

WebSphere Application Server V9

Liberty 基盤設計セミナー

運用設計



アジェンダ

- Admin Center
- 起動停止・状況確認
- 構成変更
- アプリケーションのデプロイ
- Libertyサーバーのデプロイ
- バックアップ・リストア、プロセス監視
- ログ・ログ監視……問題判別のセッションを参照

アジェンダです。ログ・ログ監視については、「問題判別」のセッションを参照してください。

Admin Center

Admin Center

使用するフィーチャー : adminCenter-1.0

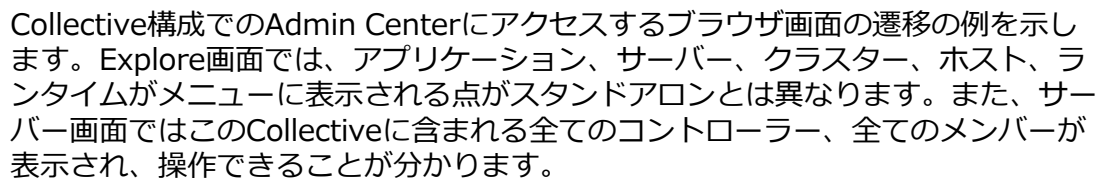
□ Admin CenterによりWebブラウザから各種の操作が可能

- <https://<hostname>:<port>/adminCenter/> を開く
 - サーバーの起動停止・状況確認
 - アプリケーションの起動停止・状況確認
 - パフォーマンス・モニター
 - 構成情報 (server.xml) の参照・更新…… (「構成変更」で後ほど説明)
 - Libertyクラスターの起動停止・状況確認 (Collective環境のみ)
 - Libertyサーバー・パッケージの配布 (Collective環境のみ)



Admin CenterはWAS traditionalの管理コンソールに類似した位置づけの機能で、Webブラウザから各種の操作が可能になっています。Admin Centerが登場した当初は限られた機能でしたが、継続デリバリーにより機能が拡充されてきました。アクセスURLや実施可能な項目はチャートの通りです。構成情報の参照・更新機能については、後ほど「構成変更」のセクションで説明します。また、スタンドアロン構成での画面遷移の例を、チャートの中に示します。使用するフィーチャーは、adminCenter-1.0です。

Collective構成でのAdmin Center利用例



Admin Center によるパフォーマンス・モニター

□ Admin Centerから各種パフォーマンス値がリアルタイムでモニター可能

- JVMのパフォーマンス値
 - ヒープ使用量、ロードされたクラス数、ActiveなJVMスレッド数、CPU利用率
- Libertyサーバーのパフォーマンス値
 - アクティブ・セッション数、アクティブLibertyスレッド数、平均応答時間、平均待ち時間、リクエスト数、利用中コネクション数
 - server.xmlでmonitor-1.0フィーチャーの設定が必要



(※) 多少のオーバーヘッド負荷はあるが本番環境での使用も可能

© 2016 IBM Corporation

Admin Centerのパフォーマンス・モニター機能では、JVMのパフォーマンス値とLibertyサーバーのパフォーマンス値がリアルタイムでモニターできます。モニター可能な項目はチャート中の通りです。リアルタイム・モニターは多少のオーバーヘッド負荷がありますが、本番環境での使用も可能です。

起動停止・状況確認

コマンドによる起動停止・状況確認

serverコマンドでの起動停止

```
<WLP>%bin>server start server1
```

サーバー server1 を起動中です。
サーバー server1 が起動しました。

```
<WLP>%bin>server stop server1
```

サーバー server1 を停止中です。
サーバー server1 は停止しました。

serverコマンドでの起動（フォアグラウンドで起動する場合）

```
<WLP>%bin>server run server1
```

IBM J9 VM バージョン pwa64forks-20120419_01 (SR1+IV19490+IV19661) (ja_JP) で、
server1 (WebSphere Application Server 8.5.5.6/wlp-1.0.9.d50620150610-1749) を起
動しています

[AUDIT] CWWKE0001I: サーバー server1 が起動されました。
(以下略)

serverコマンドでの状況確認

```
<WLP>%bin>server status server1
```

サーバー server1 は稼働中です。

- serverコマンドにより、起動・停止・状況確認など、Libertyサーバーの基本操作ができます。

Libertyサーバーは、serverコマンドのstartアクション・stopアクションを実行して起動・停止を行うことができます。起動には、startアクションに加えて、runアクションを利用することもできます。runアクションではフォアグラウンドでLibertyが起動されるため、稼働中はプロンプトは戻りません。また、Libertyサーバーの状況確認には、statusアクションを利用します。

Liberty Collective の起動停止・状況確認

serverコマンドでControllerやMemberを個々に起動停止することも可能

```
<WLP>%bin>server start controller1
```

サーバー **controller1** を始動中です。
サーバー **controller1** が始動しました。

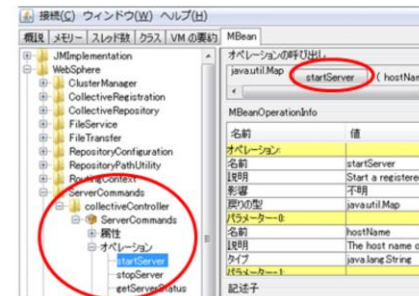
```
<WLP>%bin>server stop member1
```

サーバー **member1** を停止中です。
サーバー **member1** は停止しました。

Jconsole等でMbeanにアクセスすることでMemberやクラスター単位で起動停止することも可能

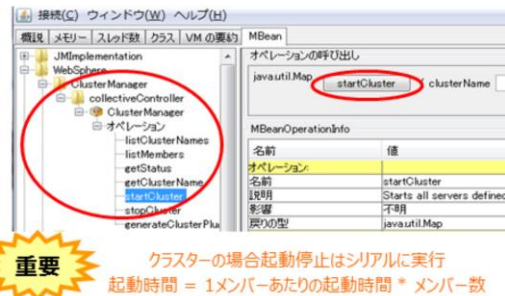
【Liberty Collective 構成の場合】

Controllerから提供される ServerCommands MBean の実行によりMemberの起動停止が可能



【Libertyクラスター構成の場合】

Controllerから提供される ClusterManager MBean の実行によりクラスターの起動停止が可能



重要

クラスターの場合起動停止はシリアルに実行
起動時間 = 1メンバーあたりの起動時間 * メンバー数

5

© 2016 IBM Corporation

Liberty Collective 構成の場合、Collective Member 個々は前ページと同様に serverコマンドのstartアクション・stopアクションを実行して起動・停止を行うことができるのに加えて、Collective Controller で提供されるServerCommands MBean のstartServer・stopServerのオペレーションを実行することで、Collective Member の起動・停止を行うこともできます。

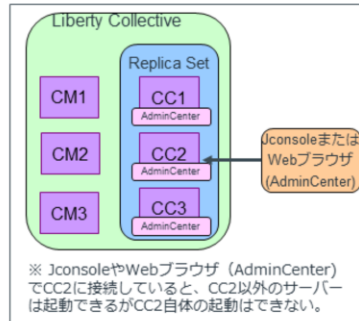
Libertyクラスター構成の場合、Collective Controller で提供される ClusterManager MBean のstartCluster・stopClusterのオペレーションを実行することで、Libertyクラスター全体の起動・停止を行います。ここで注意が必要なのは、Libertyクラスター構成の場合、起動・停止は各Memberシリアルに実行されるため、Libertyクラスター全体の起動・停止時間は、1メンバーあたりにかかる起動・停止時間とメンバー数の積になり、Member数が多ければそれだけ多くの時間がかかることになります。

起動停止・状況確認方法の比較

□ 起動停止・状況確認には4種類の方法がある

- コマンド (server start/stop/status)
- Jconsole (MBean)
- Admin Center
- Eclipse+WDT

□ 起動停止・状況確認方法の比較表



	スタンドアロン構成	Collective構成			補足
		コントローラー	単独メンバー	クラスターメンバー	
server コマンド	○	○ (個別に)	○ (個別に)	○ (個別に)	基本ローカルのみ リモートはSSH等で
Jconsole (MBean)	× (MBeanなし)	○ (個別に) Jconsole接続先サーバーの起動はできない (※)	○ (個別に)	◎ (クラスター毎)	リモートも可能
Admin Center	△起動は不可	○ (個別に) Webブラウザ接続先サーバーの起動はできない (※)	○ (個別に)	◎ (クラスター毎)	リモートも可能
Eclipse+ WDT	○	○ (個別に)	○ (個別に)	○ (個別に)	リモートも可能 開発環境・テスト環境での利用が一般的

起動停止・状況確認についてまとめます。方法としては、チャートの通り4種類あります。

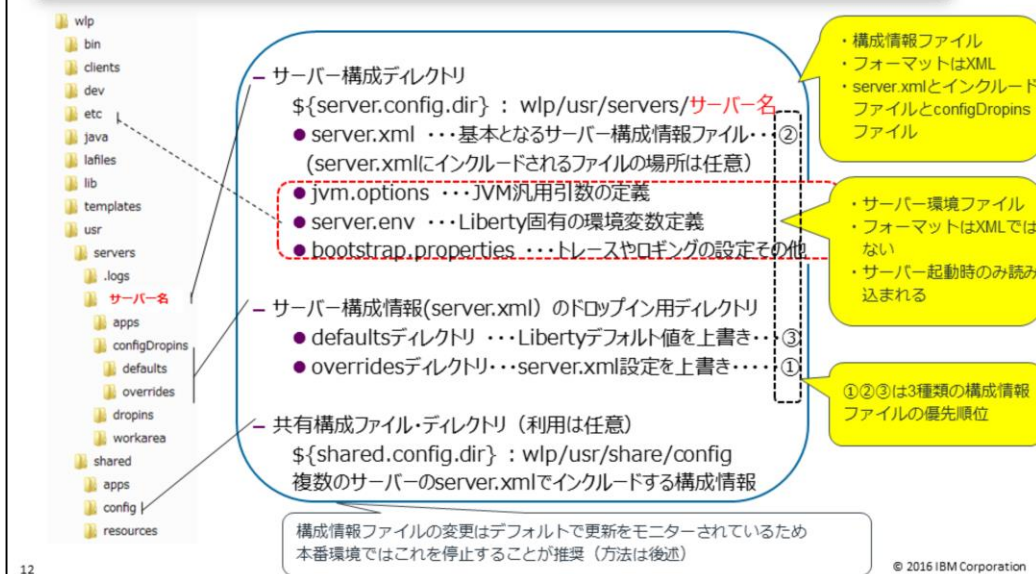
これを比較したのがチャート中の表です。右上の図はCollective構成において、JconsoleやAdmin Centerの接続先であるコントローラーは、停止はできるが起動はできないことを説明しています。共に、それ以外のコントローラーやメンバーは起動・停止ができます。

比較表から分かる通り、クラスターの扱いに関しては、JconsoleやAdmin Centerが優れていますが、全般的に見た場合、コマンドとWDTが優れています。また、WDTについては後ほど「構成変更」でも説明します。

構成変更

構成に関するファイル

構成情報ファイルとサーバー環境ファイルの分類



構成に関するファイルを復習を兼ねて、簡単にまとめておきます。

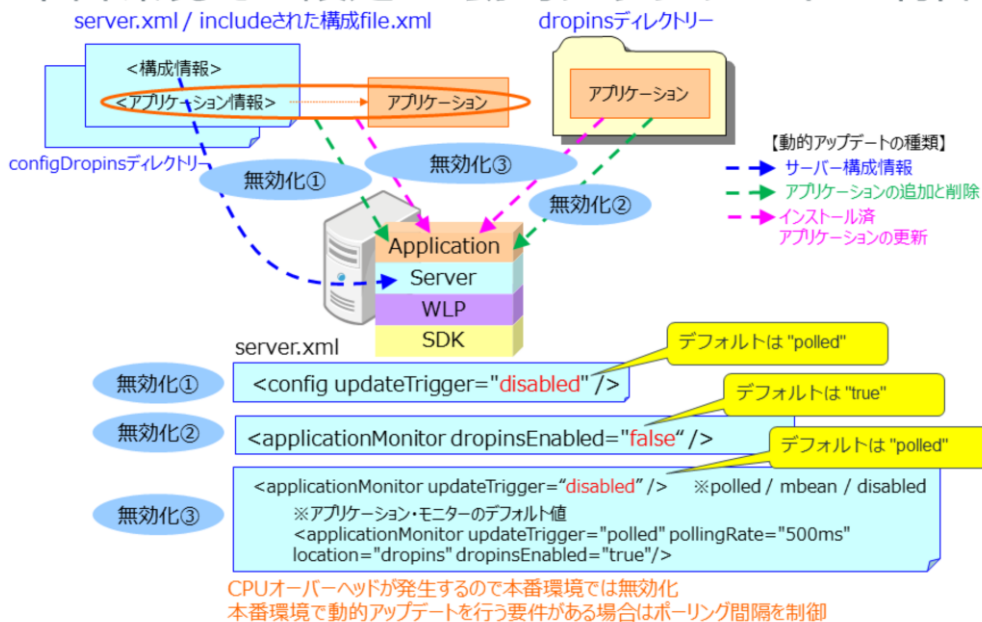
大きく分けると、構成情報ファイルとサーバー環境ファイル（赤点線枠内）の2種類があります。前者はXMLフォーマットであり、設定により動的更新が可能なファイル群です。後者はXMLフォーマットではないファイル群で、サーバー起動時のみ読み込まれる点で、前者とは扱いが異なります。

構成情報ファイルの基本となるファイルはserver.xmlファイルで、これにインクルードされるファイルは任意の場所に置くことが可能です。他のサーバー構成と共有される場合には `usr/share/config` の下に置くのが一般的です。

また、`configDropins`の下にはserver.xmlを補足する位置づけの構成情報ファイルを置くことができます。`configDropins/defaults` はLibertyの持つデフォルト値を上書きします。また、`configDropins/overrides` はserver.xmlを上書きします。

優先順位の観点では、点線枠内の①②③の通りです。

本番環境での設定 -- 動的アップデートの制御



13

© 2016 IBM Corporation

Libertyプロファイルは、サーバー構成やアプリケーション情報を動的にアップデートすることができますが、その機能を必要に応じて制御することができます。

動的アップデートの種類としては、サーバー構成情報、アプリケーションの追加と削除、インストール済アプリケーションの更新、の3つがあります。サーバー構成情報は、server.xmlあるいはincludeされた構成ファイルとconfigDropinsディレクトリーに置かれた構成ファイルから動的アップデートを行うことができます。アプリケーションの追加と削除およびインストール済アプリケーションの更新は、server.xmlあるいはincludeされた構成ファイルやアプリケーション配置ディレクトリー、dropinsディレクトリーから動的アップデートを行うことができます。これらの動的アップデート機能に関して、それぞれの方法で無効化あるいはモニターのポーリング間隔の変更を行うことができます。

構成情報をモニターする構成サービスやアプリケーション・モニターによるポーリングはオーバーヘッドが発生するので、本番環境では無効化します。本番環境で動的アップデートを行う要件がある場合は、ポーリング間隔を長く設定するなどして、できるだけオーバーヘッドが少なくなるように対応します。

【KCリンク】

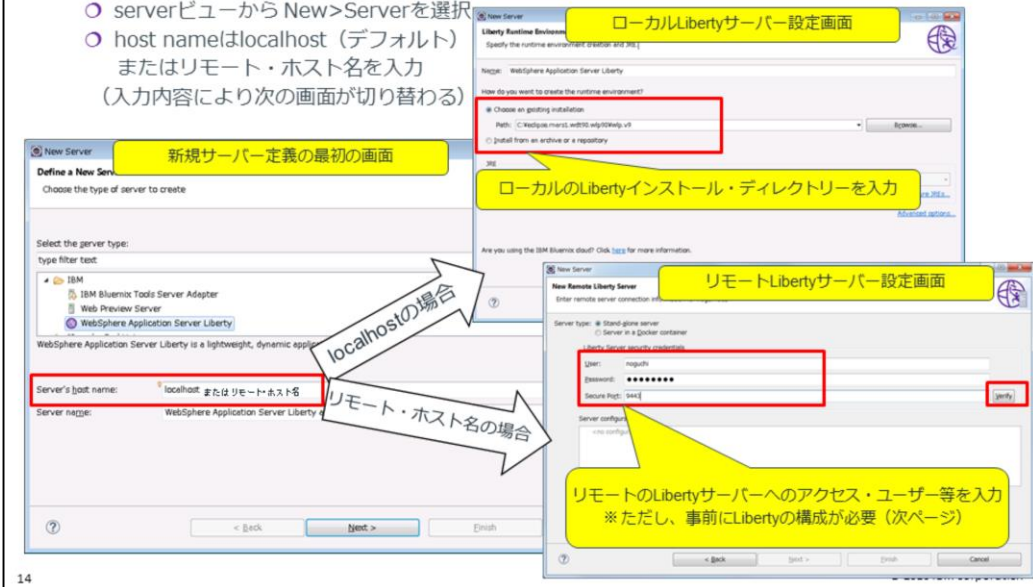
動的更新の制御:

http://www.ibm.com/support/knowledgecenter/ja/SSAW57_9.0.0/com.ibm.wellsphere.wlp.nd.doc/ae/twlp_setup_dyn_upd.html

WDTによる構成変更

❑ eclipse+WDT環境でLibertyサーバー定義を新規作成

- serverビューから New>Serverを選択
- host nameはlocalhost (デフォルト) またはリモート・ホスト名を入力 (入力内容により次の画面が切り替わる)



WDTはEclipseのプラグインで、Libertyサーバー用のアプリケーションを開発したり、構成関連ファイルを作成・編集するのに必要なツールです。従来からローカルのLibertyサーバーを操作することはできませんが、比較的最近になって、リモートのLibertyサーバーにも対応しました。

チャートではその設定方法を示しています。新規サーバー定義の画面で、「Server's host name」に「localhost」（デフォルト値）のまま次に進むと、ローカル構成の画面が表示されます。このフィールドにリモート・ホスト名を入力して次に進むと、リモート構成の画面が表示されます。

リモートLiberty設定画面では、リモートにあるLibertyサーバーのアクセス情報として、ユーザー名、パスワード、ポート番号を入力して、「Verify」を押します。ただし、このWDT構成を行うためには、事前にリモートLibertyサーバーでの設定が必要なため、次のページで、その設定方法を説明します。

WDTによるリモート・アクセスのための事前準備

○ リモートLibertyサーバーのserver.xmlを変更

■ configUtility(.bat/sh) install remoteAdministration

→表示される構成スニペットをserver.xmlにコピー & 修正

```
C:\Program Files\IBM\WebSphere\Liberty\bin>configUtility.bat install remoteAdministration
要求された構成スニペットをダウンロードしています...
<server description="Remote Administration Sample Configuration">
  <!-- NOTE: This file is for reference only. -->
  <!-- Enable restConnector-1.0 feature -->
  <featureManager>
    <feature>restConnector-1.0</feature>
  </featureManager>
  <!-- Simple administrative security configuration. -->
  <!-- TODO: Set the security configuration for Administrative access -->
  <quickStartSecurity userName="{adminUser}" userPassword="{adminPassword}"/>
  <!-- TODO: Set the SSL keystore password -->
  <keyStore id="defaultKeyStore" password="{keystorePassword}"/>
  <!-- TODO: Set HTTP Endpoint attributes -->
  <httpEndpoint id="defaultHttpEndpoint"
    host="*"
    httpPort="9080"
    httpsPort="9443"/>
  <!-- TODO: Use readDir and writeDir to specify directories that remote
  clients are allowed to have read and write access. There can be
  multiple readDir and writeDir elements. Replace writePath and readPath
  variables with your choice of locations or remove them if not needed. -->
  <remoteFileAccess>
    <writeDir>${writePath}</writeDir>
    <readDir>${readPath}</readDir>
  </remoteFileAccess>
</server>
サーバーに対して管理セキュリティが構成されていることを確認してください。
C:\Program Files\IBM\WebSphere\Liberty\bin>
```

<writeDir>は\${server.config.dir}と設定するのが一般的

<readDir>は\${wlp.install.dir}と設定するのが一般的

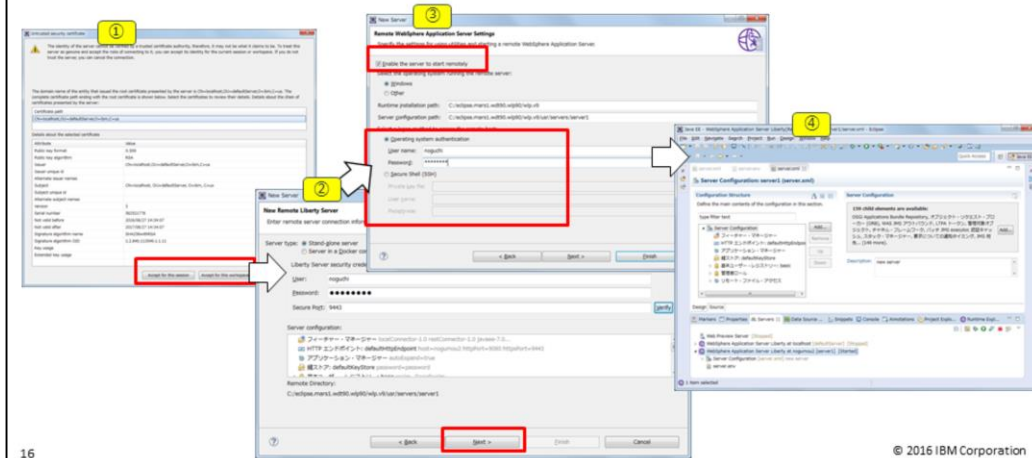
WDTでリモートからアクセスするためには、Libertyサーバーのserver.xmlを更新します。そのための構成スニペットを表示するためには、configUtilityをチャートの通り実行します。また、画面に表示される構成スニペットを枠内に示します。

<remoteFileAccess>要素以下の<writeDir>および<readDir>は黄色噴出しのように設定するとWDTからのアクセスが成功します。

WDTのリモート・アクセス構成

□ 設定手順例 (2ページ前の右下Verifyボタンからの続き)

- ① 証明書確認画面でSession (今回のみ) かWorkspace (記憶する) を選択
- ② リモートLibertyサーバーのサーバー構成を確認してNextに進む
- ③ 「リモート起動」を有効に、リモートOSユーザー情報またはSSH情報を入力
- ④ ローカル同様に起動・停止・構成変更が可能に

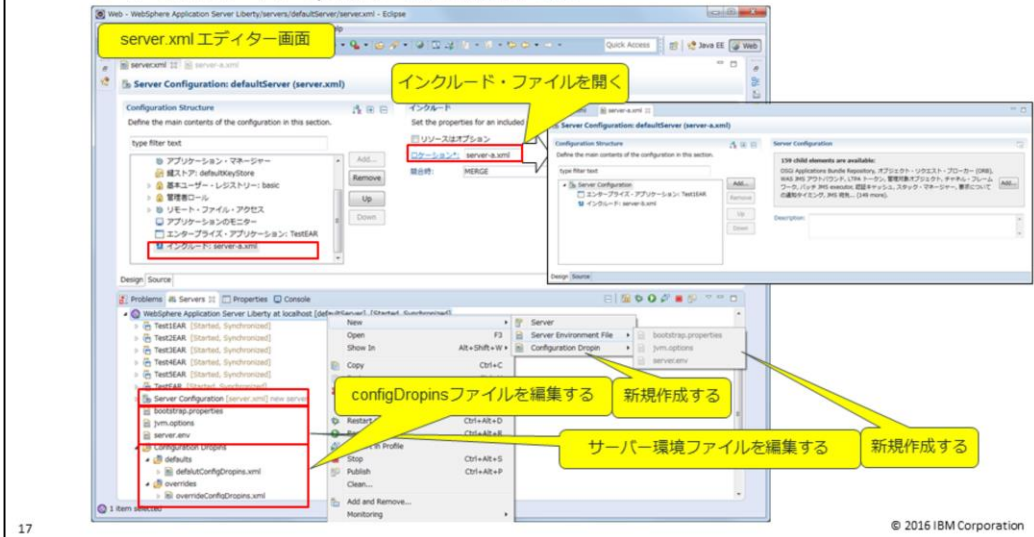


リモートLibertyの設定が完了していると、WDTでのリモートLiberty定義が可能です。チャート中の説明および画面遷移①～④の通りに設定すると、Eclipseのサーバー・ビューにリモートLibertyサーバーを登録できます。

WDTによる構成変更

❑ eclipse+WDTではローカル/リモートともに全ファイルが編集可能

- すべての構成情報ファイルの編集が可能
- すべてのサーバー環境ファイルの編集が可能



eclipse+WDTを使うとローカル/リモートのLibertyサーバーともに、構成関連ファイルである構成情報ファイルとサーバー環境ファイルの全てを作成・更新することが可能です。

構成情報ファイルのserver.xmlはLibertyサーバーを新規作成したタイミングで作成されていますが、これにインクルードするファイルも、<include>要素を追加することで編集可能になります。インクルード・ファイルはネストさせることも可能です。

configDropinsの構成情報ファイルも、画面キャプチャの通り、サーバー・ビュー内でのマウス右クリックから New > Configuration Dropinsから作成できます。

また、サーバー環境ファイルの3種類についても同様に、New > Server Environment File から新規作成することができます。

Admin Centerによる構成の変更

Admin Centerから 構成情報ファイル (※) の変更を行う

- 予め server.xml に設定が必要 (書き込み許可ディレクトリー指定)

```
<remoteFileAccess>
  <writeDir>${server.config.dir}</writeDir>
</remoteFileAccess>
```

- WDTがなくても簡単にserver.xml を編集・保存
- WDT同様に設計ビューとソース・ビューの切り替えが可能



※ 構成情報ファイル (server.xml、インクルードファイル、configDropins) も編集が可能。

18 サーバー環境ファイル (jvm.options, server.env, bootstrap.properties) は扱うことができない。

© 2016 IBM Corporation

Admin Centerで構成変更を行うためには、チャート中の通り、server.xmlを変更しておきます。

Admin Centerでの構成変更画面でも、WDT同様に、設計ビューとソース・ビューを切り替えて、編集することができます。

注意点として、Admin Centerでの構成変更では、構成情報ファイル (XML) は扱うことができますが、サーバー環境ファイル (非XML) は扱うことができません。

Admin Centerによる複数の構成ファイルの変更

- 複数の構成情報ファイルの編集が可能



- Collective構成でのサーバー選択

Admin Center で構成ファイルにアクセスするとき、スタンドアロン構成、Collective構成ともに、すべての構成情報ファイルにアクセスすることが可能です。

構成変更方法の比較

□ 構成変更とは

- 構成情報ファイル(server.xml等) を変更する
…XMLファイルのため専用のエディターを利用する方が便利
- サーバー環境ファイル (server.env, jvm.options, bootstrap.property) を変更する
…XMLファイルではないので専用のエディターの必要性は低い

□ 構成変更の方法比較 (スタンドアロン/Collective構成による違いはない)

	構成情報ファイル ・ server.xml ・ インクルードファイル ・ configDropinsファイル	サーバー環境ファイル ・ bootstrap.properties ・ server.env ・ jvm.options	補足
一般のエディター	△ 間違えやすい	△～○ 間違えやすいが 専用エディターでもあまり変わらない	どこでも利用できるのがメリット
Admin Center	○	×	ローカル/リモートともに利用可能
Eclipse +WDT	○	○	ローカル/リモートともに利用可能 開発環境・テスト環境での利用が一般的

構成変更で扱うファイルは、チャートの通り、XMLである構成情報ファイル群と、非XMLであるサーバー環境ファイルの2種類に分けられます。一般論として、XMLファイルの編集には専用のエディターを利用する方が有利です。

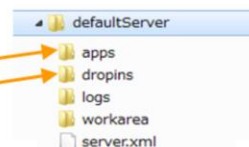
比較表では、一般のエディター、Admin Center、Eclipse+WDTの3種類を比較しています。Eclipse+WDTが全体として優れています。

アプリケーションのデプロイ

アプリケーションのデプロイ方法

□ 2通りのデプロイ方法

- appsディレクトリ下に配置しserver.xmlで指定
- dropinsディレクトリに配置し動的更新



□ 3通りの配置方法

- アーカイブ・ファイルのまま配置
 - WARファイル, EARファイルをそのままディレクトリー上に配置
- ファイルを展開して配置
 - WAR/EARファイルの名前でディレクトリーを作成し, アーカイブ・ファイルを展開
 - ファイル単位の更新が可能
- ルース・アプリケーション (※) として配置
 - アプリの配置場所を(アプリ名).xmlファイル内で指定
 - apps, dropinsディレクトリーのどちらでも利用可能
 - eclipse+WDTでも利用されている方法

例) WARファイル名が"test.war"の場合、
"test.war"というディレクトリを作成し、
その下に"test.war"ファイルを展開する

例) WARファイル名が"test.war"の場合、
"test war.xml"というファイルを作成し、
その中でWARの場所を指定する
(file/dir要素のsourceOnDisk属性で指定)

※ 詳細はリンク先 (Knowledge Center) 参照のこと

http://www.ibm.com/support/knowledgecenter/ja/SSD28V_9.0.0/com.ibm.websphere.wlp.core.doc/ae/rwlp_loose_applications.html

Libertyでのアプリケーションのデプロイ方法には2通りあります。1つ目は、appsディレクトリに配置して、server.xmlに<application>エレメントを記載していく方法です。もう1つは、dropinsディレクトリに配置することで、アプリケーションの導入を行う方法です。これら2通りについては、次ページ以降で説明します。

また、配置の方法としてはチャートの通り、3通りあります。

アプリのデプロイ方法1： appsディレクトリー

❑ server.xmlの<application>要素により指定

- location属性は必須。絶対パス・相対パスやURLで記述
- 子要素としてセキュリティ・ロールのバインディングやライブラリー参照を指定することが可能

❑ location属性を相対パスで指定した場合、

`${server.config.dir}/apps, ${shared.app.dir}`から検索される

WDTのソース・ビュー

```
<application
  id="SupportTools" location="SupportTools.war"
  name="Support Tools" type="war"
  context-root="/support" />
```

WDTの設計ビュー

Application Details

Id: SupportTools

ロケーション: SupportTools.war

名前: Support Tools

アプリケーション・タイプ: war

アプリケーション・コンテキスト・ルート: /support

☒ 自動的に開始

Application: Support Tools

- アプリケーション・バインディング
- security-role
- クラス・ローダー・サービス
- ライブラリー参照

<application>要素にネストして
<application-bnd>要素や
<classloader>要素を定義できる

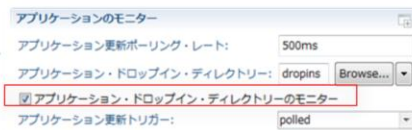
まず、appsディレクトリーに配置して、<application>エレメントを指定して導入する方法です。

アプリケーションに関して、セキュリティ属性やクラスローダーや共有ライブラリの設定などが必要な場合に利用します。また頻繁な更新が発生しない本番環境ではこちらで明示的に構成を行ってアプリケーションを導入していきます。

アプリケーションのロケーションを指定する際に、相対パスで指定した場合、サーバー構成ディレクトリー下のアプリケーション・ディレクトリーや共有アプリケーション・ディレクトリーから検索されます。

アプリのデプロイ方法2 : dropinsディレクトリ

- コード開発段階でのアプリケーションや、アプリケーション・セキュリティ・ロール等の追加構成が必要ないアプリで利用可能



- アプリケーション・モニターを構成
 - <applicationMonitor>要素で指定
 - デフォルトは\${server.config.dir}/dropins
- ドロップイン・ディレクトリーにファイルをコピーする
 - 拡張子により自動的にタイプ（EAR/WAR/WAB）を判別
 - WAR/WABは拡張子を除いたファイル名がデフォルトのコンテキストルートとなる。
例） Snoop.war は http://localhost:9080/Snoop/ でアクセス
- アプリケーションのアンインストール
 - ファイルを削除すると、自動的にアプリケーションがアンインストールされる

次にドロップイン・ディレクトリで指定する方法です。さきほど見たようなセキュリティなど詳細な設定が不要な場合は、こちらを利用できます。特に開発段階などで頻繁なアプリケーションの更新が発生する場合には、こちらの導入方法が適しています。

サーバーの構成の中で、アプリケーション・マネージャーというコンポーネントを有効にし、アプリケーションをドロップインするディレクトリを指定します。ディレクトリから削除することでアプリケーションはアンインストールされます。

Collective構成でのアプリのデプロイ方法 1

□ 方法 1 : Liberty Collective の FileTransfer MBeanを利用

Libertyクラスターに対しても、単独のメンバーに対しても利用可能

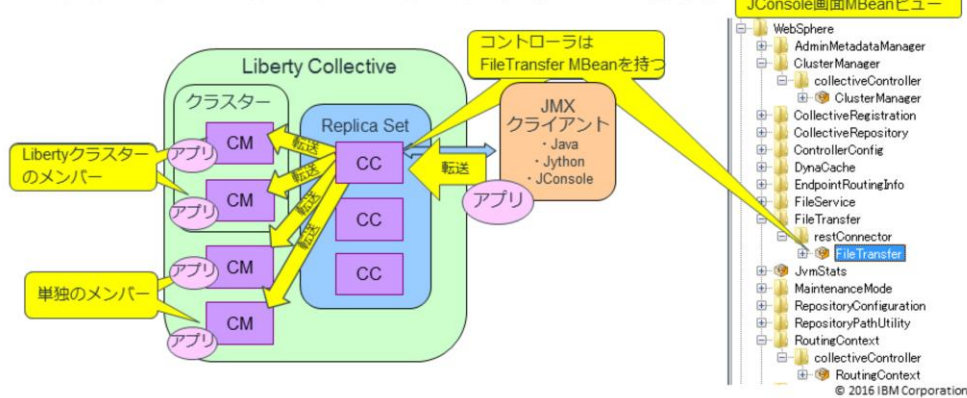
- Libertyクラスターに対するアプリ・デプロイの例 (WASDevサンプル)

https://developer.ibm.com/wasdev/downloads/#asset/scripts-jython-Transfer_an_application_to_a_cluster

※ FileTransfer MBean以外にRoutingContext MBeanとClusterManager MBeanも利用

- 単独メンバーに対するアプリ・デプロイの例 (WASDevサンプル)

https://developer.ibm.com/wasdev/downloads/#asset/scripts-jython-Transfer_an_application_to_a_member_server



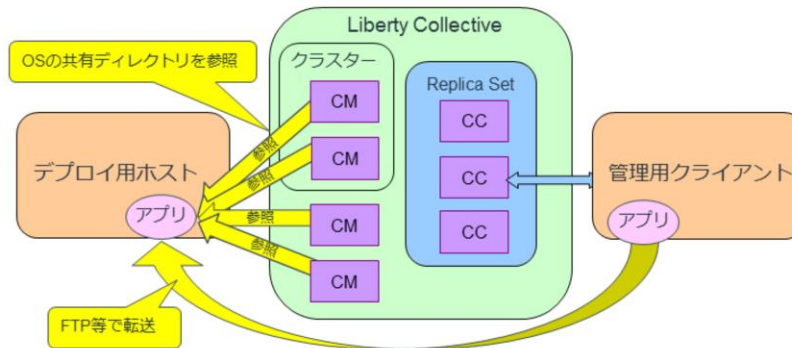
25

Libertyでのアプリのデプロイ方法としては、前ページまで見てきた通りです。これに加えて、Collective構成の場合には、さらに検討すべき要素があります。

Collectiveのコントローラーでは FileTransfer MBeanが提供されているので、これを利用してJMXクライアントからコントローラーを経由して、メンバーにアプリをデプロイすることが可能です。対象となるメンバーはクラスターのメンバーでも、単独のメンバーでも可能です。詳しい方法については、チャート中のWASDevのリンク先にあるサンプルを参照してください。

Collective構成でのアプリのデプロイ方法 2

- 方法 2 : OSの共有ディレクトリ (mount等) を利用
 - OSの共有ディレクトリ (デプロイ用ホスト) 上にアプリを配置
 - Liberty CollectiveのFileTransfer MBeanを利用しない
 - デプロイ用ホストの冗長構成が必要



26

© 2016 IBM Corporation

Collective環境では、前ページのように、コントローラーの持つFileTransfer MBeanを利用する方法がありますが、それを利用せずに、OSの共有ディレクトリーを利用する方法もあります。

デプロイ用のホスト（サーバー）を用意して、共有ディレクトリーを作成します。これをメンバーのホストからマウントして、アプリ配置用ディレクトリーとしてserver.xml内で定義します。管理用クライアントからFTP等を利用して、デプロイ用ホストの共有ディレクトリーにアプリを配置して、メンバーにアプリを参照させます。この場合、デプロイ用ホストの冗長構成についてはインフラ側で検討する必要があります。

アプリケーションから参照するライブラリの設定

□ 共有ライブラリ (library)

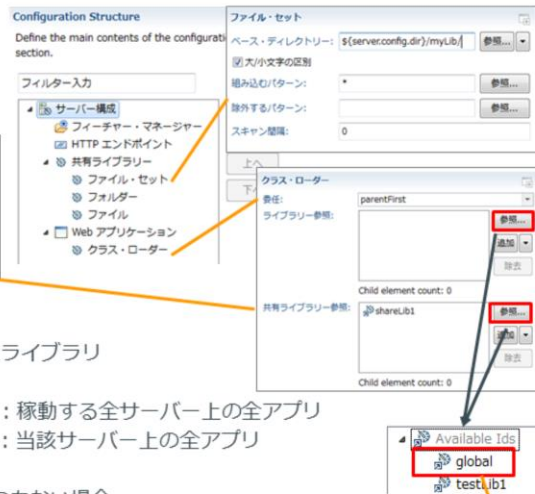
- 明示的にライブラリを構成
- アプリにクラス・ローダーを指定してライブラリを関連付ける(※)。

※ 2種類の参照(関連付け)方法

- ・ライブラリ参照 (privateLibraryRef) の場合、独立したクラスローダーは作成されず、アプリのクラスローダーによってロードされる。
- ・共有ライブラリ参照 (commonLibraryRef) の場合、独立したクラスローダーが作成される。

□ グローバル・ライブラリ

- 全てのアプリケーションから使用可能なライブラリ
- 以下の2箇所に配置可能
 - `${shared.config.dir}/lib/global` : 稼動する全サーバー上の全アプリ
 - `${server.config.dir}/lib/global` : 当該サーバー上の全アプリ
- 以下の場合にライブラリを参照
 - アプリにクラス・ローダー定義が1つも無い場合
 - アプリにクラス・ローダー定義が存在し、グローバル・ライブラリを関連付けている場合
(デフォルトでライブラリID "global"が定義されているので、参照ボタンで選択可能)
- BestPractice: クラスローダーの分離と依存性明確化のため、明示的にクラスローダー定義を構成



アプリケーションが参照するライブラリの管理についてご説明します。

まず共有ライブラリです。明示的にライブラリとして構成し、それを共有ライブラリとしてクラスローダーの管理画面から関連付けます。関連付ける方法(「参照」と呼ばれます)には、チャート内の枠内の通り、2種類の方法があります。

もう一つの定義方法が、グローバル・ライブラリです。これは全てのアプリケーションから使用可能となるライブラリであり、スコープにより2つの配置場所があります。稼動する全サーバー上のアプリで共有する場合は、共有構成ディレクトリ下の `${shared.config.dir}/lib/global` に配置します。サーバー・インスタンス内で稼動するアプリケーション内で共有する場合には `${server.config.dir}/lib/global` に配置します。

グローバル・ライブラリが利用されるための条件は、チャート内の説明の通りです。2条件のうち、後者の方法が推奨されています。アプリとライブラリとの関係が明確になると、参照が明確に定義できるので、クラスローダーの分離も容易なためです。なお、参照するグローバル・ライブラリは事前定義済みのライブラリIDである「global」を指定します。

アプリのデプロイ配置方法とライブラリー設定まとめ

□ アプリのデプロイ・配置方法の比較

デプロイ 配置	appsディレクトリ +server.xml指定	dropinsディレクトリ	補足
WAR/WAB, EARのまま 配置	本番環境向き 大規模複雑アプリ向き	開発テスト環境向き 小規模単純アプリ向き	ファイルを展開しないので、アプリが ServletContext.getRealPath()に依存 しないことを確認すること
展開して 配置	本番環境向き 大規模複雑アプリ向き	開発テスト環境向き 小規模単純アプリ向き	アプリの一部ファイル置き換えが可能
ルース・ア プリ方式	アプリを他の場所に置きた い特殊要件向き	アプリを他の場所に置きた い特殊要件向き	Eclipse+WDTで利用

□ Collective構成でのデプロイ（追加検討項目）

- 方法1：FileTransfer MBeanを使う方法
 - メンバーのホスト上にアプリを転送可能→メンバー・ホストの障害に強い
- 方法2：OSの共有ディレクトリ上を参照する方法
 - デプロイ用ホストの冗長構成はインフラ側で検討→ホストの障害に弱い

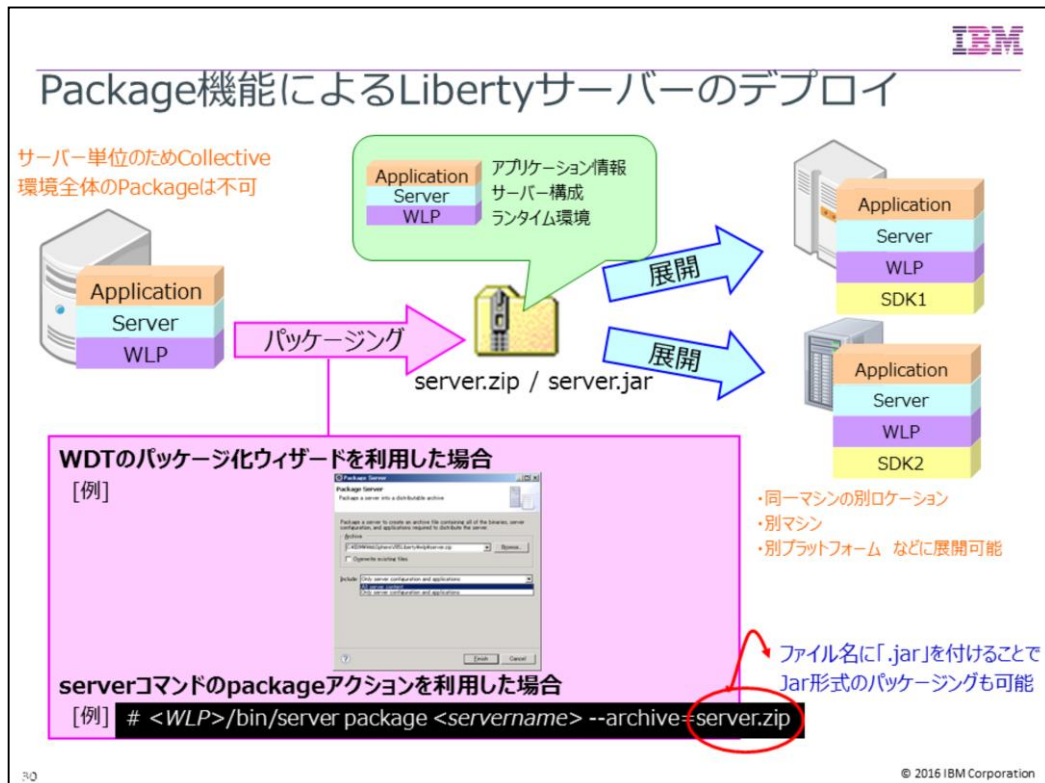
□ ライブラリー設定

- ライブラリーは共有ライブラリとグローバル・ライブラリの2種類
 - グローバル・ライブラリーでは明示的にアプリと関連付けることが推奨

このチャートでは、アプリのデプロイ配置方法とライブラリー設定をまとめています。

表の通り、デプロイ方法については、appsディレクトリ+server.xmlの方法が、本番環境には適していると言えます。また、この表の補足カラムの通り、アーカイブ・ファイルのまま配置する方法は、ServletContext.getRealPath()のメソッドをアプリが利用していると、ファイル・システム上の絶対パスを返すことができないため、Nullが返ります。アプリでこのメソッドを利用していないこと、このメソッドに依存するコンポーネントを利用していないかに注意してください。

Libertyサーバーのデプロイ



ここまでは、Libertyサーバー上にアプリをデプロイする方法を見てきました。ここでは、Libertyサーバー自体をまとめてデプロイする方法について説明します。

Package機能によるデプロイとは、一度構築したランタイム環境、サーバー構成、アプリケーション情報を全て1つのzip形式の圧縮ファイルにパッケージングし、それをzip展開することによってLibertyプロファイルを導入するという方法です。

パッケージングを行う機能はLibertyプロファイルが提供しており、WDTのパッケージ化ウィザードを利用したGUIベースで実行することも、serverコマンドのpackageアクションを利用したコマンドベースで実行することもできます。

zip展開先としては、パッケージングを行った同一マシンの別ロケーションでも、別マシン上でも、別プラットフォームのマシン上でも可能です。別プラットフォームのマシンにzip展開する場合、ディレクトリ名やファイル名の大文字小文字区別の可否などに関しては考慮する必要があります。

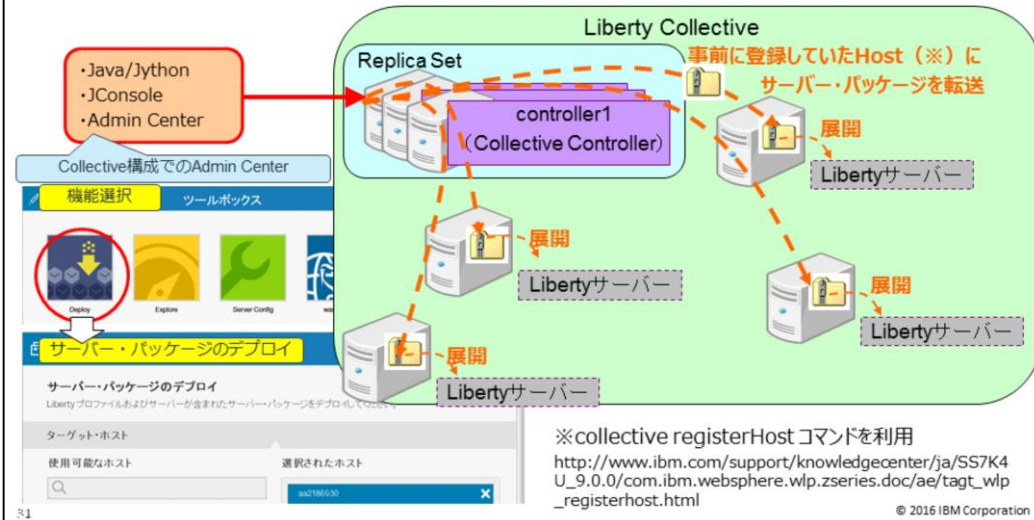
V8.5では、Package機能によりパッケージングされた後のファイルはzip形式の圧縮ファイルでしたが、V8.5.5の新機能として、実行時にファイル名に「.jar」を付けることにより、Jar形式の圧縮ファイルにすることもできるようになりました。

なお、Java SDKについてはLibertyパッケージに含めることはできないため、展開先のホストに別途用意されていることが前提となります。

Liberty Collective のPushOut構築モデル

PushOut構築

Liberty Collectiveで提供されるファイル転送機能（MBeanとして提供）を使用して
Collectiveに登録されたLibertyの存在しないHostに対してSSH接続してLibertyサーバーを導入



前ページでは、Libertyサーバーをパッケージして、Libertyをデプロイする方法を説明しました。この方法をCollective環境で利用する場合について説明します。

Collective構成において、メンバーのLibertyサーバーをデプロイするのに、Libertyパッケージを利用することができます。ここでのメンバーは、まだLibertyがインストールされていないホストを指します。そのためには、事前に「collective registerHostコマンド」を利用して、Collective環境にホストを定義しておきます。登録する内容としては、ホストのOSユーザー情報や証明書、SSH関連情報です。

この方法は、Liberty Collective のPushOut構築モデルと呼ばれます。パッケージ・ファイルの転送には、アプリのデプロイにも利用されるFileTransfer MBeanが利用されます。

WASdevサイト(英語)ではMBeanを利用したサンプルを紹介・提供しています。

Deploy Collective Members

https://developer.ibm.com/wasdev/downloads/#asset/scripts-jython-Deploy_Collective_Members

Using File Service and File Transfer MBeans with Liberty

<https://developer.ibm.com/wasdev/docs/using-file-service-and-file-transfer-mbeans-with-liberty/>

バックアップ・リストア、プロセス監視

Libertyサーバーのバックアップ・リストア

- スタンドアロン構成／Collective構成ともに
 - (方法 1 : 丸ごとバックアップ)
 - 構成変更またはアプリ更新のタイミングでLibertyインストール・ディレクトリーを丸ごとバックアップする
 - (方法 2 : Libertyサーバーごとにバックアップ)
 - Libertyサーバー個別のバックアップについては、server packageコマンドを利用することも。
(注意点)
wlp/etc 以下に共通のサーバー環境ファイルを置いた場合は、server packageコマンドで --include=usr オプションでは取れない。--include=all オプションが必要。
ただし、--include=all ではバイナリーを含むパッケージになるためサイズ削減の効果は薄い。
 - IIMを利用してインストールした場合は
 - Fix, FixPack適用のタイミングで以下3ディレクトリーをバックアップ
 - IBM Installation Managerディレクトリ
 - 共有リソース・ディレクトリ
 - エージェント・データ・ディレクトリ
- ※詳細については以下リンク先(IIM Knowledge Center)参照のこと
http://www.ibm.com/support/knowledgecenter/ja/SSDV2W_1.8.5/com.ibm.cic.agent.ui.doc/topics/t_im_backup.html

Libertyサーバーのバックアップ・リストア方法には大きく2通りの方法があります。

1つ目は、Libertyインストール・ディレクトリーの下を丸ごとtarファイル等でバックアップを取る方法です。バックアップ対象の抜けがなく、簡単な方法ですが、ファイル・サイズが大きくなるのがデメリットです。

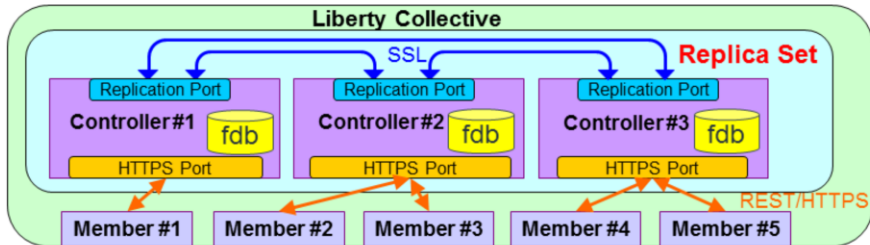
2つ目は、Libertyのサーバーごとにバックアップを取る方法です。server package コマンドを利用することができますが、usr/servers/<server_name>の外に関連ファイルがある場合、たとえば、wlp/etcにサーバー環境ファイルを置いた場合など、--include=allオプションが必要で、1つ目の方法と比べてサイズ削減の効果は薄いかもしれません。

IIMを利用してインストールした場合は、IIM関連の3つのディレクトリーもバックアップを取る必要があります。

Liberty Collective構成のバックアップ・リストア

□ Replica Setのコントローラーの世代は揃えてバックアップ

- すべてのコントローラーはリポジトリ（※）を同期させている
- （※）リポジトリはFrappe Database(fdb)とも呼ばれ、構成と状況のデータを持つファイルの場所はusr/servers/<CCname>/resources/collective/repository/fdb



□ fdbデータが破損したときや同期が取れていないときのリストア

- マスターとなるfdbだけ残して他のfdbはすべて削除する
- fdbを持つコントローラーを先に起動して残りを1個ずつ起動していく
- ※詳細については以下リンク先(Knowledge Center)参照のこと
http://www.ibm.com/support/knowledgecenter/ja/SSAW57_9.0.0/com.ibm.websphere.wlp.nd.doc/ae/tagt_wlp_collective_backup_restore.html

Collective構成でのバックアップ・リストアの注意点です。

Collectiveの複数のコントローラーはReplica Setとして一種のクラスター構成を取っており、各コントローラーが持つリポジトリ（Frappe Databaseとも呼ばれるため、略称としてfdbが使われる）を持ちます。リポジトリは構成情報と状況のデータを持ちます。そのパスは、チャート中の図の上にある通りです。

リポジトリは常に同期を取っているため、この同期が取れていないコントローラーをReplica Setに混ぜてリストアしようとする、正しく起動できないことがあります。そのため、構成変更やアプリやメンバーの状況が同一な状態で、全コントローラーのバックアップを取得して、リストアするときも、同じ世代のコントローラーでリストアします。

もし、リポジトリの同期が取れていないことが判明した場合には、チャート中の一番下のリンク先の通りの手順で復活させるようにします。

プロセス稼動状況の監視

□ 監視プロセスと監視方法

コンポーネント	IHS	Libertyプロセス
監視プロセス	httpd	java
監視方法	①psコマンド (Windows: tasklistコマンド)	①psコマンド (Windows: tasklistコマンド) ②server statusコマンド ③Collective構成の場合はコントローラーのログ
備考	Windows系で tasklistコマンドを使用できないバージョンの場合は、netstatコマンドで、Listenしているポートの確認などで代替(Libertyプロセスの場合も同様)	Collective構成でAutoScale(自動スケーリング)を利用している場合は最低稼動数を満たすように自動的にプロセスの起動を行い、ログに出力される

※ (自動スケーリングによるプロセス起動のログ出力例)

CWWKV0111I: スケーリング・コントローラーは、クラスター defaultCluster の最小インスタンスを満たすために、ホスト was02 上のユーザー・ディレクトリー /opt/IBM/Liberty2/wlp/usr のサーバー Mem3 を開始しています。
CWWKV0112I: スケーリング・コントローラーは、ホスト was02 上のサーバー Mem3 を正常に開始しました。

server statusコマンドは別途javaプロセスが起動して結果を出力→負荷が比較的高いので注意

35

© 2016 IBM Corporation

上記は、一般的なIHS/WASの監視対象プロセスおよび監視方法をまとめたものです。IHSはhttpdプロセス（親プロセスのみ）、Libertyはjavaプロセスの監視を行います。一般的には、OSのpsコマンドを使用して、プロセス名やアプリケーション・サーバー名をキーにして、出力を確認します。Libertyが提供するコマンド行ツールのserver statusコマンドでもプロセスの稼動状況を確認することができますが、このsコマンドを実行した場合には、別途javaプロセスが起動し、psコマンドの実行と比較して負荷が高くなりますので、ご注意ください。また、一部Windowsのバージョンでは、tasklistコマンドが使用できませんので、netstatコマンドで、Listenしているポート番号（IHSであれば通常80番、Libertyのデフォルトは9080番）を監視することなどで代替できます。

Collective構成で自動スケーリングを利用している場合には、プロセスの数が最低稼動数を満たすように自動的にLibertyメンバーが起動されます。その場合には、チャート中の噴出し部分のようなメッセージがログに出力されます。このメッセージIDをモニターするとプロセス監視に役立つ場合があります。

まとめ

- Admin Center
 - 機能が拡張されて更に便利に
- 起動停止・状況確認
 - コマンド、Jconsole、Admin Center、WDTから要件に応じて選択
- 構成変更
 - 一般のエディター、Admin Center、WDTから要件に応じて選択
- アプリケーションのデプロイ
 - デプロイ方法としてdropins、appsの2種類がある
 - 配置方法としてアーカイブファイル、展開、ルース・アプリの3種類がある
 - Collective環境ではFileTransfer MBeanが利用可能
- Libertyサーバーのデプロイ
 - server packageコマンドが利用可能
 - Collective環境ではFileTransfer MBeanによりPushOut構築が利用可能
- バックアップ・リストア、プロセス監視
 - バックアップ・リストアはインストール単位またはサーバー単位で
 - プロセス監視はpsコマンドで実施可能

このセッションの全体のまとめです。

ワークショップ、セッション、および資料は、IBMまたはセッション発表者によって準備され、それぞれ独自の見解を反映したものです。それらは情報提供の目的のみで提供されており、いかなる参加者に対しても法的またはその他の指導や助言を意図したものではなく、またそのような結果を生むものでもありません。本講演資料に含まれている情報については、完全性と正確性を期するよう努力しましたが、「現状のまま」提供され、明示または暗示にかかわらずいかなる保証も伴わないものとします。本講演資料またはその他の資料の使用によって、あるいはその他の関連によって、いかなる損害が生じた場合も、IBMは責任を負わないものとします。本講演資料に含まれている内容は、IBMまたはそのサプライヤーやライセンス交付者からいかなる保証または表明を引き出すことを意図したものでも、IBMソフトウェアの使用を規定する適用ライセンス契約の条項を変更することを意図したものでもなく、またそのような結果を生むものでもありません。

本講演資料でIBM製品、プログラム、またはサービスに言及していても、IBMが営業活動を行っているすべての国でそれらが使用可能であることを暗示するものではありません。本講演資料で言及している製品リリース日付や製品機能は、市場機会またはその他の要因に基づいてIBM独自の決定権をもっていつでも変更できるものとし、いかなる方法においても将来の製品または機能が使用可能になると確約することを意図したものでもありません。本講演資料に含まれている内容は、参加者が開始する活動によって特定の販売、売上高の向上、またはその他の結果が生じると述べる、または暗示することを意図したものでも、またそのような結果を生むものでもありません。パフォーマンスは、管理された環境において標準的なIBMベンチマークを使用した測定と予測に基づいています。ユーザーが経験する実際のスループットやパフォーマンスは、ユーザーのジョブ・ストリームにおけるマルチプログラミングの量、入出力構成、ストレージ構成、および処理されるワークロードなどの考慮事項を含む、数多くの要因に応じて変化します。したがって、個々のユーザーがここで述べられているものと同様の結果を得られると確約するものではありません。

記述されているすべてのお客様事例は、それらのお客様がどのようにIBM製品を使用したか、またそれらのお客様が達成した結果の実例として示されたものです。実際の環境コストおよびパフォーマンス特性は、お客様ごとに異なる場合があります。

IBM、IBM ロゴ、ibm.com、WebSphere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、www.ibm.com/legal/copytrade.shtmlをご覧ください。

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

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。



*WebSphere
Application Server
Liberty*