上しています。

注: アクションごとに使用できるプリフェッチ・ブロックは 1 つだけです。このブロックを使用する場合、**begin prefetch block** コマンドが、スクリプト内で最初の実行可能ファイルになっている必要があります。これより前に指定できるのは、ブランク行とコメントだけです。プリフェッチ・ブロックを終了するには、**end prefetch block** コマンドが必要です。

置換の使用

Fixlet 作成者は、置換を利用することで、アクション内に関連式を組み込むことができます。これを行うには、以下のように関連式を中括弧で囲みます。

```
run "{pathname of regapp "excel.exe"}"
```

この例では、プログラムの場所を指定することなくプログラムを実行しています。関連式により、「regapp」インスペクターを使用してパス名が自動的に評価されます。

```
pause while {exists running application "c:¥updater.exe"}
```

このアクションは、`running application` インスペクターを使用して、プログラムの実行が完了するまで一時停止します。

置換は再帰的な処理ではありませんが、特定のコマンドには、実行前に評価する式を 1 つ以上指定することができます。IBM Endpoint Manager クライアントは、中

括弧内に単一の式が存在することを想定します。以下のように、右中括弧より前に別の左中括弧が存在する場合、この括弧は通常の文字として処理されます。

```
echo {"a left brace: {"}
```

この例では、以下のストリングが出力に送信されます。

```
a left b  race: {
```

そのため、左中括弧を表す場合は、特殊なエスケープ文字は必要ありません。置換を終了することなくリテラルの右中括弧を出力するには、以下のように右中括弧を 2 つ指定します。

```
echo {"{a string inside braces}}"}
```

この例では、以下のストリングが出力に送信されます。

```
{a string inside braces}
```

次の例についても考えてみます。

```
appendfile {{ name of operating system } {name of operating system}
```

この例を解析した場合、二重の左中括弧は、その後に続く値が関連式ではないことを意味します。関連式の外側では、一重の右中括弧のみ必要です (関連式の内側でリテラルの右中括弧を指定する場合は、二重の右中括弧が必要です)。この例では、以下の行が `__appendfile` に出力されます。

```
{ name of operating system } WinXP
```

以下のように、**add prefetch item** コマンドが指定された置換を、プリフェッチ・ブロックで使用することもできます。

```
begin prefetch block
      parameter "manifest"="{pathname of file "manifest.spec" of client folder
        of site "AV"}"
      add prefetch item {concatenation " ; " of lines of file
        (parameter "manifest")}
end prefetch block
```

動的ダウンロードについて

動的ダウンロード機能により、アクション・スクリプトの柔軟性が拡張されます。既存の静的ダウンロード方式について理解すると、この新機能の動作を理解するのに役立ちます。

静的ダウンロード

IBM Endpoint Manager クライアントは、アクションを実行する前に、ダウンロード・コマンドまたはプリフェッチ・コマンドを検索して、実行対象のアクションを解析します。静的ダウンロードには、アクション・スクリプト内のリテラル値として、各項目の URL、SHA ハッシュ・アルゴリズム、およびサイズが含まれます。このリテラル値により、オペレーターは、アクション・スクリプトの実行内容を正確に監視することができます。これらのリテラルを使用して、アクションに関連付けられているダウンロードの番号付きリストが構成され、IBM Endpoint Manager サーバーに保管されます。アクション処理のこのステージを、**プリフェッチ処理**と呼びます。

プリフェッチ処理の結果として、クライアントは <actionid>/0 で終わる URL を要求することで、ダウンロードの必要があることを最も近い IBM Endpoint Manager リレーに通知します。その結果、通知を受信したリレーにより、指定されたアクションに対応するすべての項目がダウンロードされます。準備が整うと、リレーは当該アクション ID を使用して、クライアントに対して ping を実行します。次に、このアクションを実行するすべての IBM Endpoint Manager クライアントは、<actionid>/1、<actionid>/2 のような形式で 1 つずつファイルを確認して、ファイルの収集を行います。

ただし、ダウンロード情報はリテラル式で表されるため、当該アクションの作成時に既に判明していた URL のみ使用することができます。そのため、ダウンロードが変更されていてアクション・スクリプトが変更されていないインスタンスの場合、静的ダウンロードは使用できないことになります。

動的ダウンロード

動的ダウンロードの導入により、関連句を使用してダウンロードを指定できるようになりました。この新規コマンドは、アクション・コードの特殊なセグメント **(プリフェッチ・ブロック)** に組み込む必要があります。例えば、以下のように、最初の行に名前付き変数が含まれている AV Fixlet サイト (download.spec) でファイルを作成した場合を考えてみます。

```
name=update.exe sha1=123 sha256=678 size=456 url=http://site.com/download/patch.exe
```

このパッチには、以下のように、プリフェッチ・ブロックで関連度置換を使用してアクセスすることができます。

```
begin prefetch block
      parameter "downloadFile"="{pathname of file "download.spec" of client folder
        of site "AV"}"
      add prefetch item {line 1 of file (parameter "downloadFile")}
end prefetch block
```

このコード・ブロックにより、AV サイト内のファイルを参照する downloadFile という変数が作成されます。次に、後続のダウンロード用に、このファイルがプリフェッチ・キューに追加されます。このように、AV サイト内の Fixlet メッセージにより、対象を継続して自動的に更新するための機能が提供され、新規バージョンが使用可能になるたびに download.spec ファイルが更新されます。Fixlet からポリシー・アクションとしてアクションを適用すると、download.spec ファイルが変更されるたびに更新が実行されます。

このコード・ブロックは、**end prefetch block** コマンドで終了します。これにより、アクション・スクリプトの実行前に、ファイルが正常にダウンロードされます。プリフェッチ・ブロックは、アクション・スクリプトの先頭に配置する必要があります。スクリプトを続行するには、end prefetch block ステートメントでプリフェッチ・ブロックを終了する必要があります。

一般的な他の手法としては、それぞれが独自の URL、SHA ハッシュ・アルゴリズム、サイズを持つ複数のダウンロードのリストが格納されたデータ・ファイル (マニフェスト) を使用する方法があります。このマニフェストは必要に応じて何回でも変更できるため、スパイウェア定義やアンチウィルス定義を簡単に更新すること

ができます。これを実装する場合、以下のようなダウンロードのリストが記述された manifest.spec というファイルを作成する方法があります。

```
name=patch1.exe sha1=123 sha256=347 size=456 url=http://site.com/download/patch1.exe
name=patch2.exe sha1=234 sha256=358 size=567 url=http://site.com/download/patch2.exe
name=patch3.exe sha1=345 sha256=368 size=678 url=http://site.com/download/patch3.exe
You can then download these patches with a prefetch block that pulls these
   files from the manifest:
begin prefetch block
      parameter "manifest"="{pathname of file "manifest.spec" of client folder
        of site "AV"}"
      add prefetch item {concatenation " ; " of lines of file
        (parameter "manifest")}
end prefetch block
```

小規模な実行可能ファイルを使用して、ファイルを新たなマニフェスト内で処理することもできます。これを行うには、**execute prefetch plug-in** コマンドを使用します。以下に例を示します。

```
begin prefetch block
      add prefetch item name=myPlugIn.exe sha1=123 size=456
          url=http://mysite/plugin.exe sha2=347
      // collect the plug-in before continuing:
      collect prefetch items
      parameter "ini"="{file "prepass.ini" of site (value of setting
         "CustomSite") of client}"
      execute prefetch plug-in "{download path "myPlugIn.exe"}" /downloads
         "{parameter "ini"}" "{download path "manifest"}"
      add prefetch item {concatenation " ; " of lines of download file
         "manifest"}
end prefetch block
```

このプリフェッチ・ブロックは、プラグインをプリフェッチ・キューに追加してから、**collect prefetch items** コマンドを実行します。これにより、プリフェッチ・キューに追加された項目がダウンロードされるまで、プリフェッチ処理は行われません。ダウンロードされると、プリフェッチ処理が続行されます。正常にダウンロードされると、対象のデータ・ファイルと、そのデータ・ファイルから取得するマニフェストのパスが含まれる引数を使用して、プラグインが実行されます。最後の **add prefetch item** コマンドは、新規に作成されたマニフェスト内で指定されているダウンロードを順に待機させます。このような手法を使用して、保護ファイルをプレーン・テキストのマニフェスト内に復号化することもできます。

動的ダウンロードでは、サイズまたは SHA ハッシュ・アルゴリズムを確認してファイルを指定する必要があります。URL、サイズ、SHA ハッシュ・アルゴリズムは、アクション・スクリプトの外部にあるソースから取得可能です。このような柔軟性があるため、十分な注意が必要です。すべてのクライアントで動的ダウンロードを使用してファイルを要求できるため、ユーザーがサーバーを使用して無差別にファイルをホストしてしまう可能性があります。これを防ぐため、動的ダウンロードではホワイト・リストが使用されます。URL (アクション・スクリプトでリテラル URL を使用して明示的に許可されていないもの) からダウンロードを行うすべての要求は、IBM Endpoint Manager サーバー上の URL のホワイト・リストに指定されているいずれかの条件を満たしている必要があります (<BES Server Install Path>¥Mirror Server¥Config¥DownloadWhitelist.txt)。このファイルには、Perl regex 形式を使用した正規表現のリストが改行で区切られて記述されています。以下に例を示します。

```
http://.*¥.sitename¥.com/.*
http://software¥.sitename¥.com/.*
http://download¥.sitename¥.com/patches/JustThisOneFile¥.qfx
The first line is the least restrictive, allowing any file at the sitename
domain to be downloaded. The second line requires a specific domain host
and the third is the most restrictive, limiting the URL to a single file
named "JustThisOneFile.qfx". If a requested URL fails to match an entry in
the white-list, the download immediately fails with status NotAvailable.
```

失敗した URL を示す注記が、リレー・ログに記録されます。
ホワイト・リストが空の場合や、存在しない場合、すべての動的ダウンロードが失敗します。
ホワイト・リストの項目「.*」(ドットと星印) は、すべての URL についてダウンロードを許可することを意味します。

プリフェッチ・ブロックでは、以下のように条件ステートメントを使用することができます。

```
begin prefetch block
                  if {name of operating system = "Windows 2000"}
add prefetch item name=up.exe sha1=123 size=456
    url=http://site.com/patch2k.exe sha2=567
                  else
add prefetch item name=up.exe sha1=123 size=456
    url=http://site.com/patch.exe sha2=567
                  endif
end prefetch block
wait "{download path "up.exe"}"
```

このアクション・スクリプトは、Windows 2000 の存在によって分岐しますが、この例におけるダウンロードは静的に (リテラル・テキストとして) 記述されています。クライアントは必要な特定の項目だけをダウンロードしますが、要求された場合はすべての静的ファイルがサーバーとリレーに即座にダウンロードされます。動的ダウンロードの場合、このような状況においてパフォーマンスを向上させることができます。これは、クライアントが実際に必要とするファイルだけが、最初にサーバーとリレーにフェッチされるためです。動的ダウンロードを使用した例を以下に示します。

```
begin prefetch block
                  if {name of operating system = "Windows 2000"}
add prefetch item {"name=up.exe sha1=123 size=456
    url=http://site.com/patch2k.exe"} sha2=567
                  else
add prefetch item {"name=up.exe sha1=123 size=456
    url=http://site.com/patch.exe"} sha2=567
      endif
end prefetch block
wait "{download path "up.exe"}"
```

サーバー上で適切に構成されたホワイト・リスト・ファイルとともに、関連度置換をプリフェッチ・ブロックで使用することにより、このコードは必要なファイルだけを取得します。これにより、必要な帯域幅を節約して、処理効率を高めることができます。

ファイルの内容に応じて処理の実行を分岐することもできるため、更新を自動化することができます。これは、バージョン番号の変更処理を行う場合に特に便利です。例えば、以下のような 2 つの名前付き変数を含むファイル「manifest.txt」を作成したとします。

```
version=1234
download=name=update.exe sha1=123 size=456
   url=http://site.com/download/patch.exe sha2=567
Note that the download variable contains the name, sha1, sha2, size and URL
```

```
of the patch file.
You can then use relevance substitution to extract these variables with
an expression such as:
parameter "ver"="{key "version" of file "{download path "manifest.txt"}"}"
parameter "filename"={key "download" of file "{download path "manifest.txt"}"}
By comparing the extracted version against some stored values, you can determine
if and when you need to download the specified file. この手法を応用して、複数の
```
バージョンを処理対象に含めたり、パッチと完全置換の更新を区別したりすることもできます。

どのような手法を使用する場合でも、ファイルを一度ダウンロードすれば、各種インスペクターを使用してファイルを検証することができます。アクションがアクティブであるかどうか (またはプリフェッチ処理内かどうか) に応じて、各インスペクターでの解釈が異なる場合があります。実行前に、これらのファイルはプリフェッチ・フォルダー内に収集されます。アクション実行中は、これらのファイルは __Download フォルダー内に存在します。

また、以下のように、アクション実行前 (あるいはアクション実行中) にファイルの場所を特定できる新規インスペクターもあります。

- **download folder:** プリフェッチ解析中に、このインスペクターは、__Global¥<sitename>¥<actionid>¥named フォルダーからフォルダー・オブジェクトを返します。アクションがアクティブになり、ダウンロードが完了すると、このインスペクターは目的のフォルダー・オブジェクトを __Download ディレクトリーから返します。
- **download path "pathname":** このインスペクターは、それが存在するかどうかにかかわらず、指定されたファイルへの絶対パス名を含むストリングを返します。ダウンロード・ファイル名は、「(ダウンロード・フォルダーのパス名) & <パス区切り文字> & ファイル名」となります。
- **download file "filename":** このインスペクターは、ダウンロード・フォルダーまたは別の指定フォルダーからファイル・オブジェクトを返します。ダウンロード・ファイル名は、「ダウンロード・フォルダーのファイル 'filename'」となります。ファイルがまだダウンロード・フォルダーに存在しない場合、インスペクターは「does not exist」を返します。

これらのアクションのユーザーを保護し、ダウンロードとそのチェックサムがセキュリティー侵害されていないことを確認するかどうかは、アクション・スクリプトの作成者が決定します。介入者攻撃を防ぐことができるエンドツーエンドの認証メカニズムが、最も効果的な防御手段となります。動的ダウンロード・アクションを作成する場合、通常は前述のプラグインなどを使用して、情報を使用する前にその情報を認証するようにアクションを作成することが重要になります。また、このような認証を実行するアクション・スクリプトの手順を明確に把握しておくことも重要です。これにより、アクションのユーザーは、そのメカニズムを検証してから、そのアクションを信頼するかどうかを判断することができます。

注: アクションごとに使用できるプリフェッチ・ブロックは 1 つだけです。このブロックを使用する場合、**begin prefetch block** コマンドが、スクリプト内で最初のコマンドになっている必要があります。これより前に指定できるのは、ブランク行とコメントだけです。プリフェッチ・ブロックを残りのアクションから分離するには、**end prefetch block** コマンドが必要です。

第 2 章 実行コマンド

このセクションでは、実行コマンドについて説明します。

action launch preference low-priority

このコマンドを実行すると、プログラムを起動する後続のアクション・コマンドが、通常より低い優先順位で実行されます。これにより、大規模なパッチやサービス・パック・アップグレードによる影響を軽減することができます。

low-priority preference は、現行アクションから起動されるアプリケーションの起動優先順位にのみ影響します。この設定は、現行アクションが完了するか、クライアントが **action launch preference normal-priority** コマンドを実行するまで維持されます。

構文

```
action launch preference low-priority
```

例

```
action launch preference low-priority
run "{pathname of regapp "background_app.exe"}"
action launch preference normal-priority
```

この例では、background_app の実行前に起動優先順位が低くなるため、実行時にシステムが占有されることはありません。その後、優先順位が通常のレベルに戻されます。

th

注: このコマンドは Windows 専用です。UNIX エージェントでは、アクション・スクリプトが終了します。

バージョン 6.0 以降 -- Windows のみ

action launch preference normal-priority

このコマンドを実行すると、プログラムを起動する後続のアクション・コマンドが、通常の優先順位で実行されます。このステートメントは、**action launch preference low-priority** コマンドの後で通常の優先順位に戻す場合のみ必要です。

構文

action launch preference normal-priority

例

```
action launch preference low-priority
run "{pathname of regapp "background_app.exe"}"
action launch preference normal-priority
```

この例では、background_app の実行前に起動優先順位を低くしてから、後続の起動ステートメントについて通常の優先順位に戻しています。

注

このコマンドは Windows 専用です。UNIX エージェントでは、アクション・スクリプトが終了します。

バージョン 6.0 以降 -- Windows のみ

dos

このコマンドは、標準の DOS コマンドを発行します。DOS コマンドが失敗すると、このコマンドが記述されているアクション・スクリプトが終了します。

構文

dos <DOS command line>

例

```
dos rmdir /Q /S "{pathname of windows folder & "¥temp"}"
```

この例では、Windows ディレクトリー内の一時フォルダーから空のディレクトリーを削除しています。

```
dos scandisk.exe e:
```

この例の「e:」は、スキャン・ディスク・プログラムに渡されるパラメーターです。

注

このコマンドは Windows 専用です。UNIX エージェントでは、アクション・スクリプトが終了します。

Windows システムでは、このコマンドは、Windows API からシステム (DOS コマンド行構文) ステートメントを発行する場合と同じ動作になります。また、DOS プロンプトに DOS コマンド行を入力する場合とも、同じ動作になります。DOS コマンドは、PATH 環境変数を使用して、ユーザーのハード・ディスク上でコマンドの場所を特定しようとします。他の DOS コマンドと同様、その他の場所については、絶対パス名を指定する必要があります。

ファイル名にスペースが含まれる場合は、必ず引用符を使用してください。

バージョン 5.1 以降 -- Windows のみ

notify client ForceRefresh

このコマンドは、IBM Endpoint Manager コンソールでクライアント・コンピューターを右クリックし、「**更新を送信**」を選択した場合と同じ動作になります。IBM Endpoint Manager クライアントへの UDP 接続がブロックされる場合、このコマンドが必要になることがあります。

構文

notify client ForceRefresh

バージョン 6.0.14 以降

override

override コマンドでは、特定のコマンドをカスタマイズすることができ、既存のコマンドに複数のバリエーションを与えることができます。この強力な複合コマンドにより、ユーザーは既存のコマンド waitdetached や runhidden に似た独自のカスタム組み合わせコマンドを作成できます。既存のコマンドに制約を追加するには、コマンド本体内に事前定義済みのキーワードと値のペアを追加します。

構文

```
override <cmd>
<keyword>=<value>
<keyword>=<value>
<cmd> <rest of command line>
```

この **cmd** は wait または run のいずれかであり、キーワードと値のペアは『使用上の注意』セクションに示されている表から選択されます。

例

```
override wait
hidden=true
wait notepad.exe
```

この例では、waithidden notepad.exe と同じ機能が提供されます。

```
override wait
completion=job
hidden=true
runas=currentuser
wait __Download¥patch.exe arg1 arg2 arg3
```

この例は、現行ユーザーが非表示プロセスとしてパッチを実行し、ジョブが完了するまで待機してからアクション・スクリプトを続行する方法を示しています。

使用上の注意

キーワードは任意の順序で指定できますが、1 行に 1 つしか指定できません。「=」(等号) の前後に空白は必要ありません。空白は無視されます。

キーワードは大/小文字を区別しません。関連度置換の場合、値を中括弧 ({}) で囲むことができます。重複するキーワードがリストされている場合は、最後の値が使用されます。キーワードまたは値のいずれかが無効の場合は、コマンド全体が失敗します。プラットフォーム固有のキーワードであり、指定のプラットフォームでは意味のないものは、暗黙的に無視されます。ほとんどのコマンドと同様、コマンド・ストリングは関連度評価を通じて実行され、コンソール側での構文検査は最小限にとどまります。ただし、エージェントはアクション実行時に全検査を行います。

使用可能なキーワードとその値を、以下の表に示します。

キーワード	許容値	デフォルト値
completion	none | process | job	none | process *
priority	normal | low	normal
hidden	false | true	false
detached	false | true	false
runas	agent | currentuser	agent
* run の場合のデフォルトは none で、wait の場合のデフォルトは process です。		

Completion

- Completion=**none**。現行の run コマンド・バリアントと同様に機能します。
- Completion=**process**。現行の wait コマンド・バリアントと同様に機能します。
- Completion=**job** (Windows 上)。Windows JobObject を利用して、ターゲット・プロセスに対するいくつかの制限と、コマンドのいくつかの潜在的な障害点とを適用します。詳しくは下記を参照してください。

「completion=job」に関する制限

Windows:

プロセスに対して実行されるジョブ制御を最大限柔軟なものにするためには、override コマンドを使用して、プロセスで子プロセスをジョブから選択的に除外します。これにより、プロセスは独自のジョブ制御管理を実行することができますが、その除外された子のいずれかがジョブ・オブジェクトから削除されます。

起動されたプロセスが独自のジョブ制御を実行するという限られた状況では、その子プロセスがすべて完了するまで、ジョブのメンバーが実行中のままになることが想定されます。ただし、必ずしもこのようになるとは限りません。これに該当しない場合もあります。そのような場合は、子プロセスがまだ実行中であっても、アクションは完了します。

UNIX/Linux:

UNIX/Linux プラットフォームでは、ジョブ処理の管理にセッション ID が使用されます。セッション ID には、セッション・リーダー (起動するプロセス) のプロセス ID の値が使用されます。「completion=process」の場合と同様に、クライアントはリーダー・プロセスが終了するのを待機してから、0.5 秒のスリープ・サイクルを開始し、その後、ジョブ・リーダーのプロセス ID と一致するセッション ID を持つすべての対象を検索するプロセスを列挙します。こうしたプロセスが存在しなくなったら、ジョブは完了し、コマンドは終了します。

コマンドが返す終了コードは常にリーダー・プロセスのものであり、完了する最後のプロセスのものではありません。

バージョン 8.2 以降

Priority (Windows のみ)

- Priority=**normal**。現行の「action launch priority normal」コマンドと同様に機能します。
- Priority=**low**。現行の「action launch priority low」コマンドと同様に機能します。

Hidden (Windows のみ)

- Hidden=**true**。runhidden コマンドおよび waithidden コマンドの場合と同じように、SW_HIDE 属性をプロセスに適用します。
- Hidden=**false**。プロセスから SW_HIDE 属性を削除します。

Detached (Windows のみ)

- Detached=**true**。rundetached コマンドおよび waitdetached コマンドの場合と同じように、detach メソッドを使用してプロセスを作成します。
- Detached=**false**。normal メソッドを使用してプロセスを作成します。

RunAs

- `RunAs=agent` は、現行の wait バリアントおよび run バリアントと同じプロセス所有権特性を適用します。
- `RunAs=currentuser` は、Windows 上の現行の RunAsCurrentUser.exe を模倣します。その際、同じロジックを使用して、現行ユーザーと類似コードを識別し、userToken がソースである環境ブロックを使用してプロセスを作成します。Unix/Linux では、該当するユーザー環境変数を一貫して取得できないため、必須の Xauthority 変数を除き、環境変数の適用は一切試行されません。UNIX/Linux では、XBESClientUI の現行ユーザーとして識別されたユーザーの ID に対して、setuid の呼び出しが行われます。これは、ユーザーがローカル・コンソールと実行中の X Windows でログオンする必要がある、非常に特殊なプラットフォーム依存テストです。

詳しくは、『 11 ページの『override』』セクションを参照してください。

---

run

このコマンドは、指定されたプログラムを実行します。プロセスを作成できない場合、アクション・スクリプトは終了します。run コマンドでは、プロセスの終了を待機せずに、アクション・スクリプトの次の行が実行されます。コマンド行には、実行可能ファイル名 (およびオプションでパラメーター) を指定します。あるプログラムが完了してから別のプログラムを開始するには、**wait コマンド**を使用します。

構文

run <command line>

例

```
run "{pathname of regapp "wordpad.exe"}"
run "c:¥winnt¥ftp.exe" ftp.mycorp.net
run wscript /e:vbs x.vbs arg1 arg2
```

これらの例は、スクリプトを実行する方法と、スクリプトに引数を渡す方法について示しています。コマンド行を引用符で囲むことをお勧めします。ファイル名にスペースが含まれている場合は、必ず引用符を使用してください。

注

Windows コンピューターでは、このコマンドは、<command line> を使用して CreateProcess API を呼び出す場合と同じ動作になります。これは、Windows の「ファイル名を指定して実行」ダイアログで <command line> を使用する場合とも、同じ動作になります。<command line> から実行可能ファイルを特定する方法の説明については、CreateProcess に関する Windows の資料を参照してください。

バージョン 5.1 以降

rundetached

rundetached コマンドは、Windows マシンで CreateProcess() を呼び出す際に DETACHED_PROCESS フラグを設定することで、**run** コマンドを変更します。デフォルトでは、作成されたプロセスはその親のコンソールを継承します。デタッチを行うと、この動作が禁止されます。これにより、新規プロセスでのユーザーとの対話方法をより詳細に制御することができます。

特に、これを使用することで、プログラムの実行時にポップアップ DOS ウィンドウの表示を抑制することができます。これは **run** コマンドの場合と同様ですが、作成されたプロセスは親のコンソールにアクセスしないため、混乱を招くような DOS ウィンドウは表示されません。Rundetached は、対話式プログラムの実行には使用しないでください。対話式プログラムに使用すると、そのプログラムはユーザー・インターフェースを表示できなくなるため、ハングしているように見える場合があります。そのため、このコマンドは、ユーザー・インターフェースを表示しないプログラムを実行する場合のみ使用してください。

構文

rundetached <command line>

例

```
rundetached "{pathname of regapp "background_app.exe"}"
rundetached "c:¥winnt¥ftp.exe" ftp.filesite.net
```

これらの例は、プログラムを実行して、プログラムにいくつかの引数を渡す方法について示しています。コマンド行を引用符で囲むことをお勧めします。ファイル名にスペースが含まれている場合は、必ず引用符を使用してください。

注

このコマンドは Windows 専用です。UNIX エージェントでは、アクション・スクリプトが終了します。Windows コンピューターでは、このコマンドは、Windows API から CreateProcess(CommandLine) ステートメントを発行する場合と同じ動作になります。これは、Windows の「ファイル名を指定して実行」ダイアログで

CommandLine を使用する場合とも、同じ動作になります。CommandLine から実行可能ファイルを特定する方法の説明については、CreateProcess() に関する Windows の資料を参照してください。

バージョン 5.1 以降 -- Windows のみ

runhidden

このコマンドは、CreateProcess() を使用して、非表示ウィンドウでコマンドを起動します。このコマンドは、STARTUPINFO dwFlag を STARTF_USESHOWWINDOW に設定し、wShowWindow を SW_HIDE に設定することで、ウィンドウを非表示にします。作成されるプロセスによってこのフラグを変更し、以降はウィンドウを再表示するように変更することができます。

このコマンドを起動すると、以下のアクション・コマンド行が即座に実行されます。起動が完了するのを待機してからアクションを続行するには、**waithidden** コマンドを使用します。

構文

runhidden <command line>

例

```
runhidden "{pathname of regapp "wordpad.exe"}"
runhidden "c:¥winnt¥ftp.exe" ftp.mycorp.net
runhidden wscript /e:vbs x.vbs arg1 arg2
```

これらの例は、非表示ウィンドウでスクリプトを実行する方法と、スクリプトにいくつかの引数を渡す方法について示しています。 コマンド行を引用符で囲むことをお勧めします。ファイル名にスペースが含まれている場合は、必ず引用符を使用してください。

注

このコマンドは Windows 専用です。UNIX エージェントでは、アクション・スクリプトが終了します。

起動されたプロセスでユーザー入力が必要な場合、コマンドでウィンドウを表示するよう明示的に指定されていない限り、このプロセスはウィンドウを非表示にしたまま入力を待機します。

Windows コンピューターでは、このコマンドは、<command line> を使用して CreateProcess() API を呼び出し、ウィンドウを非表示にするためのフラグを設定した場合と同じ動作になります。<command line> から実行可能ファイルを特定する方法の説明については、CreateProcess() に関する Windows の資料を参照してください。

バージョン 6.0 以降 -- Windows のみ

script

このコマンドをアクション・スクリプトと混同しないでください。script キーワードは、指定された名前を持つ外部スクリプト (JavaScript や Visual Basic などのスクリプト言語用に作成されたもの) を実行します。適切なスクリプト・エンジンがインストールされていない場合、またはスクリプトを実行できない場合、script キーワードが記述されたアクション・スクリプトは終了します。指定されたスクリプトが終了するまで、アクション・シェル・コマンドの次の行は実行されません。

構文

script<script name>

例

```
script attrib.vbs
```

Visual BASIC スクリプトの attrib.vbs を実行します。

注

このコマンドは Windows 専用です。UNIX エージェントでは、アクション・スクリプトが終了します。

Windows コンピューターでは、このコマンドは、Windows から wscript "scriptName" ステートメントを発行して完了を待機する場合と同じ動作になります。これは、Windows の「ファイル名を指定して実行」ダイアログで scriptName を使用する場合とも、同じ動作になります。スクリプトにパラメーターを渡す必要がある場合、代わりに **run** コマンドを使用してください。

バージョン 5.1 以降 -- Windows のみ

wait

wait コマンドは、**run** コマンドと同様に動作しますが、処理を続行する前にプロセスまたはプログラムの完了を待機する点で異なります。

構文

wait <command line>

例

```
wait "scandskw.exe"
```

scandskw プログラムを実行し、このプログラムが完了するまで待機してから、アクション・スクリプトを続行します。引用符で囲むことをお勧めします。ファイル名にスペースが含まれている場合は、必ず引用符を使用してください。

注

Windows コンピューターでは、このコマンドは、Windows API から CreateProcess <command line> ステートメントを発行して完了を待機する場合と同じ動作になります。

バージョン 5.1 以降

waitdetached

Waitdetached は、プログラムの完了を待機中に、ポップアップ DOS ウィンドウの表示を抑制する場合に使用します。これは **wait** コマンドと同様ですが、作成されたプロセスは親のコンソールにアクセスしないため、混乱を招くような DOS ウィンドウは表示されません。Rundetached は、対話式プログラムの実行には使用しないでください。対話式プログラムに使用すると、そのプログラムはユーザー・インターフェースを表示できなくなるため、ハングしているように見える場合があります。そのため、このコマンドは、ユーザー・インターフェースを表示しないプログラムを実行する場合のみ使用してください。

構文

waitdetached <command line>

例

```
waitdetached "scandskw.exe"
waitdetached wscript /e:vbs x.vbs arg1 arg2
```

この例は、スクリプトを実行し、スクリプトにいくつかの引数を渡し、このスクリプトが完了するまで待機してから、アクション・スクリプトを続行する方法について示しています。

注

このコマンドは Windows 専用です。UNIX エージェントでは、アクション・スクリプトが終了します。

Windows コンピューターでは、このコマンドは、Windows API から CreateProcess (CommandLine) ステートメントを発行して完了を待機する場合と同じ動作になります。

バージョン 5.1 以降 -- Windows のみ

waithidden

このコマンドは **runhidden** コマンドと類似しており、CreateProcess を使用して非表示ウィンドウでコマンドを実行します。このコマンドは、STARTUPINFO dwFlag を STARTF_USESHOWWINDOW に設定し、wShowWindow を SW_HIDE に設定することで、ウィンドウを非表示にします。このアクションは、プロセスが完了するまで待機してから、後続のアクション・コマンドを続行します。

構文

waithidden <command line>

例

```
waithidden "{pathname of regapp "notepad.exe"}"
waithidden "c:¥winnt¥ftp.exe" ftp.myurl.net
waithidden wscript /e:vbs x.vbs arg1 arg2
```

これらの例は、非表示ウィンドウでスクリプトを実行する方法と、スクリプトにいくつかの引数を渡す方法について示しています。 コマンド行を引用符で囲むことをお勧めします。ファイル名にスペースが含まれている場合は、必ず引用符を使用してください。

注

このコマンドは Windows 専用です。UNIX エージェントでは、アクション・スクリプトが終了します。

起動されたプロセスでユーザー入力が必要な場合、コマンドでウィンドウを表示するよう明示的に指定されていない限り、このプロセスはウィンドウを非表示にしたまま入力を待機します。

Windows コンピューターでは、このコマンドは、<command line> を使用して CreateProcess API を呼び出し、ウィンドウを非表示にするためのフラグを設定した場合と同じ動作になります。<command line> から実行可能ファイルを特定する方法の説明については、CreateProcess に関する Windows の資料を参照してください。

バージョン 6.0 以降 -- Windows のみ

第 3 章 フロー制御コマンド

action may require restart

このコマンドを実行すると、クライアントは、再起動が必要であるという明確なサインをシステムで検索します。このサインが存在する場合、再起動が行われるまで、アクションがコンソールで「再起動の保留中」として表示されるように、アクション完了ステータスが設定されます。再起動が完了すると、当該アクションの完了ステータスの値は、アクションの関連度の関連性がなくなった場合は「成功」となり、まだ関連性が存在する場合は「失敗」となります。

再起動の明確なサインが存在しない場合、当該アクションの完了ステータスの値は、アクションの関連度の関連性がなくなった場合は「成功」となり、まだ関連性が存在する場合は「失敗」となります。

構文

action may require restart

例

```
action may require restart
```

バージョン 5.1 以降

action parameter query

このコマンドにより、アクションの実行中に関連度を利用して、パラメーターのデータ入力を行うことができます。パラメーター名にはブランクを含めることができ、大/小文字が区別されます。パラメーター名、説明、および値はそれぞれ二重引用符 (") で囲む必要があります。一度入力すると、以降の呼び出しにおいてユーザー入力がデフォルトになります (Endpoint Manager の場合、このユーザーはデプロイメントのアクションの適用を承認するコンソール・オペレーターです)。

構文

action parameter query "<parameter name>" [with description "<description>"] [and] [with default [value] "<default value>"]

parameter name は、関連度パラメーターの名前であり、**with description** オプションを使用して、ユーザーにプロンプトを表示することができます。**and with default** オプションにより、パラメーターのデフォルト値を指定することができます。

例

```
action parameter query "InstallationPoint" with description "Please enter
  the location of the shared installation point:"
action parameter query "Registry key" with description "Please enter your
```

```
  desired registry key" and with default value "null"
action parameter query "tips" with description "Enter 'on' or 'off' to control
  Fixlet tips." With default "on" regset "[HKEY_CURRENT_USER¥Software¥BigFix]"
  "tips"="{parameter "tips" of action}"
```

注: ユーザーが入力するパラメーター値には %xx を使用できます。xx は 2 桁の 16 進数を表し、これを使用して埋め込みたい文字を指定します。% 記号を埋め込むには、%25 を使用します。二重引用符を埋め込むには、%22 を使用します。

アクションの実行中に、コンソール・オペレーターによって入力されたアクション・パラメーターの値を取得できます。例えば、アクション内で関連度置換を使用できます。例: {parameter "parameter name" of action}

関連度置換は、**action parameter query** コマンド行自体に対しては実行**されません**。これは、アクションの送信前にこのコマンドが IBM Endpoint Manager コンソールで解釈されるためです。これにより、Fixlet 作成者は、当該アクションの実行に必要なデプロイメント固有パラメーターについてオペレーターに確認することができます。

バージョン 5.1 以降

action requires login

このコマンドは、クライアントに対し、コンピューターが再起動されて管理者がログインするまで、現在のアクションは実行されないことを通知します。このアクションがマシン上で完了すると、「**ログインの保留中**」となっているインスペクターが true を返します。

構文

action requires login

例

```
action requires login
```

注

IBM Endpoint Manager UNIX エージェントは、このアクションを無視します。

バージョン 5.1 以降

action requires restart

このコマンドは、クライアントに対し、次の再起動が完了するまで、現在のアクションは実行されないことを通知します。このアクションがマシン上で完了すると、「再起動の保留中」となっているインスペクターが true を返します。アクション内に「action requires restart」コマンドが存在する場合、IBM Endpoint Manager コンソールは、影響を受けるマシンが再起動されるまで「再起動の保留中」をレポートします。

構文

action requires restart

例

```
action requires restart
```

バージョン 5.1 以降

continue if

このコマンドにより、パラメーターとして指定されている値が true に評価された場合に、スクリプト内の次の行を実行できます。指定された値が false に評価された場合、処理はエラーなしで停止します。関連度置換を使用して、値を計算することができます。このコマンドは、アクションの残りの部分を実行するための特定の条件を確実に満たす場合に便利です。アクション・スクリプトが終了した行番号は、コンソールにレポートされます。アクションで必要なインバリアントを識別する **continue if** ステートメントを挿入すると、IBM Endpoint Manager のユーザーは、この行番号を使用して、アクションが失敗した原因を特定することができます。

構文

```
continue if <true condition>
```

true condition は、評価する関連式を表します。

例

```
continue if {name of operating system = "Win2k"}
download now http://www.real-time.com/downloads/win98/dun40.exe
```

この例では、オペレーティング・システムが Windows 2000 である場合のみ、dun40.exe ファイルがダウンロードされます。

```
continue if {(size of it = 325 and sha1 of it = "013e48a5") of file
  "dun40.exe" of folder "__Download"}
wait __Download/dun40.exe /Q:A /R:N
```

この例では、size と sha1 の値が指定されたとおりの値である場合のみ、dun40.exe ファイルが実行されます。

バージョン 5.1 以降

exit

exit コマンドは、アクションを終了し、呼び出し元に整数値を返します。

```
exit {integer exit code}
```

このコマンドでは、関連度置換を使用できます。このコマンドが実行されると、整数の値がコードの終了コード・インスペクターに転送され、アクションはその行で終了します。終了コードでは、最終行に達する前にスクリプトが終了した場合に、完了実行アクション (完了するまで実行が中断されないアクション) の固定されたステータスが影響を受ける可能性があります。exit コマンドの後に実行可能な行がない場合、アクションは正常に完了します。exit コマンドの後に他のコマンドが存在する場合、完了実行アクションは失敗します。

IBM Endpoint Manager コンソールでは、最後の exit コマンドの値が「アクション情報を表示」ダイアログに表示され、その他のステータス情報も表示されます。

関連式では、「exit code of <action>」インスペクターを使用して、この値を評価することができます。

構文

exit <{expression}>

expression は、呼び出し元に返される整数値です。この値は、32 ビットの符号付き数値に制限されています。ただし、UNIX の場合、この制限は 8 ビットまたは 16 ビットのいずれかになります。この制限は、WIFEXITED マクロを実行することで判別できます。

例

```
wait 'foo'
parameter "error" = "{exit code of action}"
if {parameter "error" != "0"}
      exit {parameter "error"}
endif
// continue processing
```

この例は、ゼロ以外の終了コードの場合にエラーをレポートするスクリプトを示しています。これにより、スクリプトを途中で終了し、終了コードを呼び出し元にレポートすることができます。これは、終了コード・インスペクターの値を変更できる 4 つのスクリプト・コマンド (wait、waithidden、waitdetached、および exit) の 1 つです。3 つの wait コマンドは、実行可能ファイルに基づいて終了コードを設定します (OS により、数値のサイズが制限されます)。exit コマンドは、使用している OS に関係なく、渡された数値に応じて終了コードを符号付きの 32 ビット数値として設定します。

DOS と組み合わせて exit コマンドを使用できますが、システム・コマンド API の制限により、DOS で終了コード自体を設定することはできません。

UNIX シェル・スクリプトの注:「x-sh」タイプのアクションでは、クライアントによるシェル・スクリプトの処理が終了すると、スクリプトの終了コードがインスペクター値の内部に収集されます。UNIX シェル・スクリプトからの終了コードは、クライアント・ログに書き込まれます。

バージョン 8.0

if、elseif、else、endif

if、elseif、else、および endif の各コマンドを使用すると、アクション・コマンドを条件付きで実行することができます。これらの条件ステートメントは、以下に示すように、式を中括弧で囲むことで機能します。

```
if {EXPR1}
      statements to execute on EXPR1 = TRUE
elseif {EXPR2}
```

```
      statements to execute on EXPR1 != TRUE and EXPR2 = TRUE
else
      statements to execute when EXPR1 != TRUE and EXPR2 != TRUE
endif
```

上記のアクション例で、**if** ステートメントに続く中括弧内の式が true である場合、それ以降のステートメント (endif ステートメントまで) が評価されます。**If** ブロックは、任意の深さレベルまでネストすることができます。

通常の **if** ブロック・セマンティクスが適用されます。**endif**、**elseif**、または **else** までのすべてのステートメントが、1 つのブロックを構成します。**elseif {EXPR}** ステートメントと **else** ステートメントはオプションです。任意の数の **elseif** ステートメントを使用できますが、末尾には 1 つの **else** ブロックのみ使用できます。

プリフェッチ

IBM Endpoint Manager クライアントは、実際にアクションを実行する前に、プリフェッチ対象のダウンロードを検索して、実行対象のアクションを解析します。プリフェッチ処理で解析が適切に実行されなかった場合、**アクション構文**エラーが返され、アクションは実行されません。複数の環境で機能するアクションを作成し、if ステートメントの 1 つの分岐のみで解析が正しく行われた場合に、これが問題となる可能性があります。例えば、特定のプラットフォームに固有のファイルをロードする場合を考えてみます。

次のようなスクリプトは、正常に機能するように見えます。

```
if {not exists key "foo" of registry}
      prefetch windows_file ...

else if {not exists package "bar" of rpm}
      prefetch UNIX_file ...

endif
```

この例では、Windows レジストリー・キーが最初のプリフェッチをトリガーし、UNIX パッケージが 2 番目のプリフェッチをトリガーします。ここで問題となるのは、UNIX システムではレジストリー・インスペクターに障害が発生し、Windows ではパッケージ・インスペクターに障害が発生することです。これにより、両方の場合で、プリフェッチ・パーサーがエラーをスローします。

この問題を解決するには、以下のようにクロスプラットフォーム・インスペクター (name of operating system など) を使用して、正しくないブロックが評価されないようにします。

```
if {name of operating system starts with "Win"}
      if {not exists key "foo" of registry}
            prefetch windows_file ...
      endif

else if {name of operating system starts with "Redhat"}
      if {not exists package "bar" of rpm}
            prefetch UNIX_file ...
      endif

Endif
```

最初に適切なオペレーティング・システムを確認することで、このタイプのプリフェッチ解析エラーを回避できます。ただし、状況によっては、潜在的なエラーを回避できない場合もあります。例えば、プリフェッチ・フェーズにおいて、まだ存在しないファイルをアクションで作成してアクセスするような場合です。

```
wait chkntfs c: > c:¥output.txt
if {line 2 of file "c:¥output.txt" as lowercase contains "not dirty"}
      regset "HKLM¥Software¥MyCompany¥" "Last NTFS Check"="OK"

else
      regset "HKLM¥Software¥MyCompany¥" "Last NTFS Check"="FAIL"

endif
```

この Windows の例では、スクリプトが実際に実行されるまで、出力ファイルは存在しません。プリフェッチ・パーサーは、このファイル内容を検査するときに、ファイルが存在しないことを通知します。その後、エラーをスローしてアクションを終了します。ただし、if 条件を修正して、プリフェッチを成功させることができます。1 つの方法として、プリフェッチ検査で常に TRUE を返す「not active of action」式を使用する方法があります。この式を使用することにより、問題となるブロックを事前解析中に回避することができます。

```
wait chkntfs c: > c:¥output.txt
if {not active of action OR (line 2 of file "c:¥output.txt" as lowercase
    contains "not dirty")}
      regset "HKLM¥Software¥MyCompany¥" "Last NTFS Check"="OK"

else
      regset "HKLM¥Software¥MyCompany¥" "Last NTFS Check"="FAIL"

endif
```

アクションが事前解析または実行されているかどうかを最初に確認することで、プリフェッチを正常に実行でき、アクションの実行時に目的の動作を行うことができます。

構文

```
if {<expression>}
      <statements>
endif
```

例

```
if {name of operating system = "WinME"}
   prefetch patch1.exe sha1:e6dd60e1e2d4d25354b339ea893f6511581276fd size:4389760
     http://download.microsoft.com/download/whistler/Install/310994/WIN98MeXP/EN-US
       /WinXP_EN_PRO_BF.EXE
   wait __Download¥patch1.exe

elseif {name of operating system = "WinXP"}
   prefetch patch2.exe sha1:92c643875dda80022b3ce3f1ad580f62704b754f size:813160
     http://www.download.windowsupdate.com/msdownload/update/v3-19990518/cabpool
      /q307869_f323efa52f460ea1e5f4201b011c071ea5b95110.exe
   wait __Download¥patch2.exe

else
   prefetch patch3.exe sha1:c964d4fd345b6e5fd73c2235ec75079b34e9b3d2 size:845416
     http://www.download.windowsupdate.com/msdownload/update/v3-19990518/cabpool
      /q310507_2f3c5854999b7c58272a661d30743abca15caf5c.exe
   wait __Download¥patch3.exe

endif
```

このコード・スニペットは、オペレーティング・システムに基づいて、ファイルのプリフェッチ、名前変更、およびダウンロードを行います。

バージョン 6.0 以降

parameter

parameter コマンドを使用すると、アクションの実行中に新規のアクション変数を作成できます。形式は次のとおりです。

```
parameter "x" = "{expression}"
```

このコマンドでは、インスペクター **parameter "x"** を使用してパラメーターにアクセスできます。このパラメーターは、現行アクション内でのみ検査可能です。パラメーターは、IBM Endpoint Manager コンソールによって、アクションに追加されたヘッダーのアクションの開始直前に初期化されます。

既に値が存在するパラメーターをリセットすることはできません。リセットしようとした場合、クライアントは、パラメーターのリセットが試行された行でアクションを中止します。式の評価結果によって発生したエラーは、指定されたパラメーターを未定義の状態にすることによって処理されます。

parameter コマンドのルールは、以下のとおりです。

パラメーター式は、強制的にストリングになります。

結果の値が存在しない複数の式では、空のパラメーター値が生成されます。

結果の値が 1 つであり、強制的にストリングにすることができる複数の式には、その値が割り当てられます。

結果の値が複数作成される複数の式では、アクションが失敗します。

構文

parameter "<x>" = "<{expression}>"

x はパラメーターの名前であり、**expression** は値です。パラメーター名と式は、両方とも引用符で囲む必要があります。

例

```
parameter "loc" = "{pathname of folder (value of variable "tmp" of environment)}"
createfile until end
      Operating system = {name of operating system}
      Processor count = {number of processors}
end
delete "{parameter "loc"}¥config.txt"
copy __createfile "{parameter "loc"}¥config.txt"
```

この例では、tmp フォルダーのパス名が含まれる「loc」というパラメーターを定義し、オペレーティング・システムとプロセッサー数が含まれる新規の「名前=値」ファイルを作成し、tmp フォルダーから構成ファイルを削除して新規ファイルで置き換えています。

バージョン 6.0 以降

pause while

このコマンドでは、指定された関連式が true と評価された場合、アクションは次のコマンドへは進みません。値が false と評価された場合、または値を評価できない場合、即座に処理が続行されてアクションの次のコマンドが実行されます。条件を定義するには、関連度置換の構文を使用します。

構文

pause while <true condition>

true condition は、評価する関連式を表します。

例

```
pause while {exists running application "updater.exe"}
pause while {not exists file "C:¥70sp3¥result.log"}
pause while {not exists section "ResponseResult" of file "C:¥70sp3¥result.log"}
```

バージョン 5.1 以降

restart

restart コマンドは、コンピューターを再起動します。オプションの <delay seconds> パラメーターを指定した場合、指定された遅延時間の経過後に自動的にシャットダウンされます。

ユーザーがログインしている場合、指定された遅延時間のカウントダウンを示すダイアログが表示されます。この場合、インターフェースには、「**キャンセル**」ボタンではなく「**すぐに再起動**」ボタンが表示されます。また、クライアント UI が表示されている場合、再起動までの最小遅延時間は 60 秒です。

遅延パラメーターが指定されていない場合、コンピューターを再起動するボタンを押すためのプロンプトが表示されます。

構文

restart [<delay seconds>]

delay seconds は、再起動までの遅延時間を指定するオプション・パラメーターです。

例

```
restart 180
```

この例では、3 分後にコンピューターが再起動されます。

注

遅延再起動は、強制的な再起動です。文書の変更を保存するためのプロンプトなどは表示されません。追加のプロンプトを表示することなく、マシンが再起動されます。

バージョン 5.1 以降

set clock

このコマンドにより、クライアントが登録サーバーに再登録され、クライアントのクロックが、対話中にサーバーから受信した時刻に設定されます。このコマンドは、クライアントのクロックが同期していない場合に便利です。この BES 専用コマンドは、クライアントが評価ライセンスで稼働している場合は使用できません。

構文

set clock

例

```
set clock
```

バージョン 5.1 以降

shutdown

shutdown コマンドは **restart** コマンドに類似していますが、これは単にコンピューターをシャットダウンするだけで、リブートは行いません。

オプションの <delay seconds> パラメーターを指定した場合、指定された遅延時間の経過後に自動的にシャットダウンされます。

ユーザーがログインしている場合、指定された遅延時間のカウントダウンを示す UI が表示されます。この場合、UI には、「**キャンセル**」ボタンではなく「**すぐにシャットダウン**」ボタンが表示されます。

遅延パラメーターが指定されていない場合、コンピューターをシャットダウンするボタンを押すためのプロンプトが表示されます。

構文

shutdown [<delay seconds>]

delay seconds は、シャットダウンまでの遅延時間を指定するオプション・パラメーターです。

例

```
shutdown 180
```

このコマンドは、3 分後にコンピューターをシャットダウンします。

注意

遅延シャットダウンは、強制的なシャットダウンです。文書の変更を保存するためのプロンプトなどは表示されません。追加のプロンプトを表示することなく、マシンがシャットダウンされます。

バージョン 5.1 以降

第 4 章 ファイル・システム・コマンド

action log

action log コマンドでは、アクション・ログを保持する方法を指定できます。通常は、コマンドおよびパラメーターを含む、アクションのすべての側面がログに記録されます。ただし、パラメーターには、秘密鍵の確立やパスワードの暗号化解除に関する情報が含まれることがあります。これらのアクションをプライベートにしておく場合は、action log コマンドは、コマンドのタイプを指定して使用します。これにより、機密性が高い可能性のあるパラメーターは、ログに記録されなくなります。

構文

action log <type>

この **type** は次のいずれかになります。

command

all

例

```
action log all
```

アクションのコマンドとパラメーターの両方がログに記録されます。

```
action log command
```

アクションによって実行されるコマンドのみがログに記録され、パラメーターは記録されません。

バージョン 8.2 以降

add nohash prefetch item

クライアントとリレーは、URL に従って対象ファイルを収集するのではなく、使用する正確なファイルを指定するアクション ID と順序数に従って対象ファイルを収集します。このプリフェッチ・コマンドでは、SHA1 または SHA256 ハッシュ・アルゴリズムは*使用*できません。

このコマンドは、ダウンロード項目をプリフェッチ・キューに追加します。**begin prefetch block** コマンドと **end prefetch block** コマンドの間に指定する必要があります。これは単数コマンドで、一度に 1 つのダウンロードのみ指定することができます。関連度置換はこのコマンドの引数で使用できませんが、IBM Endpoint Manager サーバーは、アクションの作成時にダウンロードをキャッシュすることができます。クライアントが順序数ファイルを要求した場合、リレーによってそれら

のファイルがすべて収集されます。クライアントは、このコマンドが TRUE の条件ブロックにある場合にのみ、対象項目をダウンロードします。

構文

```
add nohash prefetch item [name=<n>] [size=<s>] url=<u>
```

各項目の意味は以下のとおりです。

<n> オプションの名前です。32 文字に制限されています (英数字、ダッシュ、下線、および先頭以外のピリオドを含む)。名前が明示的に指定されていない場合、当該 URL の最後の構成要素 (最後のスラッシュ以降) から名前が取得されます。コマンドごとに指定できるのは、1 つのダウンロード項目のみです。

<s> オプションのファイル・サイズです。これは必須ではありませんが、指定した場合は、有用な進行状況の情報をプログラムで提供することができます。

<u> 必須の URL です。名前が指定されていない場合、指定された URL の最後の構成要素から名前が取得されます。

引数は任意の順序にできますが、認識されないコマンドでは構文エラーが発生します。

例

```
begin prefetch block
      add nohash prefetch item url=http://www.mysite/downloads/download25.exe
end prefetch block
wait {download path "download25.exe"}
```

この例では、プリフェッチ・ブロックで静的ダウンロードを使用し、ハッシュなしで取得しています。これは本質的に安全な手法ではありませんが、IBM Endpoint Manager サーバーのホワイト・リストを使用して URL を検証しています。

バージョン 7.2 以降

add prefetch item

このコマンドは、ダウンロード項目をプリフェッチ・キューに追加します。このコマンドは、**begin prefetch block** コマンドと **end prefetch block** コマンドの間に指定する必要があります。このコマンドでは、複数のダウンロードをセミコロンで区切って指定できます。

関連度置換が適用されない限り、IBM Endpoint Manager サーバーは、アクションの作成時にダウンロードをキャッシュします。IBM Endpoint Manager リレーは、クライアントが要求したファイルだけを収集します。クライアントは、このコマンドが TRUE の条件ブロックにある場合のみ、ファイルを要求します。

コマンド行でダウンロード項目をリストするのではなく、以下のように各項目を 1 つのファイルに集約し (1 行につき 1 項目)、関連度置換を使用することもできます。

```
{concatenation ";" of lines of file <your file>}
```

ダウンロード情報が含まれる Fixlet サイトのファイルを指定する場合、この方法が一般的です。

```
add prefetch item [name=<n>] sha1=<h1> sha256=<h2> size=<s> url=<u> [; ...]
```

各項目の意味は以下のとおりです。

<n> オプションの名前です。英数字、ダッシュ、下線、および先頭以外のピリオド文字に制限されています。名前が明示的に指定されていない場合、当該 URL の最後の構成要素 (最後のスラッシュ以降) から名前が取得されます。

<h1> 指定ファイルの必須 sha1 です。

<h2> 指定ファイルの必須 sha256 です。

<s> 必須のファイル・サイズです。

<u> 必須の URL です。名前が指定されていない場合、指定された URL の最後の構成要素から名前が取得されます。

[; ...] コマンドが複数であることを示します。セミコロンで区切ることで、追加のファイルを指定できます。

このコマンドの引数では関連度置換を使用できますが、置換を使用すると、IBM Endpoint Manager サーバーは、アクションの作成時にダウンロード項目をキャッシュできません。

引数は任意の順序で指定できますが、認識されない引数は無視されます。

関連度置換なしで このコマンドを使用した場合、IBM Endpoint Manager クライアントとリレーは、アクション ID と順序数に従って対象ファイルを収集します。関連度置換ありで このコマンドを使用した場合、クライアントとリレーは、URL と SHA ハッシュ・アルゴリズムに従って対象ファイルを収集します。SHA ハッシュ・アルゴリズムを指定せずにダウンロードを指定するには、**add nohash prefetch item** コマンドを使用します。

例

```
begin prefetch block
      if {name of operating system = "Windows 2000"}
            add prefetch item {"name=up.exe sha1=12 size=45 url=http://ms.com
               /hot2k.exe"}
      else
            add prefetch item {"name=up.exe sha1=12 size=45 url=http://ms.com
               /hot.exe"}
      endif
end prefetch block
wait {download path "up.exe"}
```

この例は、プリフェッチ・ブロックでの条件付きダウンロードを示しています。最初に OS を確認することにより、適切なファイルだけがプリフェッチされるため、時間と帯域幅を大幅に節約できる可能性があります。

バージョン 7.2 以降

appendfile

appendfile コマンドは、サイト・ディレクトリー (デフォルトでは C:¥Program Files¥BigFix¥__Data¥<site name>) に **__appendfile** というテキスト・ファイルを作成します。このコマンドを呼び出すたびに、指定されたテキストがファイルの末尾に追加されます。このコマンドは、診断ファイルを作成したり、エンド・ユーザーのマシンの属性を取り込むファイルを動的に作成したりする場合に便利です。このファイルは、アクション・シェル・コマンドの開始時に自動的に削除されます。

構文

appendfile<text>

text は、ファイル内に挿入される情報を示します。

例

```
appendfile This file will contain details about your computer
appendfile Operating System={name of operating system}
appendfile Windows is installed on the {location of windows folder} drive
```

上記コマンドは、OS と Windows の場所を append ファイルに記録します。

```
appendfile {("Disk " & name of it & ", free space=" & free space of it
  as string) of drives}
```

上記の例は、クライアント PC 上のすべてのドライブの名前と使用可能な空き容量を記録します。

注

appendfile コマンドは、後でスクリプト・インタープリターに渡されるスクリプトを作成するアクションの一部として使用します。例えば、以下の構文でアクション・コマンドを使用して、.ini ファイルを作成できます。

```
appendfile [HKR]
appendfile HostBasedModemData¥Parameters¥Driver,ModemOn,1,00,00
delete {location of system folder}¥smcfg.ini
copy __appendfile {location of system folder}¥smcfg.ini
run smcfg
```

これと同じ方法で、.bat ファイル、.cmd ファイル、Visual Basic スクリプト、bash シェル・スクリプトなどを作成することもできます。

バージョン 5.1 以降

archive now

このコマンドは、Archive Manager を起動します。アーカイブの動作モードが手動モードに設定されている場合、このコマンドは、構成されている一連のファイルのアーカイブとアップロードをトリガーします。適切なアーカイブ・モードを手動モードに設定するには、以下の設定を使用します。

_BESClient_ArchiveManager_OperatingMode = 2

archive now コマンドは、動作モードが手動モードに設定されていない場合に Failed ステータスを返します。既存のアーカイブが現在アップロード中の場合も、Failed ステータスを返します。

構文

archive now

例

```
archive now
```

このコマンドは、構成されている一連のファイルのアーカイブとアップロードを開始します。

バージョン 5.1 以降

begin prefetch block

このコマンドは、一連のコマンドを開始してファイルをダウンロードします。通常、IBM Endpoint Manager アクションを使用してファイルをダウンロードする場合、確実性を保証するためにチェックサムが評価されます。ただし、ダウンロードのターゲットが流動的である場合 (アンチウィルス定義など)、この要件は厳しすぎる可能性があります。そうした場合に対処するため、IBM Endpoint Manager には動的ダウンロード・コマンドが用意されています。このコマンドは、**begin prefetch block** と **end prefetch block** で囲みます。詳細については、*『プリフェッチ・ブロックについて』*と*『動的ダウンロードについて』*の説明を参照してください。

この機能は IBM Endpoint Manager リレー構造と密接に統合されており、ダウンロード速度と帯域幅を最適化することができます。アクションでファイルが要求されると、リレーはそのキャッシュを確認し、使用可能な場合は対象ファイルを即座に転送します。使用可能でない場合、要求は、IBM Endpoint Manager サーバーに達するまで行をスキップします。

動的ダウンロードは、ホワイト・リストを使用して、信頼済みサイトだけにアクセスするようにします。ホワイト・リストには、以下のファイルが含まれます。

Windows システム:

```
<BES Server Install Path>¥Mirror Server¥Config¥DownloadWhitelist.txt
```

Linux システム:

```
<BES Server Install Path>/Mirror Server/config/DownloadWhitelist.txt
```

このファイルには、正規表現として形式設定された URL のリストが含まれます (`http://.*¥.mysite¥.com/.*` など)。プリフェッチ・ステートメントで指定する URL をダウンロードするには、この URL がホワイト・リスト内の項目と一致している必要があります。この URL がホワイト・リスト内に存在しない場合、コマンドは **NotAvailable** エラーで失敗します。

以下の既存のコマンドは、プリフェッチ・ブロック内で使用できます。

```
// コメント行およびブランク行
if/elseif/else/endif
parameter
action parameter query (お客様によりコメント扱い)
```

以下の新規コマンドは、プリフェッチ・ブロック内では使用できますが、ブロック外では使用できません。

```
add prefetch item
add nohash prefetch item
collect prefetch items
execute prefetch plug-in
```

構文

begin prefetch block

アクション・スクリプトでは 1 つのプリフェッチ・コマンド・ブロックのみ使用でき、**end prefetch block** コマンドで終了する必要があります。

このコマンドより前に記述できるのは、コメントとブランク行だけです。プリフェッチ・ブロックを使用してアクションを処理する場合、 **download**、**download as** と **prefetch** は、アクション・スクリプトのどの場所でも使用できません。**download now as** コマンドは使用できますが、*プリフェッチ・ブロック内とプリフェッチ・ブロックより前では使用できません*。

例

```
// action script to automatically update a URL manifest from a custom site
begin prefetch block
    parameter "ini"="{file "server_bf.ini" of site (value of setting
        "MyCustomSite") of client}"
    // prefetch the plug-in that provides the download list
    add prefetch item name=plugin.exe sha1=123 sha256=347 size=12
       url=http://www.mysite/downloads/myplugin.exe
    // collect above prefetch file (needed to create a manifest composed of URLs)
    collect prefetch items
    // execute the plug-in that produces a manifest from the ini data file
    execute prefetch plug-in "{download path "plugin.exe"}" /downloads
        "{parameter "ini"}"
        "{download path "urllist"}"
    // URL manifest formatted as lines containing: name=<n> sha256=<h2> size=<s>
        url=<url>
    add prefetch item {concatenation " ; " of lines of download file "urllist"}
end prefetch block
// action is now active, update the files:
waithidden "{download path "plugin.exe"}" /update "{parameter "ini"}" "{location of
    download folder}"
```

この例では、別のファイルを処理するプラグインをダウンロードし、ダウンロードする追加ファイルのリストを含むマニフェストを作成しています。プリフェッチ・ブロックが終了すると、ファイルはダウンロードされてダウンロード・フォルダーに移動されるため、残りのアクションを続行できます。

注:

古いコンソールとクライアントでは、新しいプリフェッチ機能を使用したアクション・スクリプトは拒否され、構文エラーが存在するスクリプトとして認識されま

す。古いリレーでは、サーバーとクライアントで処理できる場合でも、動的ダウンロード・アクションは処理されません。アクション・スクリプトで使用できるプリフェッチ・ブロックは 1 つだけです。

特定のコマンド (**download**、**download as** および **prefetch**) は、プリフェッチ・ブロックが含まれるアクション・スクリプトのどの場所でも使用できません。**download now** コマンドはスクリプト内で使用できますが、プリフェッチ・ブロックより後 に記述する必要があります。

いくつかの新規インスペクターが追加されています。これらのインスペクターで関連度置換を使用すると、アクション・スクリプトでダウンロード・ファイルを参照できます。これらのインスペクターには、**download path "<name>"**、**download file "<name>"**、**download folder** などがあります。詳細については、インスペクターのガイドを参照してください。

バージョン 7.2 以降

collect prefetch items

このコマンドは、**add nohash prefetch items** や **add prefetch items** などのコマンドによって項目がプリフェッチ・キューに追加された後で、IBM Endpoint Manager リレーからそれらの項目を収集します。アクションのプリフェッチ処理は、指定されたすべてのファイルが収集されるまで中断されます。**add prefetch** コマンドで新しい名前のダウンロードが指定された場合、この時点で名前が変更されます。

通常、このコマンドは、プラグインを取得する場合や、プラグインで処理できる一連のファイルを取得する場合に使用します。この場合、最初にファイルがプリフェッチ・リストに追加されて収集され、次に後続の **execute prefetch plug-in** コマンドによって処理されます。これにより、追加のダウンロードが含まれるファイルを作成できます。各 **collect prefetch items** コマンドは、同期点として処理されます。これにより、アクションのプリフェッチ処理は、ファイルがダウンロードされるまで待機してから、処理を続行します。ファイルが使用可能になると、アクションは最初から再処理されます。そのため、マシン状態の変更によって更新された可能性があるすべてのファイルについて、アクションで補正処理を行うことができます。アクション内の次のコマンドは、**collect prefetch items** コマンドが実行され、プリフェッチ・リストのすべてのファイルがダウンロードされた後で、はじめて処理されます。

end prefetch block コマンドは、自動収集を実行します。これにより、後続のアクション・コマンドで必要なファイルが用意されます。

収集されたこれらのファイルは、各種の新規インスペクターを使用して検証できます。プリフェッチ処理においてアクションがアクティブであるかどうかに応じて、各インスペクターでの解釈が異なる場合があります。これらのインスペクターの詳細については、**『動的ダウンロードについて』**を参照してください。

構文

```
collect prefetch items
```

例

```
begin prefetch block
      parameter "ini"="{file "server_bf.ini" of site (value of setting
       "MyCustomSite") of client}"
      add prefetch item name=myPlugIn.exe sha1=12 size=12 sha256=347 size=456
        url=http://mysite/plugin.exe
      // collect the plug-in before continuing:
      collect prefetch items
      execute prefetch plug-in "{download path "myPlugIn.exe"}" /downloads
        "{parameter "ini"}" "{download path "urllist"}"
      add prefetch item {concatenation " ; " of lines of download file "urllist"}
end prefetch block
```

この例では、collect ステートメントにより、残りのプリフェッチ・ブロックに進む前に、プラグインが正常にダウンロードされていることが確認されます。

バージョン 7.2 以降

copy

このコマンドは、ソース・ファイルを、指定された宛先ファイルにコピーします。宛先が既に存在する場合、または何らかの原因によってコピーが失敗した場合 (宛先ファイルがビジー状態である場合など)、copy コマンドが記述されたアクション・スクリプトは終了します。

構文

copy<Source_FileName> <Destination_FileName>

Source_Filename および **Destination_Filename** は、それぞれコピー元およびコピー先となるファイルの名前です (通常は引用符で囲みます)。

例

```
copy "{name of drive of windows folder}¥win.com" "{name of drive of
   windows folder}¥bigsoftware¥win.com"
```

このコマンドは、win.com ファイルを bigsoftware フォルダーにコピーします。

```
delete "c:¥windows¥system¥windir.dll"
copy " __Download¥windir.dll" "c:¥windows¥system¥windir.dll"
```

このアクション・シェル・コマンドのペアは、コピー・アクションを実行する前にターゲット・ファイルを削除します (存在する場合)。

バージョン 5.1 以降

createfile until

このコマンドは、サイト・ディレクトリーに **__createfile** というテキスト・ファイルを作成します。これにより、終了ストリングまで、ファイル内に一連のステートメントを挿入できます。コマンドの形式は、以下のとおりです。

```
createfile until <end-delim-string>
      line 1
      line 2
      ...
end-delim-string
```

注: 「line 1」、「line 2」などのラベルが付いている行に、誤って end-delim-string が含まれていないことを確認してください。これが含まれていると、アクション・パーサーは、end-delim-string の最初のインスタンスより後のアクション・コマンドを検索します。

構文

createfile until <delimiter>

statements...

delimiter

例

```
parameter "config" = "{pathname of folder (value of variable "tmp" of
     environment)}¥config.txt"
createfile until end
      Operating system = {name of operating system}
      Processor count = {number of processors}
end
delete "{parameter "config"}"
copy __createfile "{parameter "config"}"
```

この例では、tmp フォルダーの構成ファイルのパス名が含まれる「config」というパラメーターを定義し、オペレーティング・システムとプロセッサー数が含まれる新規の「名前=値」ファイルを作成し、tmp フォルダーから構成ファイルを削除して新規ファイルで置き換えています。

バージョン 6.0 以降 -- Windows のみ

delete

このコマンドは、指定されたファイルを削除します。ファイルは存在するが削除できない場合、delete コマンドが記述されたアクション・スクリプトは終了します。これは、書き込み保護されている場合や、CD-ROM から削除しようとした場合などに発生する可能性があります。ただし、ファイルがまったく存在しない場合、アクション・スクリプトは引き続き実行されます。

構文

delete<FileName>

FileName は、削除するファイルの名前です (通常は引用符で囲みます)。関連度置換は、delete アクション・コマンド行の引数で実行されます。

例

```
delete "c:¥program files¥bigsoftware¥module.dll"
delete "{name of drive of windows folder}¥win.com"
```

これらの例では、指定されたファイルを削除しています。変数置換 (中括弧で囲む) を使用して、パス名を指定できます。

注

ファイル名にスペースが含まれている場合は、ファイル名を引用符で囲んでください。引用符で囲まれていない場合、ファイル・システムは、パスまたはファイル名にスペースが含まれているファイルにアクセスできません。

バージョン 5.1 以降

download

Deprecated: **download as** または **download now as** を使用します。

このコマンドは、URL で指定されたファイルをダウンロードします。これは、バージョン 2.0 クライアント・エディションとの後方互換性のために用意されているコマンドで、引き続きサポートされます。このコマンドにより、従来の IBM Endpoint Manager アクションを適切に処理できます。その他のすべてのアプリケーションでは、このコマンドは、**download as** コマンドと **download now as** コマンドで置き換えられています。

ダウンロードされたファイルは、download コマンドを発行した Fixlet サイトのローカル・フォルダーに関連する「__Download」というフォルダーに保存されます (フォルダー名の先頭は 2 つの下線)。

ダウンロードが失敗した場合、アクション・スクリプトは終了します。ファイル名は、URL の最後のスラッシュ以降の部分から取得されます。

例えば、次のコマンドを考えてみます。

```
download ftp://ftp.microsoft.com/deskapps/readme.txt
```

上記のアクション例では、Microsoft サイトから readme.txt ファイルをダウンロードし、readme.txt というファイル名でローカルの __Download フォルダーに自動保存しています。

ファイル名は URL から取得されます。最後の / 文字または ¥ 文字以降のすべての部分が、ファイル名として使用されます。これにより、問題のあるファイル名が生成される場合があります。以下に例を示します。

```
URL: http://skdkdk.ddddd.com/cgi-bin/xyz?jjj=yyy
```

この例では、xyz?jjj=yyy という無効な名前でファイルが作成されます。通常、URL の末尾に仮引数を追加することにより、この問題を回避することができます。

```
http://skdkdk.ddddd.com/cgi-bin/xyz?jjj=yyy?file=/ddd.txt
```

この例では、ddd.txt という名前でファイルが作成され、__Download ディレクトリーに保存されます。**download as** コマンドと **prefetch** コマンドを使用しても、この状況に対処することができます。

構文

```
download [option] <File_URL>
```

[options] には、以下の 2 つのオプション・キーワードのいずれかを指定することができます。

open: ShellExecute API を呼び出し、ダウンロードの完了後に、生成されたファイル名を渡します。

now: アクションの開始前にプリフェッチする処理とは対照的に、IBM Endpoint Manager クライアントに対して、アクション実行のその時点でダウンロードを開始するように指示します。エージェントは、リレー・システムを経由するのではなく、指定された URL からダウンロードを直接収集します。

File_URL は、ダウンロードするファイルの場所です。

例

```
download http://download.mycompany.com/update/bfxxxx.exe
```

この例では、mycompany サイトから bfxxxx.exe ファイルをプリフェッチし、ダウンロードしたファイルをデフォルト・サイトの「__Download」フォルダーに送信します。

```
download open http://download.bigfix.com/update/bfxxxx.exe
```

この例では、ダウンロードの完了後、デフォルト・サイトの「__Download」フォルダーに bfxxxx.exe ファイルをプリフェッチして保存し、プログラムを実行します。

```
download now http://download.mycompany.com/update/bfxxxx.exe
```

この例では、コマンドが実行されると同時に、mycompany サイトから bfxxxx.exe ファイルをダウンロードします。

```
download "http://download.microsoft.com/download/prog.exe"
run "__Download¥prog.exe"
```

このアクション・セットは、ダウンロード処理を自動化します。これにより、実行可能パッチをシングルクリックで適用することができます。ダウンロードされたプログラムは、download コマンドによって Fixlet サイトに格納され、このサイトの「__Download」ディレクトリーから実行されます。Fixlet サイト・ディレクトリーはすべてのコマンドの作業ディレクトリーであり、__Download ディレクトリーはこのディレクトリーに存在しています。

注: 関連度置換は、**download** アクション・コマンド行自体に対しては実行**されません**。その理由は、これらのアクションは、ダウンロードを送信する他のコンポーネントによってスキャンされますが、それらのコンポーネントは当該クライアントの評価コンテキストを共有しないさまざまなマシンで実行されるためです。ただし、この制限があることにより、IBM Endpoint Manager は、リレー階層を使用してダウンロードをクライアントにプリフェッチすることができます。

バージョン 5.1 以降

download as

このコマンドは、URL で指定されたファイルをダウンロードします。このファイルの名前を変更することもできます。ダウンロードされたファイルは、**download as** コマンドを発行した Fixlet サイトのローカル・フォルダーに関連する「__Download」というフォルダーに保存されます (フォルダー名の先頭は 2 つの下線)。

例えば、次のコマンドを考えてみます。

```
download as intro.txt ftp://ftp.microsoft.com/deskapps/readme.txt
```

上記のアクション例では、Microsoft サイトから readme.txt ファイルをダウンロードし、intro.txt というファイル名でローカルの __Download フォルダーに自動保存しています。ダウンロードが失敗した場合、アクション・スクリプトは終了します。

このコマンドを、sha1 または sha256 の値が設定された **continue if** とともに発行すると、ファイルをプリフェッチすることができます。

構文

```
download as <name> <url>
```

name は、特殊文字やパス区切り文字などが含まれていない単純なファイル名です。ファイル名が以下のいずれかのルールに違反している場合、download コマンドは失敗します。

名前は 32 文字以内で指定する必要があります。

名前は、ASCII 文字 (a から z、A から Z、0 から 9、-、_、および先頭以外のピリオド) だけで構成する必要があります。

url は、指定されたファイルの完全 URL です。

例

```
download as myprog.exe http://www.website.com/update/prog555.exe
```

この例では、Web サイト上の指定フォルダーから prog555.exe ファイルをダウンロードしてアクション・サイトである「__Download」フォルダーに送信し、ファイル名を myprog.exe に変更しています。

```
download as patch1 http://www.download.windowsupdate.com/msdownload/update/v3-
     19990518/cabpool/q307869_f323efa52f460ea1e5f4201b011c071ea5b95110.exe
continue if {(size of it = 813160 and sha1 of it =
      "92c643875dda80022b3ce3f1ad580f62704b754f") of file "patch1" of
      folder "__Download"}
```

この例では、指定されたファイルをダウンロードしてファイル名を patch1 に変更してから、size と sha1 が正しい場合のみ、処理を続行します。

注:

関連度置換は、**download as** アクション・コマンド行自体に対しては実行**されません**。その理由は、これらのアクションは、ダウンロードを送信する他のコンポーネントによってスキャンされますが、それらのコンポーネントは当該クライアントの評価コンテキストを共有しないさまざまなマシンで実行されるためです。ただし、この制限があることにより、IBM Endpoint Manager は、リレー階層を使用してダウンロードをクライアントにプリフェッチすることができます。

バージョン 6.0 以降 -- Windows のみ

download now as

このコマンドは、URL で指定されたファイルをダウンロードします。このファイルの名前を変更することもできます。ダウンロードされたファイルは、**download now as** コマンドを発行した Fixlet サイトのローカル・フォルダーに関連する __Download というフォルダーに保存されます (フォルダー名の先頭は 2 つの下線)。

ダウンロードが失敗した場合、アクション・スクリプトは終了します。

例えば、次のコマンドを考えてみます。

```
download now as intro.txt ftp://ftp.microsoft.com/deskapps/readme.txt
```

上記のアクション例では、Microsoft サイトから readme.txt ファイルを即座にダウンロードし、intro.txt というファイル名でローカルの __Download フォルダーに自動保存しています。

構文

```
download now as <name> <url>
```

name は、特殊文字やパス区切り文字などが含まれていない単純なファイル名です。ファイル名が以下のいずれかのルールに違反している場合、download コマンドは失敗します。

名前は 32 文字以内で指定する必要があります。

名前は、ASCII 文字 (a から z、A から Z、0 から 9、-、_、および先頭以外のピリオド) だけで構成する必要があります。

url は、指定されたファイルの完全 URL です。

例

```
download now as myprog.exe http://www.website.com/update/prog555.exe
```

この例では、Web サイト上の指定フォルダーから prog555.exe ファイルを即座にダウンロードしてアクション・サイトである「__Download」フォルダーに送信し、myprog.exe というファイル名を付けています。

```
download now as patch2 http://www.download.windowsupdate.com/msdownload/update
 /v3-19990518/cabpool/q310507_2f3c5854999b7c58272a661d30743abca15caf5c.exe
continue if {(size of it = 845416 and sha1 of it =
 "c964d4fd345b6e5fd73c2235ec75079b34e9b3d2") of file "patch2.exe" of folder
 "__Download"}
```

この例では、指定されたファイルを Web サイトから即座にダウンロードしてアクション・サイトである __Download フォルダーに送信し、patch2 というファイル名を付けています。size と sha1 が正しい場合のみ、アクションを続行します。

注: 関連度置換は、**download now as** アクション・コマンド行自体に対しては実行**されません**。その理由は、これらのアクションは、ダウンロードを送信する他のコンポーネントによってスキャンされますが、それらのコンポーネントは当該クライアントの評価コンテキストを共有しないさまざまなマシンで実行されるためです。

ただし、この制限があることにより、IBM Endpoint Manager は、リレー階層を使用してダウンロードをクライアントにプリフェッチすることができます。

バージョン 6.0 以降 -- Windows のみ

end prefetch block

このコマンドは、プリフェッチ・ブロックの末尾を示します (**begin prefetch block** を参照)。**begin prefetch block** コマンドを指定した場合は、必ずこのコマンドが必要になります。このコマンドにより、**collect prefetch items** コマンドが自動的に実行されます。つまり、プリフェッチ・リストに追加されたすべてのファイルが、ブロックの終了時に使用可能になるということです。

構文

```
end prefetch block
```

アクション・スクリプトでは 1 つのプリフェッチ・コマンド・ブロックのみ使用でき、**begin prefetch block** コマンドと **end prefetch block** コマンドの間に記述する必要があります。

プリフェッチ・ブロックより前に記述できるのは、コメントとブランク行だけです。プリフェッチ・ブロックを使用してアクションを処理する場合、**download as** と **prefetch** は、アクション・スクリプトのどの場所でも使用できません。プリフェッチ・ブロックの後に **download now** コマンドを記述することはできますが、*プリフェッチ・ブロックより前とプリフェッチ・ブロック内に記述することはできません。*

```
begin prefetch block
 add prefetch item sha1=123 sha256=689 size=456 url=http://ms.com/downloads/hotfix123.exe
end prefetch block
wait {download path "hotfix123.exe"}
```

このコードは、プリフェッチ・ブロックでの静的ダウンロードを示しています。動的な関連度置換は使用していませんが、バージョン 7.2 以降では、これが推奨されるダウンロード形式です。**end prefetch block** コマンドでもファイル (hotfix123.exe) が収集されるため、後続の **wait** コマンド (実行後に完了を待機) においては、このファイルが必ず使用可能な状態になっています。

注: 古いコンソールとクライアントでは、新しいプリフェッチ機能を使用したアクション・スクリプトは拒否され、構文エラーが存在するスクリプトとして認識されます。古いリレーでは、サーバーとクライアントで処理できる場合でも、動的ダウンロード・アクションは処理されません。

バージョン 7.2 以降

execute prefetch plug-in

このコマンドは、指定されたコマンドに引数を渡して、そのコマンドを実行します。このコマンドは、長時間の実行可能ファイル用には設計されていないため、完了までにクライアントが待機する時間は 60 秒だけです。このコマンドを使用して、ダウンロードを認証または実行できます。また、このコマンドを使用して、後続の **add prefetch item** コマンドで検査可能な値を作成できるカスタム・ロジックを実行することもできます。

アンチウィルス定義を更新する場合など、このコマンドを使用して、ファイルを処理するコードを実行すると、ダウンロードする一連の URL が含まれる別のファイルを作成することができます。

構文

```
execute prefetch plug-in "executable pathname" <args>
```

各項目の意味は以下のとおりです。

"executable pathname"
実行するプラグインの絶対パス名です。このコマンドは、実行時間が短く、迅速に結果が返される実行可能ファイル用に設計されています。IBM Endpoint Manager クライアントは、コマンドの完了まで 60 秒間待機します。この待機期間を中断できるのは、シャットダウン要求だけです。60 秒が経過すると、クライアントはメッセージを記録し、このコマンドを無効にします。コマンドが無効になると、このコマンドを使用するすべてのアクションは、クライアントが再起動されるまで実行されません。通常、このコマンドは、60 秒よりも大幅に短い時間で完了するものと想定されているため、実行に 2 秒以上かかっている場合、クライアントは該当するメッセージを記録します。関連度置換を使用して、パス名を指定できます。

<args> 実行可能ファイルに渡される引数です。

execute prefetch プラグイン・アプリケーションの終了コードは、クライアントに対して失敗か成功かを通知するものであるため、重要なコードです。0 (ゼロ) は成功を表し、それ以外のすべての終了コードは失敗を表し、失敗したアクション試行として処理されます。デバッグ目的で使用できるように、この終了コードはクライアント・ログに記録されます。

例

```
begin prefetch block
      parameter "ini_file"={file "server_bf.ini" of site (value of setting
       "MyCustomSite") of client}
      add prefetch item name=plugin.exe sha1=123 sha256=789 size=12
       url=http://mysite/myplugin.exe
      collect prefetch items
      // execute the plug-in to produce a manifest from the ini_file:
      execute prefetch plug-in "{download path "plugin.exe"}" /downloads
       "{parameter "ini_file"}" "{download path "manifest"}"
add prefetch item {concatenation " ; " of lines of download file "manifest"}
end prefetch block
```

この例では、マニフェストを作成する ini_file を処理するプラグインをダウンロードしています。

バージョン 7.2 以降

extract

このコマンドは、ダウンロード・フォルダー (__Download) 内の指定のアーカイブからファイルを抽出し、結果を同じフォルダーに格納します。

アーカイブ・ファイルは、圧縮 tar ファイルに類似しています。IBM Endpoint Manager は、Archivewriter というツールを使用して、アーカイブを構成します。これは、ディレクトリー全体をコンピューターにコピーする場合に便利です。多くの場合、こうしたコピーは、セットアップ実行可能ファイルとともに、複数のファイルが含まれるインストーラーで必要になります。IBM Endpoint Manager コンソールには、この種のアーカイブを使用するディレクトリーを容易に配布できるウィザードが用意されています。

バージョン 8.2 では、このコマンドにより、オプションの 2 つ目の引数を使用してターゲット・ディレクトリーを指定できます。

構文

extract <Archive File> [<Destination_Directory>]

この宛先ディレクトリーはオプションであり、デフォルトでは __download ディレクトリーに設定されます。

例

```
extract InstallMyApp.zip
```

この例では、__Download フォルダー内の構成ファイル InstallMyApp を抽出し、結果を __Download フォルダーに戻してから、元の InstallMyApp ファイルを削除しています。

```
extract InstallMyApp.zip "d:/temp"
```

これは前述の例と同じですが、ターゲット・ディレクトリーを指定している点が異なります。

注: ファイル名に空白が含まれている場合でも、ファイル名を引用符で囲まないでください。これは、引用符を有効なファイル名の文字として使用できるシステムと整合性をとるためです。

バージョン 5.1 以降

folder create

このコマンドは、指定されたフォルダーを作成します。フォルダーを作成できない場合、create コマンドが記述されたアクション・スクリプトは終了します。これは、書き込み保護されている場合や、CD-ROM に書き込もうとした場合などに発生する可能性があります。また、パスが既に存在するが、そのパスがフォルダーを参照していない場合にも、スクリプトが終了します。

構文

folder create<FolderName>

FolderName は、作成するフォルダーの名前です (通常は引用符で囲みます)。関連度置換は、folder create コマンド行の引数で実行されます。

例

```
folder create "c:¥program files¥bigsoftdir"
folder create "{name of drive of windows folder}¥Extras"
```

これらの例では、指定されたフォルダーを作成しています。変数置換 (中括弧で囲む) を使用して、パス名を指定できます。

注

フォルダー名のスペースがそのまま保持されるよう、フォルダー名は常に引用符で囲むことをお勧めします。

バージョン 8.0

folder delete

このコマンドは、指定されたフォルダーを削除します。これは再帰的なコマンドで、フォルダー内のすべてのファイルとフォルダーが削除されます。フォルダーは存在するが削除できない場合、delete コマンドが記述されたアクション・スクリプトは終了します。これは、書き込み保護されている場合や、CD-ROM から削除しようとした場合などに発生する可能性があります。このアクションは、フォルダー内容がビジー状態である場合にも失敗します (フォルダー内容は予測不能な状態のままになります)。このような状況は、フォルダー内のファイルが、別のアプリケーションによって使用されている場合などに発生する可能性があります。

ただし、フォルダーがまったく存在しない場合、アクション・スクリプトは引き続き実行されます。

構文

folder delete <FolderName>

FolderName は、削除するフォルダーの名前です (フォルダー名のスペースを保持するために、通常は引用符で囲みます)。関連度置換は、delete folder コマンド行の引数で実行されます。

例

```
folder delete "c:¥program files¥bigsoftdir"
folder delete "{name of drive of windows folder}¥Temp"
```

これらの例では、指定されたフォルダーを削除しています。変数置換 (中括弧で囲む) を使用して、パス名を指定できます。

重要: このコマンドは再帰的であり、クライアント上のすべてのファイルを削除できます。

バージョン 8.0

move

このコマンドは、ソース・ファイルを、指定された宛先ファイルに移動します。アクション作成者は、このコマンドを使用してファイル名を変更することもできます。宛先が既に存在する場合、ソース・ファイルが存在しない場合、または何らかの原因によって移動が失敗した場合、move コマンドが記述されたアクション・スクリプトは終了します。

構文

move<Source_FileName> <Destination_FileName>

Source_Filename と **Destination_Filename** は、それぞれ移動元ファイルと移動先ファイルの名前です (通常は引用符で囲みます)。

例

```
move "c:¥program files¥bigsoftware¥module.dll" "c:¥temp¥mod.dll"
```

このコマンドは、mod.dll ファイルを移動してファイル名を変更します。ファイル名とフォルダー名にスペースが含まれている場合は、引用符で囲む必要があります。

```
delete "c:¥updates¥q312456.exe"
move "__Download¥q312456.exe" "c:¥updates¥q312456.exe"
```

上記のコマンド行は、最初にファイルを削除し、次に、別の場所にある同じファイルを、削除したファイルが存在していた場所に移動しています。

バージョン 5.1 以降

prefetch

prefetch コマンドにより、アクションの開始前にファイルをダウンロードできます。事前にファイルをダウンロードして検査するための、対応する **continue if** ステートメントは必要ありません。**download** コマンドよりも、prefetch コマンドのほうが優先されます。

例えば、次のコマンドを考えてみます。

```
prefetch a.exe sha1:01234567890123456789012345678901234567890
   sha256:0a1b2345678901234567g8901j234e5678g90y12r3456789345678923167e3se
   size:11723 http://x/z.exe
```

上記のアクション例では、指定サイトから z.exe ファイルをプリフェッチし、a.exe というファイル名でローカルの __Download フォルダーに自動保存しています。

構文

```
prefetch <name> sha1:<value> size:<value> sha256:<value> <url>
```

name は、特殊文字やパス区切り文字などが含まれていない単純なファイル名です。ファイル名が以下のいずれかのルールに違反している場合、prefetch コマンドは失敗します。

名前は 32 文字以内で指定する必要があります。

名前は、ASCII 文字 (a から z、A から Z、0 から 9、-、_、および先頭以外のピリオド) だけで構成する必要があります。

sha1:value または **sha256:value** はセキュア・ハッシュ・アルゴリズムの値、**size:value** はファイルのサイズ (バイト単位)、**url** はサイトの場所 (ファイル名を含む) をそれぞれ表しています。

例

```
prefetch patch3 sha1:92c643875dda80022b3ce3f1ad580f62704b754f
  size:813160 http://www.download.windowsupdate.com/msdownload
  /update/v3-19990518/cabpool/
   q307869_f323efa52f460ea1e5f4201b011c071ea5b95110.exe
```

このコード行は、Web サイト上の指定フォルダーから特定のファイルをプリフェッチしてアクション・サイトである「__Download」フォルダーに送信し、ファイル名を patch3 に変更します。

```
if {name of operating system = "WinXP"}
   prefetch patch.exe sha1:92c643875dda80022b3ce3f1ad580f62704b754f
    size:813160 http://www.download.windowsupdate.com/msdownload
     /update/v3-19990518/cabpool/
      q307869_f323efa52f460ea1e5f4201b011c071ea5b95110.exe

else
   prefetch patch.exe sha1:c964d4fd345b6e5fd73c2235ec75079b34e9b3d2
    size:845416 http://www.download.windowsupdate.com/msdownload
     /update/ v3-19990518/cabpool/
      q310507_2f3c5854999b7c58272a661d30743abca15caf5c.exe

endif
utility __Download¥patch.exe
wait __Download¥patch.exe
```

このコードは、オペレーティング・システムに基づいてファイルをプリフェッチして patch.exe という名前でユーティリティー・キャッシュに保存し、処理が完了するまで待機してからアクションを続行します。

注: 関連度置換は、**prefetch** アクション・コマンド行自体に対しては実行**されません**。その理由は、これらのアクションは、ダウンロードを送信する他のコンポーネントによってスキャンされますが、それらのコンポーネントは当該クライアントの評価コンテキストを共有しないさまざまなマシンで実行されるためです。ただし、この制限があることにより、IBM Endpoint Manager は、リレー階層を使用してダウンロードをクライアントにプリフェッチすることができます。

バージョン 6.0 以降 -- Windows のみ

relay select

relay select コマンドは、IBM Endpoint Manager クライアントに対して、最も近いリレーを選択するように強制します (そのリレーが使用可能な場合)。このコマンドは、次の機会にリレー選択を行うように、クライアントに対して要求します。このコマンドは、保留中のリレー選択の成功や失敗にかかわらず、必ず即座に成功します。

構文

relay select

例

```
relay select
```

このコマンドは、IBM Endpoint Manager クライアントに対して、最も近いリレーを検索して接続するように指示します。

バージョン 5.1 以降

utility

utility コマンドを使用すると、共通して使用されるプログラムを特殊なキャッシュに入れることができます。以下に例を示します。

```
utility __Download/RunQuiet.exe
```

この例では、共通の **RunQuiet** プログラムをユーティリティー・キャッシュに入れ、このプログラムが複数回ダウンロードされないようにしています。

6.0 クライアントには 2 つのディスク・キャッシュがあります。1 つはユーティリティー・プログラム用、もう 1 つはアクション・ペイロード用です。アクション・ペイロード・キャッシュに入れられたファイルが、ユーティリティー・キャッシュからファイルをプッシュすることはありません (その逆も同様)。

6.0 クライアントは、アクション・ダウンロードの sha1 または sha256 の値を使用して、クライアント上に既に存在するすべての一致するユーティリティー (「RunQuiet」など) を特定します。

アクション固有のフォルダーが作成され、プリフェッチされたダウンロードが格納されます。sha1 または sha256 の値に一致する既存のファイルは、再度ダウンロードする必要はありません。その他のファイルは、すべて親リレーからプリフェッチされます。すべてのダウンロードがクライアント上で使用可能になると、アクション固有のフォルダーからアクション・サイトの __Download フォルダーにファイルが移動され、アクションが開始されます (これは 6.0 以前のクライアント動作からの変更点です)。

アクションが完了すると、__Download フォルダー内に残っている sha1 または sha256 でプリフェッチされたすべてのファイルが、ユーティリティー・キャッシュ

の対象となります。これらのファイルでは、それぞれの sha1 または sha256 の値が再計算されます。sha1 または sha256 の値に一致するファイルは、すべてユーティリティー・キャッシュに移動できます。

LRU (Least Recently Used) 方式を使用して、キャッシュが制限サイズ内に維持されます。短い間隔の場合に限り、キャッシュ制限を 1 ファイル分だけ超えることがあります。

構文

utility <pathname>

例

```
prefetch patch.exe sha1:92c643875dda80022b3ce3f1ad580f62704b754f
__size:813160 http://www.download.windowsupdate.com/msdownload/update/v3-
__19990518/cabpool/q307869_f323efa52f460ea1e5f4201b011c071ea5b95110.exe
utility __Download¥patch.exe
wait __Download¥patch.exe
```

この例では、ファイルをプリフェッチして patch.exe という名前でユーティリティー・キャッシュに保存し、処理が完了するまで待機してからアクションを続行しています。

バージョン 6.0 以降 -- Windows のみ

第 5 章 設定コマンド

setting

設定とは、個々の Fixlet サイトまたはクライアント・コンピューターに適用できる指定値のことです。各設定には、関連付けられている時間があります。このタイム・スタンプを使用して、優先順位を設定します (最新の設定が、それ以前のすべての設定よりも優先されます)。

IBM Endpoint Manager コンソールのオペレーターは、設定を作成して伝搬することができます。コンソールによって発行された設定は、現在の日時でタグ付けされます。設定は、複数のグループに分けられています (各サイト用、クライアント用など)。設定の各グループはそれぞれ独立しており、クライアント上で永続的に存在します。

設定は、Fixlet のアクションでも作成できます。この場合、通常は置換 {now} を使用します。この置換は、アクションの実行時に評価されます。「effective date of <setting>」などのインスペクターを使用して、これらの設定とそのタイム・スタンプを検証できます (詳細については、「Inspector Guides」を参照してください)。

構文

setting "<name>"="value" on "<date>" for client

setting "<name>"="value" on "<date>" for site "<sitename>"

name=value は設定を表し、**date** は、優先順位を設定するためのタイム・スタンプです。これらは、**クライアント・**コンピューターまたは指定された**サイト**に対して設定できます。

例

```
setting "name"="Bob" on "31 Jan 2007 21:09:36 gmt" for client
```

この例では、この設定の作成時にコンソールで指定された MIME 日時スタンプを持つクライアント・マシンで、name 変数を Bob に設定しています。この設定は、これより前の日付を持つ他のすべての名前設定よりも優先されます。

```
setting "preference"="red" on "{now}" for site "color_site"
```

この例では、このコマンドが含まれるアクションの実行時に、{now} が MIME 日時として評価され、ストリングに置換されます。このコマンドは、指定された Fixlet サイトについて、「preference」変数を「red」に設定します。同じ名前のサイトが複数存在しない限り、完全な収集 URL を使用することなくサイトを指定できます。各サイトで、それぞれ異なる「preference」設定を持つことができます。

```
setting "time"="{now}" on "{now}" for current site
```

この例では、time 変数を現行サイトの現在時刻に即座に設定しています。

```
setting "division"="%22design group%22" on "15 Mar 2007 17:05:46 gmt" for client
```

この例では、%xx を使用して、その 16 進数に相当する特殊文字を指定しています。この場合、%22 を使用して、変数の値を二重引用符で囲んでいます。

注

クライアントがリセットされた場合、設定の発効日は削除され、後続の setting コマンドで上書きされます。クライアントがリセットされる可能性があるのは、コンピューター ID が競合する場合 (最も多いのは、複数のシステムにコピーされるイメージに同じコンピューター ID が誤って含まれてしまう場合)、新規サーバーに対するアクション・サイト・マストヘッドが変更された場合、新規 ID の収集指示がクライアントに対して発行された場合などです。

次に実行されるアクションによって新しい発効日が設定されますが、設定*値*はリセット前の値と同じになります。この値にはリレー選択などに関する情報が含まれるため、同じ値が保持されます。このように、デプロイメントのリセットが行われても、ネットワーク・リレー構造をリセットするための新規アクションを発行する必要はありません。

バージョン 5.1 以降

setting delete

このアクションは、クライアント・コンピューター上の指定された setting 変数を削除します。これには、元の設定のタイム・スタンプと比較されるタイム・スタンプも含まれます。削除日付が設定日付よりも新しい場合、設定が削除されます。それ以外の場合、削除コマンドは無視されます。

構文

setting delete "<name>" on "<date>" for client

setting "<name>" on "<date>" for site "<site_url>"

name は削除する設定を表し、**date** は設定を削除する日付です。**クライアント**・コンピューター上または指定された**サイト**上の設定を削除できます。

例

```
setting delete "name" on "14 Apr 2007 21:09:36 gmt" for client
```

この例では、指定されたタイム・スタンプが、対応する設定時間より新しい場合、クライアント・マシン上の「name」変数が削除されます。それ以外の場合、削除コマンドは無視されます。

```
setting delete "abc" on "{now}" for site "siteurl"
```

この例では、指定されたサイト上の「abc」変数が即座に削除されます。

```
setting delete "abc" on "{now}" for current site
```

この例では、現行サイト上の「abc」変数が即座に削除されます。

バージョン 5.1 以降

第 6 章 レジストリー・コマンド

regdelete

このコマンドは、現在存在しているかどうかにかかわらず、指定された名前のレジストリー・キー 値を削除します。

構文

regdelete "<registry key>" "<value name>"

registry key はキーの名前で、**value name** は削除するレジストリー・キー内の値です。

例

```
regdelete "[HKEY_CLASSES_ROOT¥ShellScrap]" "NeverShowExt"
```

この例では、指定されたレジストリー・キーから NeverShowExt 値が削除されます。

注

このコマンドは Windows 専用です。UNIX エージェントでは、アクション・スクリプトが終了します。

空でないレジストリー・キーとそのすべてのサブキーを削除するには、以下のようなファイルを「del.reg」などの名前で作成する必要があります。

```
REGEDIT4
[-HKEY_CURRENT_USER¥keep¥removethisandbelow]
```

このファイルには 3 行が必要で、最終行はブランクにする必要があります。レジストリー・パスの先頭にダッシュ (-) があることに注意してください。

ここで、以下のようなアクションを実行できます。

```
regedit /s del.reg
```

このアクションを実行すると、removethisandbelow というキーと、そのすべてのサブキーが削除されます。**appendfile** コマンドを使用して、この .reg ファイルを作成できます。

指定されたキーがまだ存在しない場合は、このコマンドによって作成されます。

バージョン 5.1 以降 -- Windows のみ

regset

このコマンドは、レジストリー・キーを、特定の名前と値に設定します。キーがまだ存在しない場合、このコマンドにより、その初期値を使用してキーが作成されます。

構文

regset "<registry key>" "<value name>"=<value>

registry key は対象となるキーで、**value name** は **value** に設定するキー値です。これらの値は、Regedit (レジストリーを編集する Windows プログラム) のルールに従って、REGEDIT4 レジストリー・ファイル内の値のとおりに入力します。ストリング値は引用符で区切り、標準的な 4 バイト整数 (dword) は DWORD を使用して識別します。以下に示すように、その後に 16 進数 (先行ゼロ) で入力した数値が続きます。

例

```
regset "[HKEY_CURRENT_USER¥Software¥Microsoft¥Office¥9.1¥Word¥Security]"
  "Level"=dword:00000002
```

この例では、指定されたレジストリーの Level 値をダブルワード 2 に設定しています。

```
regset "[HKEY_CURRENT_USER¥Software¥BigCorp Inc.]" "testString"="bob"
```

この例では、指定されたレジストリー・キーの testString 値を bob に設定しています。

```
regset "[HKEY_CLASSES_ROOT¥ShellScrap]" "AlwaysShowExt"=""
```

この例では、指定されたレジストリー値のデータをクリアしています。

注

このコマンドは Windows 専用です。UNIX エージェントでは、アクション・スクリプトが終了します。

これらの例では、大括弧 [ ] を使用してレジストリー・キーの名前を囲んでいることに注意してください。前述のとおり、これは REGEDIT4 レジストリー・ファイルのルールに従っています。この構文は、RegSet コマンドには必要ですが、レジストリー・インスペクターでは不要です。

regset コマンドを使用した場合、IBM Endpoint Manager クライアントによって .reg ファイルが動的に作成され、結果の .reg ファイルが自動的に実行されることに注意してください。このコマンドを使用しない場合は、このファイルを手動で作成してレジストリーを更新する必要があります。.reg ファイルのルールの 1 つとして、**value** フィールドの ¥ は、二重の円記号 (¥¥) として指定する必要があります。そのため、レジストリー・キー HKEY_LOCAL_MACHINE¥SOFTWARE¥Microsoft¥Windows NT¥CurrentVersion の値 SourcePath2 に c:¥I386 を割り当てる場合、以下のようなコマンド定義になります。

```
regset "[HKEY_LOCAL_MACHINE¥SOFTWARE¥Microsoft¥Windows NT¥CurrentVersion]"
  "SourcePath2"="c:¥¥I386"
regset "[HKEY_LOCAL_MACHINE¥SOFTWARE¥Microsoft¥Windows NT¥CurrentVersion]"
  "SourcePath2"={escape of "c:¥I386"}
```

2 番目の例では、**escape** 関連句を使用して、バックスラッシュを二重バックスラッシュに自動変換しています。

多数の regset コマンドを発行する必要がある場合、**appendfile** コマンドまたは **createfile until** コマンドを使用して、正しくフォーマット設定された regedit ファイルを作成してから、regedit をサイレント実行することもできます。

```
Createfile until end-reg-edit-commands
      REGEDIT4
[HKEY_LOCAL_MACHINE¥Software¥Microsoft¥Windows NT¥CurrentVersion]
  "SourcePath1"="c:¥¥I386"
  "SourcePath2"="{escapes of pathname of windows folder}"
end-reg-edit-commands
move __createfile setup.reg
wait regedit /s setup.reg
```

指定されたキーがまだ存在しない場合は、このコマンドによって作成されます。

バージョン 5.1 以降 -- Windows のみ

第 7 章 Wow64 コマンド

action uses wow64 redirection

このコマンドを使用すると、クライアントは、新しい 64 ビット・バージョンの Windows オペレーティング・システム (Windows 2003 x64 や Windows XP Pro x64 など) に組み込まれている **Windows On Windows64** (Wow64) 機能により、そのクライアント用に構成された 32 ビット環境から Wow64 環境にリダイレクトされます。

64 ビット OS でのアクションで、**true** の値を使用してこのコマンドを実行すると、クライアントにより、ファイル名を含む後続のすべてのコマンドで Wow64 リダイレクトが有効になります。この状態は、アクションが完了するまで、またはクライアントが **action uses wow64 redirection false** コマンドを実行するまで、継続します。

関連度置換を使用して、<true|false> の値を指定できます。Wow64 が提供するファイル・システム・リダイレクトは、Wow64DisableWow64FsRedirection という Windows API を使用すると無効になり、Wow64RevertWow64FsRedirection という Windows API を使用すると再度有効になります。

この設定により影響を受けるコマンドには、以下のものがあります。

- dos
- run、wait、rundetached、waitdetached、runhidden、waithidden
- delete、copy、move、open

構文

action uses wow64 redirection<true|false>

例

```
action uses wow64 redirection true
```

この例では、Wow64 リダイレクトをオンにしています。

```
action uses wow64 redirection false
```

この例では、Wow64 リダイレクトをオフにしています。

注

このコマンドは Windows 専用です。UNIX エージェントでは、アクション・スクリプトが終了します。

バージョン 6.0 以降 -- Windows のみ

regdelete64

Regdelete64 では、**regdelete** コマンドと同じ構文を使用しますが、64 ビット・バージョンの Regedit を起動してレジストリーを設定する前に、Wow64DisableWow64FsRedirection の呼び出しを行います。これにより、64 ビット・マシンで使用可能な 64 ビット・レジストリーを使用できるようになります。このコマンドは、指定された名前を持つレジストリー・キー値を削除します。値がまだ存在しない場合、このコマンドは失敗し、後続のコマンドは何も実行されません。

構文

regdelete64 "<registry key>" "<value name>"

registry key はキーの名前で、**value name** は削除するレジストリー・キー内の値です。

例

```
regdelete64 "[HKEY_CLASSES_ROOT¥ShellScrap]" "NeverShowExt"
```

この例では、指定されたレジストリー・キーから NeverShowExt 値が削除されます。

注

このコマンドは Windows 専用です。UNIX エージェントでは、アクション・スクリプトが終了します。

指定されたキーがまだ存在しない場合は、このコマンドによって作成されます。

バージョン 6.0 以降 -- Windows のみ

regset64

Regset64 では、**regset** コマンドと同じ構文を使用しますが、64 ビット・バージョンの Regedit を起動してレジストリーを設定する前に、Wow64DisableWow64FsRedirection の呼び出しを行います。これにより、ネイティブの 64 ビット・レジストリーを使用して、レジストリー・キーを特定の名前と値に設定できるようになります。キーがまだ存在しない場合、このコマンドにより、その初期値を使用してキーが作成されます。

構文

regset64 "<registry key>" "<value name>"=<value>

registry key は対象となるキーで、**value name** は **value** に設定するキー値です。これらの値は、Regedit (レジストリーを編集する Windows プログラム) のルールに従って、REGEDIT4 レジストリー・ファイル内の値のとおりに入力します。ストリング値は引用符で区切り、標準的な 4 バイト整数 (dword) は DWORD を使用して識別します。以下に示すように、その後に 16 進数 (先行ゼロ) で入力した数値が続きます。

例

```
regset64 "[HKEY_CURRENT_USER¥Software¥Microsoft¥Office¥9.1¥Word¥Security]"
    "Level"=dword:00000002
```

この例では、指定されたレジストリーの Level 値をダブルワード 2 に設定しています。

```
regset64 "[HKEY_CURRENT_USER¥Software¥BigCorp Inc.]" "testString"="bob"
```

この例では、指定されたレジストリー・キーの testString 値を bob に設定しています。

```
regset64 "[HKEY_CLASSES_ROOT¥ShellScrap]" "AlwaysShowExt"=""
```

この例では、指定されたレジストリー値のデータをクリアしています。

注:

このコマンドは Windows 専用です。UNIX エージェントでは、アクション・スクリプトが終了します。

これらの例では、大括弧 [ ] を使用してレジストリー・キーの名前を囲んでいることに注意してください。前述のとおり、これは REGEDIT4 レジストリー・ファイルのルールに従っています。この構文は、RegSet コマンドには必要ですが、レジストリー・インスペクターでは不要です。

regset64 コマンドを使用した場合、IBM Endpoint Manager クライアントによって .reg ファイルが動的に作成され、結果の .reg ファイルが自動的に実行されることに注意してください。このコマンドを使用しない場合は、このファイルを手動で作成してレジストリーを更新する必要があります。.reg ファイルのルールの 1 つとして、**value** フィールドの ¥ は、二重のバックスラッシュ (¥¥) として指定する必要があります。

そのため、レジストリー・キー HKEY_LOCAL_MACHINE¥SOFTWARE¥Microsoft¥Windows NT¥CurrentVersion to c:¥I386 の値 SourcePath2 を割り当てる場合、以下のような定義コマンドになります。

```
regset64 "[HKEY_LOCAL_MACHINE¥SOFTWARE¥Microsoft¥Windows NT¥CurrentVersion]"
   "SourcePath2"="c:¥¥I386"
regset64 "[HKEY_LOCAL_MACHINE¥SOFTWARE¥Microsoft¥Windows NT¥CurrentVersion]"
   "SourcePath2"={escape of "c:¥I386"}
```

2 番目の例では、**escape** 関連句を使用して、バックスラッシュを二重バックスラッシュに自動変換しています。

指定されたキーがまだ存在しない場合は、このコマンドによって作成されます。

バージョン 6.0 以降 -- Windows のみ

script64

Script64 では、**script** コマンドと同じ構文を使用しますが、スクリプトの実行前に Wow64DisableWow64FsRedirection の呼び出しを行います。これにより、ネイティブの 64 ビット・スクリプト・コマンドを発行して、64 ビット・プロセッサー上に構築されている Windows 32 ビット環境をバイパスできます。

script キーワードは、指定された名前を持つ外部スクリプト (JavaScript や Visual Basic などのスクリプト言語で作成されたもの) を実行します。適切なスクリプト・エンジンがインストールされていない場合、またはスクリプトを実行できない場合、script キーワードが記述されたアクション・スクリプトは終了します。指定されたスクリプトが終了するまで、アクション・シェル・コマンドの次の行は実行されません。

構文

script64 <script name>

例

```
script64 attrib.vbs
```

Visual BASIC スクリプト attrib.vbs をネイティブの 64 ビット・モードで実行します。

注

このコマンドは Windows 専用です。UNIX エージェントでは、アクション・スクリプトが終了します。

Windows コンピューターでは、このコマンドは、Wow64DisableWow64FsRedirection を呼び出し、その後で Windows から wscript "scriptName" ステートメントを発行した場合と同じ動作になります。

バージョン 6.0 以降 -- Windows のみ

第 8 章 管理権限コマンド

administrator add

このコマンドでは、特定の IBM Endpoint Manager クライアントを管理する特定のユーザーを指名できます。これを行うには、発効日を持ち、パラメーターとして渡される設定を使用します。この日付はオプションではありません。発効日のテストは、通常の**設定**の場合と同様です。

構文

administrator add<administrator name> on <date>

例

```
administrator add "bob" on "21 Aug 2002 17:39:14 gmt"
```

この例では、bob というコンソール・オペレーターに、対象のコンピューターの管理権限を割り当てることができます (指定された日付に有効になります)。

バージョン 5.1 以降

administrator delete

このコマンドにより、指定された管理者の管理権限を削除できます。これを行うには、発効日を持ち、パラメーターとして渡される設定を使用します。この日付はオプションではありません。発効日のテストは、通常の**設定**の場合と同様です。

構文

administrator delete<administrator name> on <date>

例

```
administrator delete "bob" on "21 Aug 2002 17:39:14 gmt"
```

この例では、bob というコンソール・オペレーターの管理権限が、指定された日付に削除されます。

バージョン 5.1 以降

第 9 章 BigFix クライアントのメンテナンス・コマンド

module add

このコマンドは、指定されたインスペクター・ライブラリー・ファイルを、クライアントが使用するインスペクター・ライブラリー・セットに追加します。インスペクター・ライブラリーを置き換える場合、module add コマンドでそのライブラリーを指定するだけでなく、**module delete** コマンドでも指定する必要があります。アクションを完了するには、**module commit** コマンドを発行する必要があります。

構文

module add "<module name>"

例

```
module add "dellinspect.dll"
```

注

内部使用専用です。

バージョン 5.1 以降

module commit

add コマンドと delete コマンドは、インスペクター・ライブラリーに対する変更をコミットするためのステージを設定します。commit コマンドは、実際の削除と追加を実行します。

構文

module commit

例

```
delete "dellinspect.dll"
copy "{pathname of client folder of site "dell"}¥dellinspect.dll""dellinspect.dll"
module add "dellinspect.dll"
module commit
```

注

内部使用専用です。

バージョン 5.1 以降

module delete

このコマンドは、指定されたインスペクター・ライブラリー・ファイルを削除対象としてマークします。アクションを完了するには、**module commit** コマンドを発行する必要があります。

構文

module delete "<module name>"

例

```
module delete "inspectors.dll"
```

注

内部使用専用です。

バージョン 5.1 以降

第 10 章 ロック・コマンド

action lock indefinite

このコマンドは、アクション・ロックをオンにします。これは発効日から開始され、有効期限が切れることはありません。日付は MIME 時刻形式です (15 Mar 2007 12:42:51 -0700 など)。

構文

action lock indefinite "<effective date>"

例

```
action lock indefinite "{now}"
```

この例では、アクション・ロックが即座にオンになります。

バージョン 5.1 以降

action lock until

このコマンドは、発効日から有効期限に達するまで、アクションをロックします。有効期限は MIME 時刻形式です (19 Jul 2007 12:42:51 -0700 など)。インスペクターとともに {now} などの置換を使用できます。この置換は、時間を評価してストリングに挿入します。

構文

action lock until "<expire date>" "<effective date>"

例

```
action lock until "{now + 3*days}" "{now}"
```

この例では、アクションが即座にロックされ、3 日後にロック解除されます。

```
action lock until "{apparent registration server time + 10 * minutes}" "{apparent
    registration server time}"
```

この例では、現在の **apparent registration server time** (クライアントが最後にサーバーに登録された時間に基づく) を使用して、アクションが 10 分間ロックされます。

バージョン 5.1 以降

action unlock

このコマンドは、クライアントのロックを解除し、すべてのアクションでの動作を可能にします。発効日フィールドを使用して、作成された順序どおりにアクションのロックとロック解除が実行されます。日付は MIME 時刻形式です (29 Nov 2008 12:42:51 -0700)。

構文

action unlock "<effective date>"

例

```
action unlock "{now}"
```

この例では、アクションのロックが即座に解除されます。

バージョン 5.1 以降

第 11 章 サイト・メンテナンス・コマンド

site force evaluation

このコマンドは、クライアントを使用して、当該サイトのすべての Fixlet を再評価します。このコマンドは、ファイルまたは設定の更新後に、可能な限り早急にサイト全体についての Fixlet の関連度を再計算する場合に便利です。

構文

site force evaluation

例

```
site force evaluation
```

バージョン 5.1 以降

site gather schedule disable

このコマンドは、現在のサイトからのスケジュール済み収集を無効にします。アクション・サイトの場合、このコマンドは無効です。

構文

site gather schedule disable

例

```
site gather schedule disable
```

バージョン 5.1 以降

site gather schedule manual

このコマンドは、現在のサイトからの手動収集を有効にします。アクション・サイトの場合、このコマンドは無効です。

構文

site gather schedule manual

例

```
site gather schedule manual
```

バージョン 5.1 以降

site gather schedule publisher

このコマンドは、現在のサイトから、当該サイトのマストヘッドで指定されているサイトへの収集スケジュールを設定します。

構文

site gather schedule publisher

例

```
site gather schedule publisher
```

バージョン 5.1 以降

site gather schedule seconds

このコマンドは、元のサイトからの収集スケジュールを、指定された秒数に設定します。

構文

site gather schedule seconds<seconds>

例

```
site gather schedule seconds 360
```

この例では、サイト収集スケジュールを 6 分に設定しています。

バージョン 5.1 以降

subscribe

このコマンドは、マストヘッド・ファイルで指定されているサイトに対して、クライアントをサブスクライブします。IBM Endpoint Manager コンソールには、サイトの追加を自動化するための「**サイトを管理**」ダイアログが用意されています。

構文

subscribe "<masthead file name>"

例

```
subscribe "__Download¥Sitename.fxm"
```

注

IBM Endpoint Manager では、このコマンドがマスター・アクション・サイトでアクションとして実行されなかった場合、エラーが返されます。このコマンドは、Enterprise Fixlet サイトに対するクライアントのサブスクライブと、アクション・サイト・マストヘッド・ファイルの更新を行う場合に便利です。

バージョン 5.1 以降

unsubscribe

このコマンドは、現在の Enterprise Fixlet サイトから自動的にサブスクライブを解除します。

構文

unsubscribe

例

```
unsubscribe
```

バージョン 5.1 以降

第 12 章 コメント

二重スラッシュ

// で始まる行はコメントであり、アクションを実行する際には無視されます。

構文

//

例

```
// The following command will replace the file on the C drive:
copy "{name of drive of windows folder}¥win.com" "{name of drive of windows
       folder}¥bigsoftware¥win.com"
```

二重スラッシュを使用することで、アクション・スクリプトをコメント化することができます。

バージョン 5.1 以降

付録. サポート

この製品について詳しくは、以下のリソースを参照してください。

- IBM Knowledge Center
- IBM Endpoint Manager サポート・サイト
- IBM Endpoint Manager Wiki
- 知識ベース
- フォーラムおよびコミュニティー

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒103-8510
東京都中央区日本橋箱崎町19番21号
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

以下の保証は、国または地域の法律に沿わない場合は、適用されません。 IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。 IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。 IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

表示されている IBM の価格は IBM が小売り価格として提示しているもので、現行価格であり、通知なしに変更されるものです。卸価格は、異なる場合があります。

本書はプランニング目的としてのみ記述されています。記述内容は製品が使用可能になる前に変更になる場合があります。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラット

フォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。これらのサンプル・プログラムは特定物として現存するままの状態で提供されるものであり、いかなる保証も提供されません。 IBM は、お客様の当該サンプル・プログラムの使用から生ずるいかなる損害に対しても一切の責任を負いません。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生的創作物にも、次のように、著作権表示を入れていただく必要があります。

このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。

商標

IBM、IBM ロゴおよび ibm.com は、世界の多くの国で登録された International Business Machines Corporation の商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、 http://www.ibm.com/legal/copytrade.shtml をご覧ください。

Adobe、PostScript は、Adobe Systems Incorporate の米国およびその他の国における登録商標または商標です。

IT Infrastructure Library は英国 Office of Government Commerce の一部である the Central Computer and Telecommunications Agency の登録商標です。

インテル、Intel、Intel ロゴ、Intel Inside、Intel Inside ロゴ、Centrino、Intel Centrino ロゴ、Celeron、Xeon、Intel SpeedStep、Itanium、および Pentium は、Intel Corporation または子会社の米国およびその他の国における商標または登録商標です。

Linux は、Linus Torvalds の米国およびその他の国における登録商標です。

Microsoft、Windows、Windows NT および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

ITIL は英国 The Minister for the Cabinet Office の登録商標および共同体登録商標であって、米国特許商標庁にて登録されています。

UNIX は The Open Group の米国およびその他の国における登録商標です。

Java™ およびすべての Java 関連の商標およびロゴは Oracle やその関連会社の米国およびその他の国における商標または登録商標です。

Cell Broadband Engine は、Sony Computer Entertainment, Inc.の米国およびその他の国における商標であり、同社の許諾を受けて使用しています。

Linear Tape-Open、LTO、LTO ロゴ、Ultrium および Ultrium ロゴは、HP、IBM® Corp. および Quantum の米国およびその他の国における商標です。

製品資料に関するご使用条件

これらの資料は、以下の条件に同意していただける場合に限りご使用いただけます。

適用条件

このご使用条件は、IBM Web サイトのすべてのご利用条件に追加して適用されます。

個人使用

これらの資料は、すべての著作権表示その他の所有権表示をしていただくことを条件に、非商業的な個人による使用目的に限り複製することができます。ただし、IBM の明示的な承諾をえずに、これらの資料またはその一部について、二次的著作物を作成したり、配布 (頒布、送信を含む) または表示 (上映を含む) することはできません。

商業的使用

これらの資料は、すべての著作権表示その他の所有権表示をしていただくことを条件に、お客様の企業内に限り、複製、配布、および表示することができます。ただし、IBM の明示的な承諾をえずにこれらの資料の二次的著作物を作成したり、お客様の企業外で資料またはその一部を複製、配布、または表示することはできません。

権利

ここで明示的に許可されているもの以外に、資料や資料内に含まれる情報、データ、ソフトウェア、またはその他の知的所有権に対するいかなる許可、ライセンス、または権利を明示的にも黙示的にも付与するものではありません。

資料の使用が IBM の利益を損なうと判断された場合や、上記の条件が適切に守られていないと判断された場合、IBM はいつでも自らの判断により、ここで与えた許可を撤回できるものとさせていただきます。

お客様がこの情報をダウンロード、輸出、または再輸出する際には、米国のすべての輸出入関連法規を含む、すべての関連法規を遵守するものとします。

IBM は、これらの資料の内容についていかなる保証もしません。これらの資料は、特定物として現存するままの状態で提供され、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任なしで提供されます。

Printed in Japan

日本アイ・ビー・エム株式会社
〒103-8510 東京都中央区日本橋箱崎町19-21