WAS Liberty 最新情報セミナー 2015

～ 次世代ランタイム Libertyで変わる Java EE システム～

4. Libertyプロファイルの運用管理

© 2015 IBM Corporation

1

Disclaimer

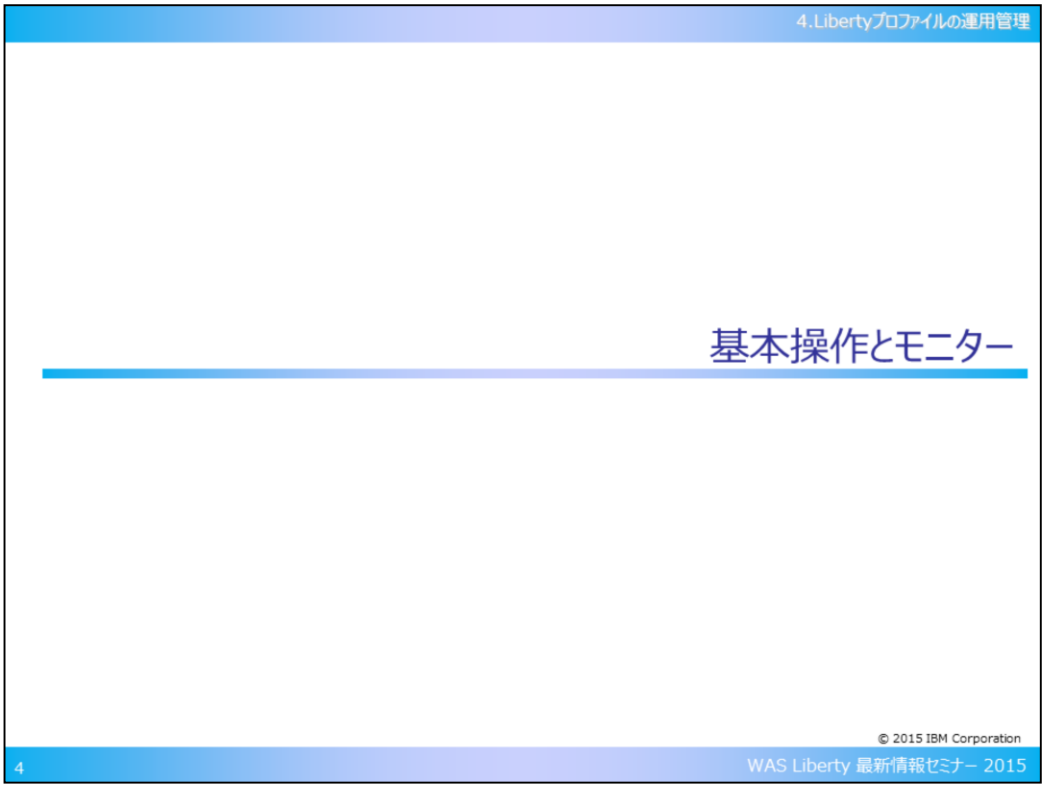
- この資料は日本アイ・ビー・エム株式会社ならびに日本アイ・ビー・エム システムズ・エンジニアリング株式会社の正式なレビューを受けておりません。
- 当資料は、資料内で説明されている製品の仕様を保証するものではありません。
- 資料の内容には正確を期するよう注意しておりますが、この資料の内容は2015年7月現在の情報であり、製品の新しいリリース、PTFなどによって動作、仕様が変わる可能性があるのでご注意下さい。
- 今後国内で提供されるリリース情報は、対応する発表レターなどでご確認ください。
- ＩＢＭ、ＩＢＭロゴおよびibm.comは、世界の多くの国で登録されたInternational Business 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、www.ibm.com/legal/copytrade.shtmlをご覧ください。
- 当資料をコピー等で複製することは、日本アイ・ビー・エム株式会社ならびに日本アイ・ビー・エム システムズ・エンジニアリング株式会社の承諾なしではできません。
- 当資料に記載された製品名または会社名はそれぞれの各社の商標または登録商標です。
- 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。
- Microsoft, Windows 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。
- 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。
- 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。

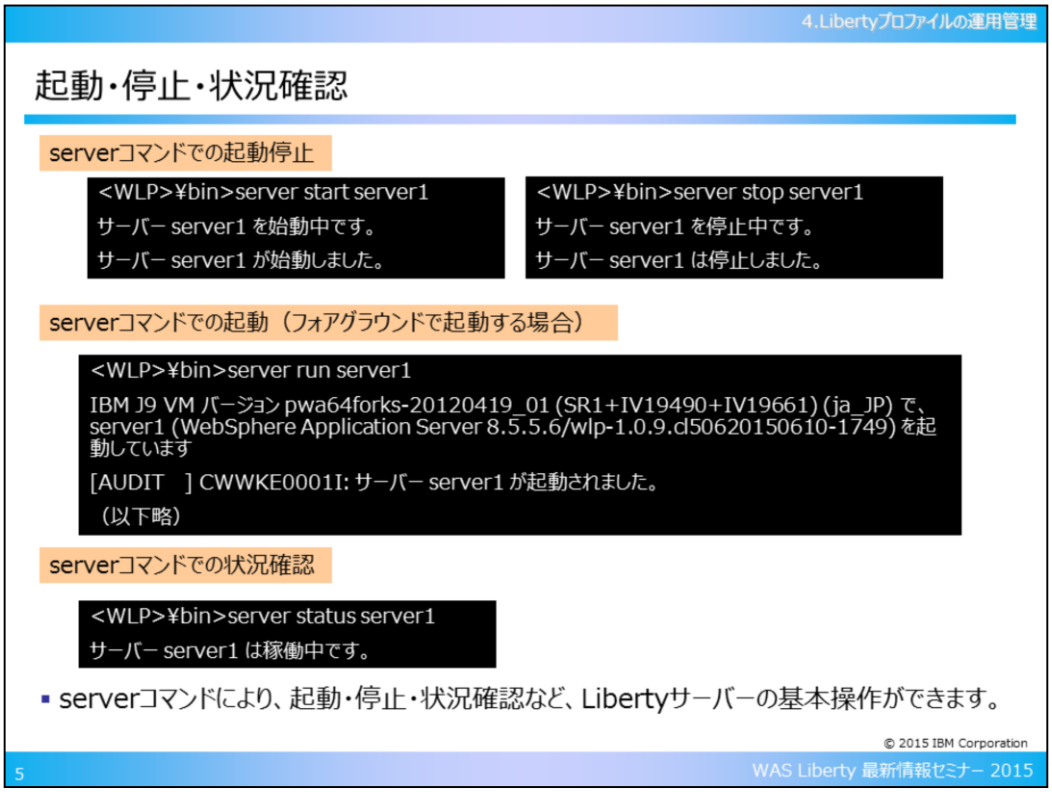
© 2015 IBM Corporation

目次

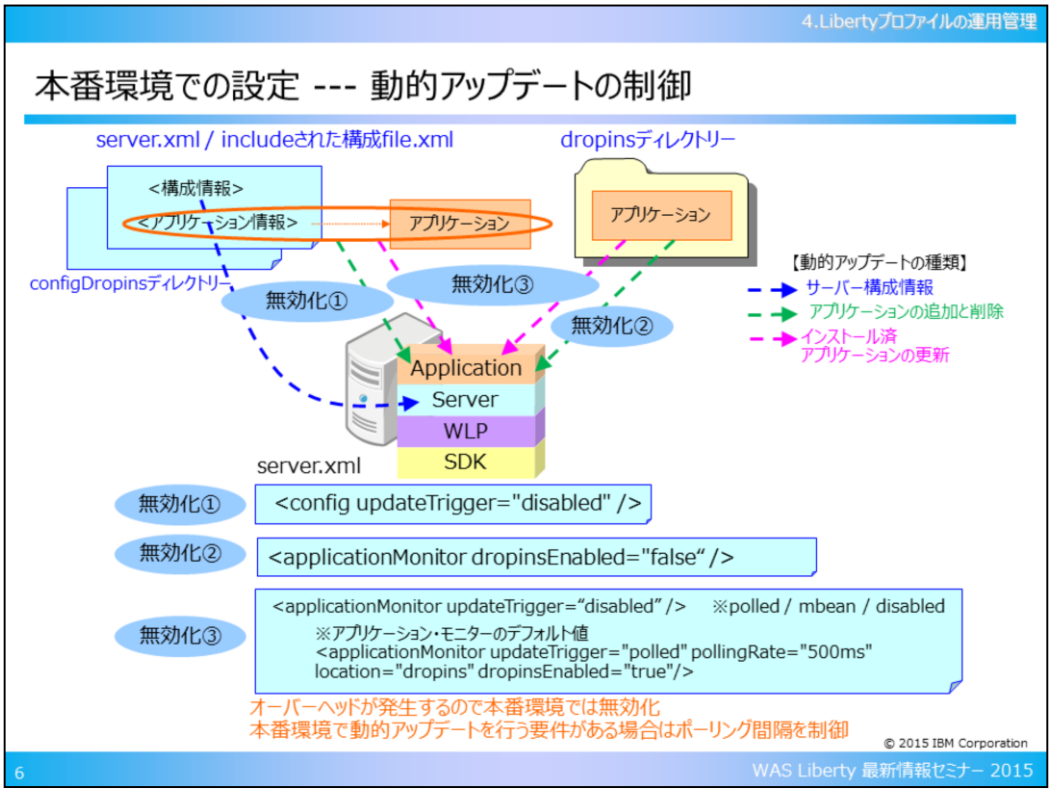
- 基本操作とモニター
  - 起動・停止・状況確認
  - 本番環境での設定
  - モニター方法
  - AdminCenter
- ログと問題判別
  - ログ・ファイル
  - GCログ
  - Dump
  - 診断関連フィーチャー
- DevOps
  - Package機能によるデプロイ
  - DevOpsツール
- まとめ

© 2015 IBM Corporation


4.Libertyプロファイルの運用管理
基本操作とモニター
© 2015 IBM Corporation
4
WAS Liberty 最新情報セミナー 2015



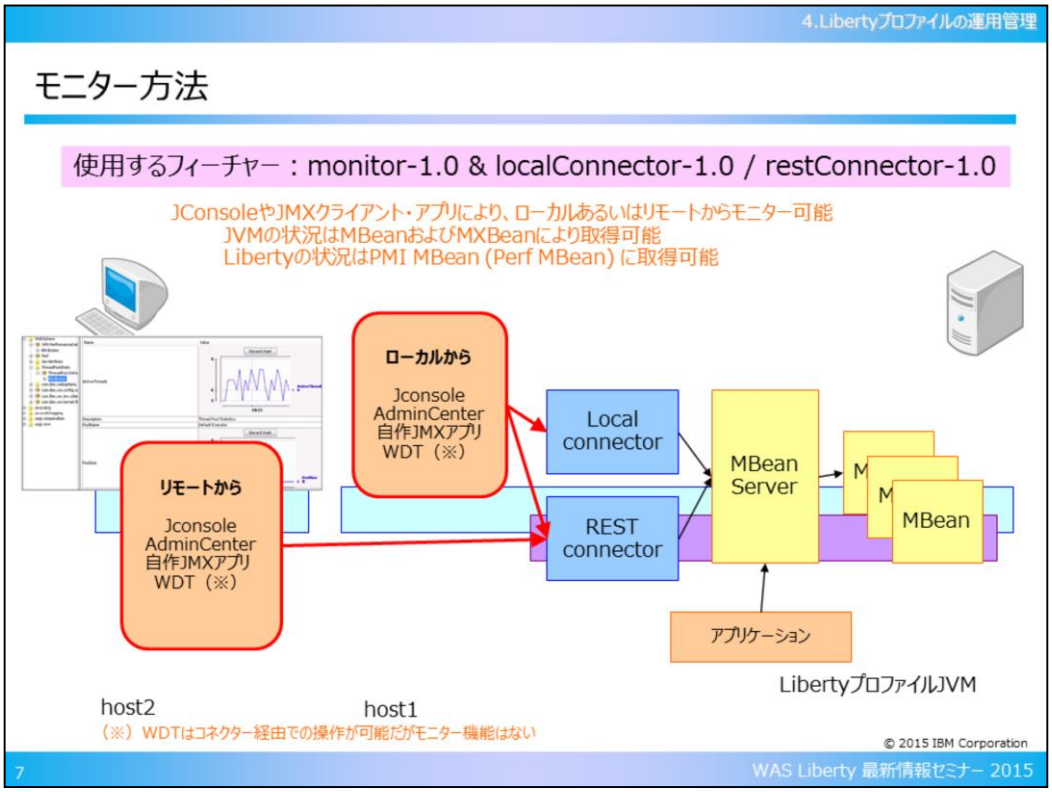
通常のLibertyサーバーは、serverコマンドのstartアクション・stopアクションを実行して起動・停止を行うことができます。起動には、startアクションに加えて、runアクションを利用することもできます。runアクションではフォアグラウンドでLibertyが起動されるため、稼働中はプロンプトは戻りません。また、Libertyサーバーの状況確認には、statusアクションを利用します。



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。



Libertyプロファイルのサーバー・ランタイム環境をモニターするためにはmonitor-1.0フィーチャーを使用します。そして、ローカルツールからモニター・データを確認するためにはlocalConnector-1.0フィーチャー、リモートツールからモニター・データを確認するためにはrestConnector-1.0フィーチャーを使用します。

monitor-1.0フィーチャーを有効にするとモニターが開始され、モニター・データは標準MBeanおよびMXBeanとしてレポートされます。モニター用のMXBeanとして、「WebSphere:type=JvmStats」（JVM）、「WebSphere:type=ServletStats,name=*」（スレッドプール）、「WebSphere:type=ThreadPoolStats,name=DefaultExecutor」（Webアプリケーション）などがあります。また、従来のPMI MBean（Perf MBean）も使用できますので、必要に応じてモニター・データを確認することができます。

localConnector-1.0フィーチャーでは、ローカルJMXコネクターが提供されます。このコネクターはJVMに組み込まれ、同一ホスト・マシン上で同一ユーザー ID、同一JDKで実行しているユーザーのみが使用することができます。Jconsoleなどの JMXクライアントやAttach API を使用するその他のJMX クライアント、RAD/WDTなどによるローカル・アクセスが可能になります。

restConnector-1.0フィーチャーでは、セキュアなJMXコネクターが提供されます。このコネクターは、任意のJDKを使用してローカルまたはリモートで使用することができます。RESTベース・コネクター経由のJMX クライアントによるリモート・アクセスが可能になります。SSLおよび基本ユーザー・セキュリティー構成が必要になります。

【参考】モニター・データ（1/2）

JVM

- Heap: 現行JVMに使用されているヒープ・サイズ
- FreeMemory: 現行JVMに使用可能な空きヒープ
- UsedHeap: 現行JVMの使用済みヒープ
- ProcessCPU: JVMプロセスで使用されたCPUのパーセンテージ
- GcCount: JVMの始動以降にGCが行われた回数
- GcTime: GC時間の合計累算値
- UpTime: JVM の始動以降の時間(ミリ秒単位)

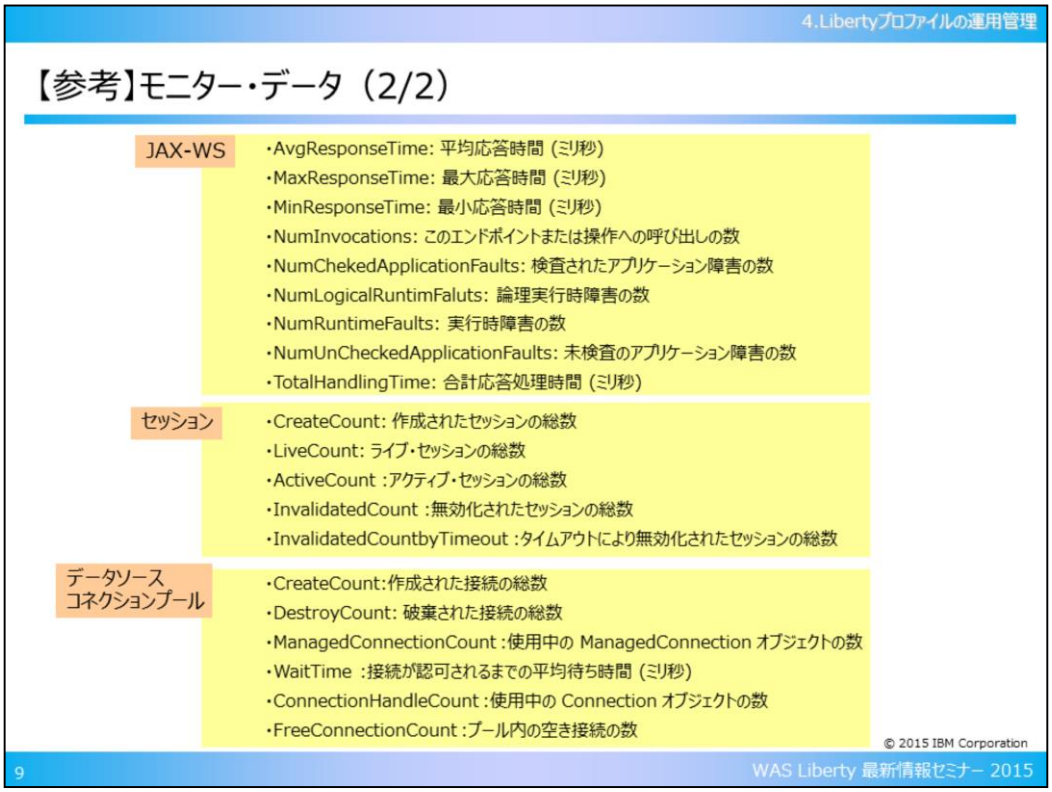
スレッドプール

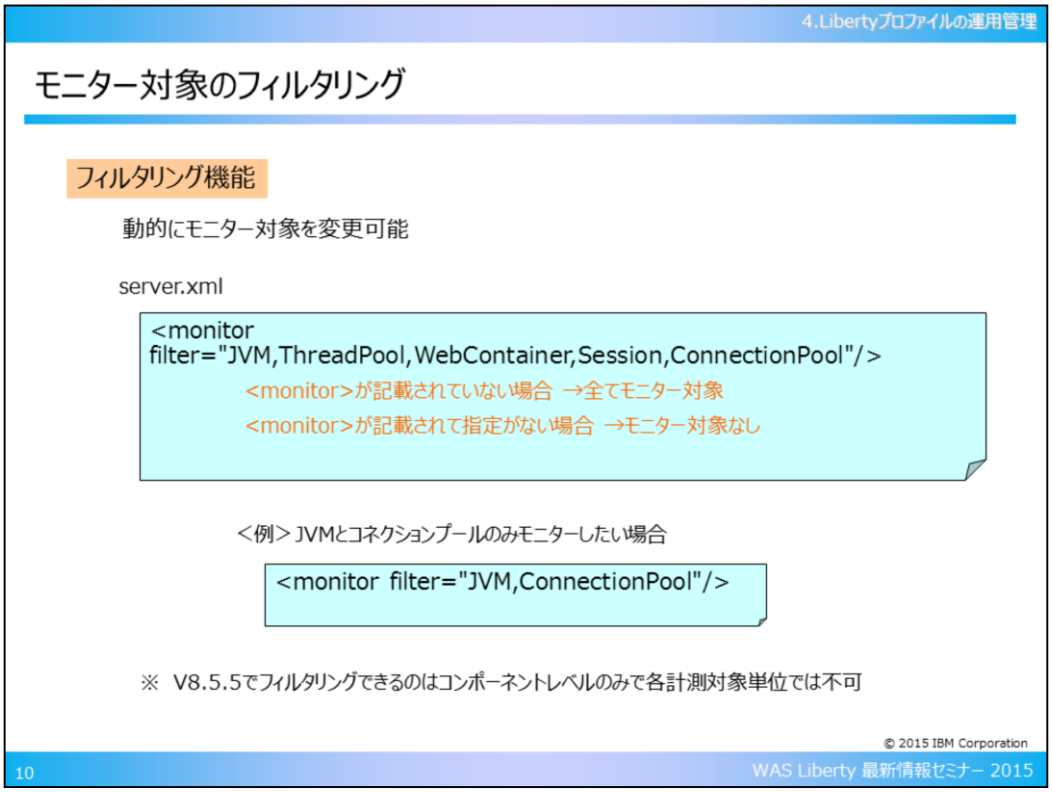
- PoolName: スレッドプール名
- AcitiveThreads: 要求を処理中のアクティブ・スレッド数
- PoolSize: スレッドプールのサイズ

Webアプリケーション

- AppName: アプリケーションの名前
- ServletName: サーブレットの名前
- RequestCount: このサーブレットに対するヒット数
- ResponseTime: 平均応答時間 (ナノ秒)
- Description: カウンターの説明
- RequestCountDetails: 最終タイム・スタンプなど、RequestCount の詳細
- ResponseTimeDetails: 取得したスナップショットの数、最小値、最大値など、ResponseTime の詳細

© 2015 IBM Corporation

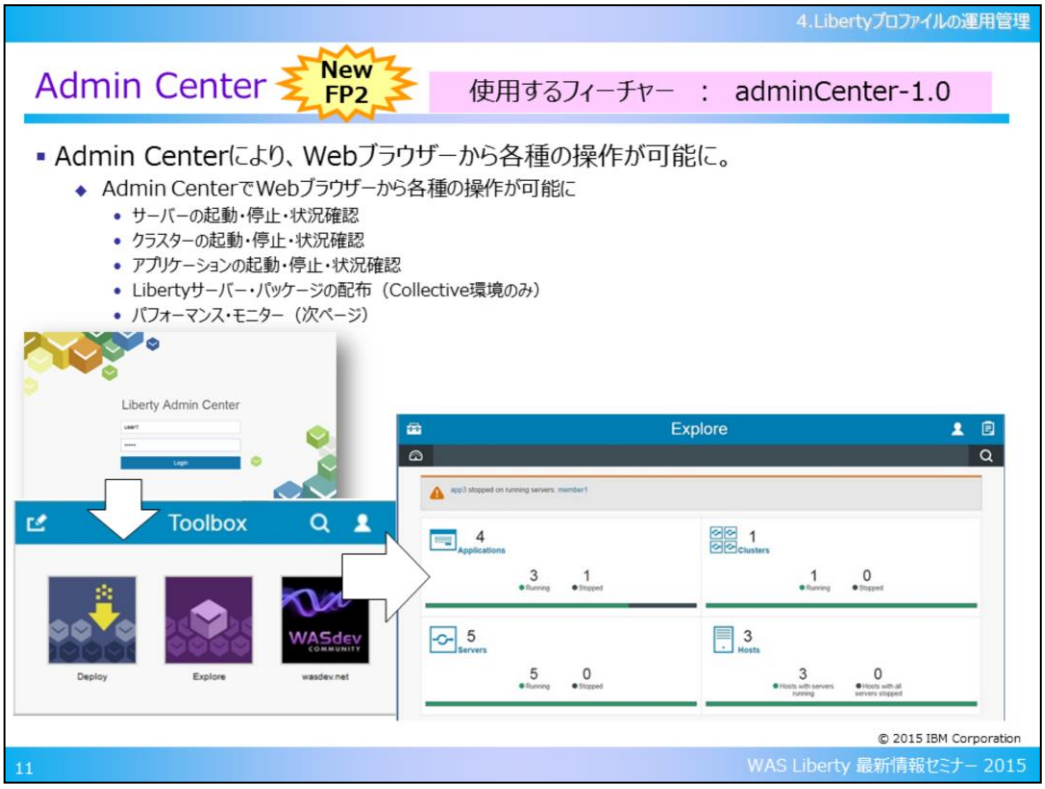

4.Libertyプロファイルの運用管理
【参考】モニター・データ（2/2）
JAX-WS
・AvgResponseTime: 平均応答時間 (ミリ秒)
・MaxResponseTime: 最大応答時間 (ミリ秒)
・MinResponseTime: 最小応答時間 (ミリ秒)
・NumInvocations: このエンドポイントまたは操作への呼び出しの数
・NumChekedApplicationFaults: 検査されたアプリケーション障害の数
・NumLogicalRuntimFaluts: 論理実行時障害の数
・NumRuntimeFaults: 実行時障害の数
・NumUnCheckedApplicationFaults: 未検査のアプリケーション障害の数
・TotalHandlingTime: 合計応答処理時間 (ミリ秒)
セッション
・CreateCount: 作成されたセッションの総数
・LiveCount: ライブ・セッションの総数
・ActiveCount :アクティブ・セッションの総数
・InvalidatedCount :無効化されたセッションの総数
・InvalidatedCountbyTimeout :タイムアウトにより無効化されたセッションの総数
データソース
コネクションプール
・CreateCount:作成された接続の総数
・DestroyCount: 破棄された接続の総数
・ManagedConnectionCount :使用中の ManagedConnection オブジェクトの数
・WaitTime :接続が認可されるまでの平均待ち時間 (ミリ秒)
・ConnectionHandleCount :使用中の Connection オブジェクトの数
・FreeConnectionCount :プール内の空き接続の数
© 2015 IBM Corporation
9
WAS Liberty 最新情報セミナー 2015



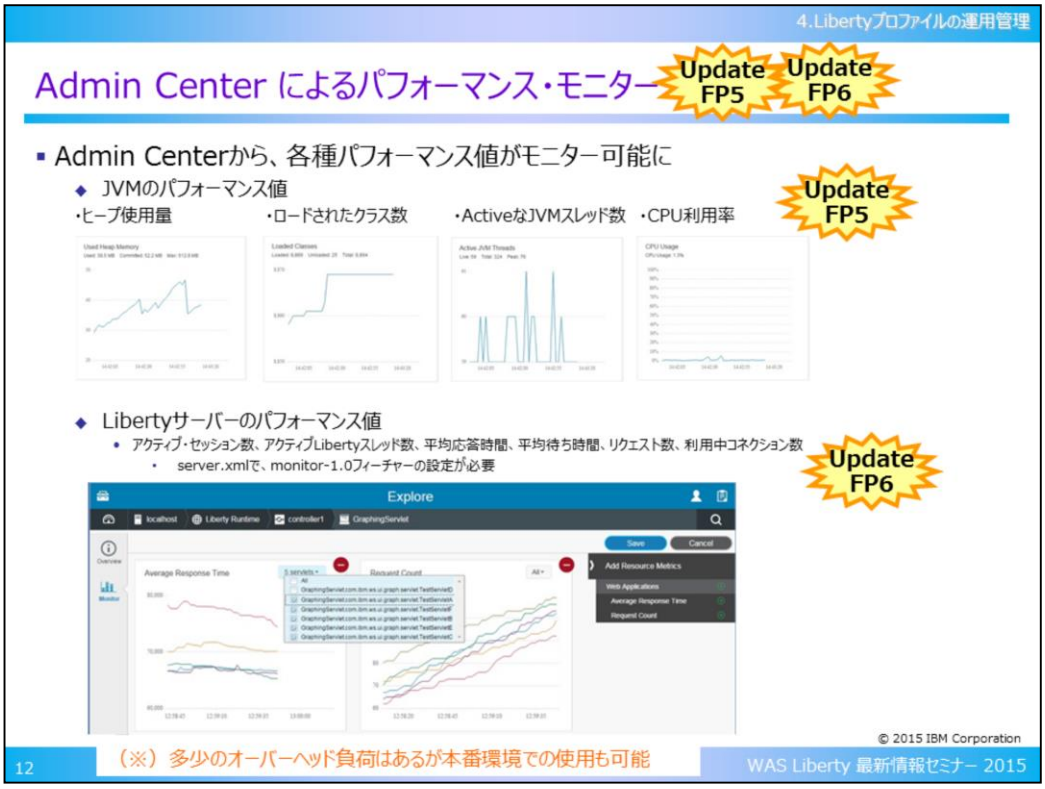
V8.5.5の新機能として、Libertyプロファイルのモニターにフィルタリングができるようになりました。

モニター対象は、server.xmlに<monitor filter>エレメントを記述することで動的に変更することができます。特に何の記述もない場合、monitor-1.0フィーチャーを有効にした時点で全ての項目のモニターが行われます。<monitor filter>エレメントの中に項目を記述すると、その項目のみがモニターされるようになります。また、<monitor filter>エレメントを記述して中をブランクにすると、どの項目もモニターされないようになります。

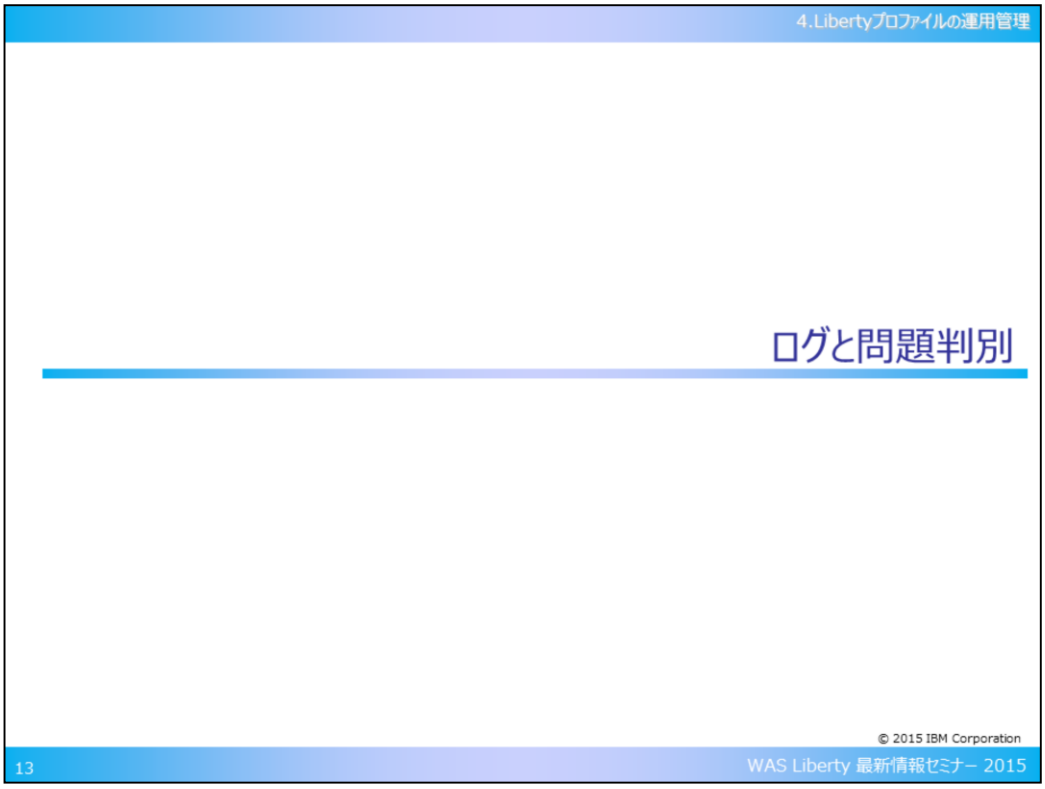
V8.5.5のフィルタリング機能では、コンポーネントレベルでの設定のみであり、各計測対象単位（参考ページ記載の各項目）でのフィルタリングすることはできません。

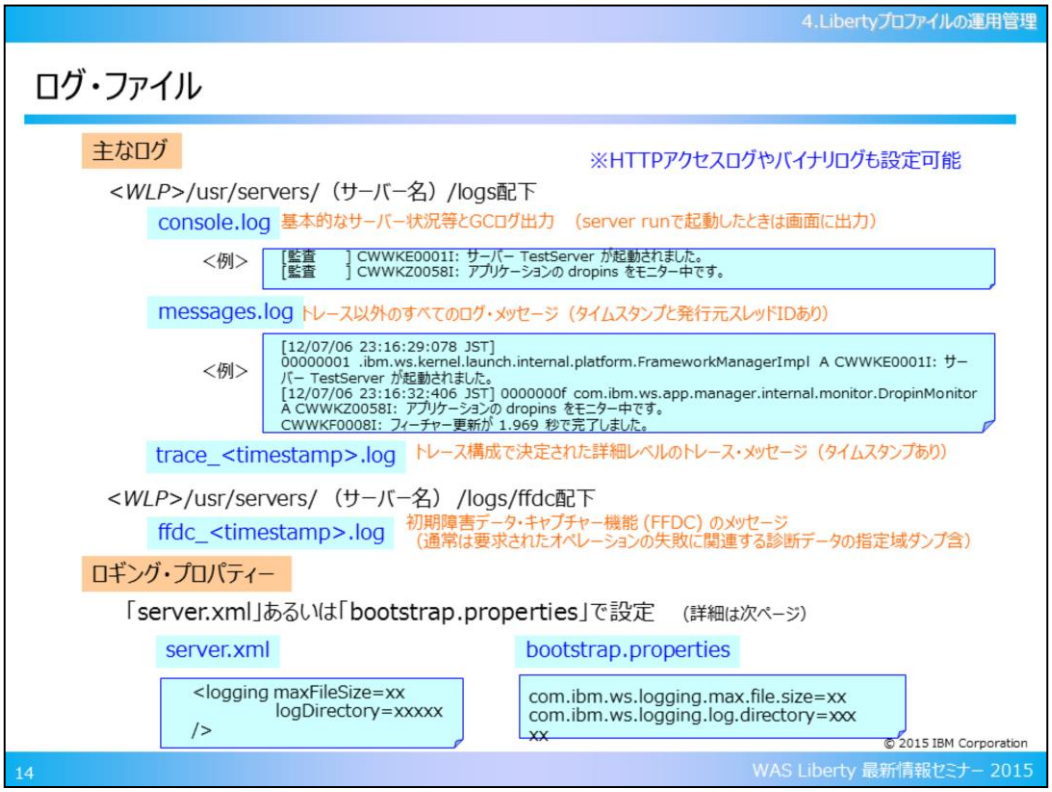


FixPack 2からAdmin Center機能が追加されました。Webブラウザーから各種の操作が可能になっています。サーバー、クラスター、アプリケーションの起動・停止・状況確認です。これ以外にも、Libertyサーバー・パッケージの配布（Collective環境のみ）やパフォーマンス・モニターも行うことができます。パフォーマンス・モニターについては、次ページで説明します。使用するフィーチャーは、adminCenter-1.0です。



Admin Centerは、FixPack 5からの新機能として、JVMのパフォーマンス値のリアルタイム・モニターが可能になっています。さらに、FixPack 6からの新機能として、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）のリアルタイム・モニターも可能になっています。このためには、server.xmlでmonitor-1.0フィーチャーを有効にする必要があります。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。


4.Libertyプロファイルの運用管理
ログと問題判別
© 2015 IBM Corporation
13
WAS Liberty 最新情報セミナー 2015



Libertyプロファイルの主なログは、console.log、messages.log、trace_<timestamp>.log、ffdc_<timestamp>.logなどがあり、デフォルトではLibertyプロファイルの各サーバーディレクトリ配下のlogsディレクトリに出力されます。

console.logには、JVMプロセスからリダイレクトされた基本的なサーバー状況およびオペレーション・メッセージが出力されます。タイムスタンプは出力されません。messages.logには、System.outやSystem.errおよびその他のメッセージが出力されます。タイムスタンプと発行元スレッドIDが出力されますので、障害時の問題判別の際などは、こちらのログを確認します。

trace_<timestamp>.logには、現行トレース構成で設定された詳細レベルのトレースやメッセージが出力されます。タイムスタンプは出力されます。

ffdc_<timestamp>.logには、初期障害データ・キャプチャー機能（FFDC）のメッセージが出力されます。通常は、要求されたオペレーションの失敗に関連する診断データの指定域ダンプが含まれます。

ログファイルの出力先や最大サイズや出力内容などの詳細は、ロギング・プロパティーとしてserver.xmlあるいはbootstrap.propertiesで設定することができます。

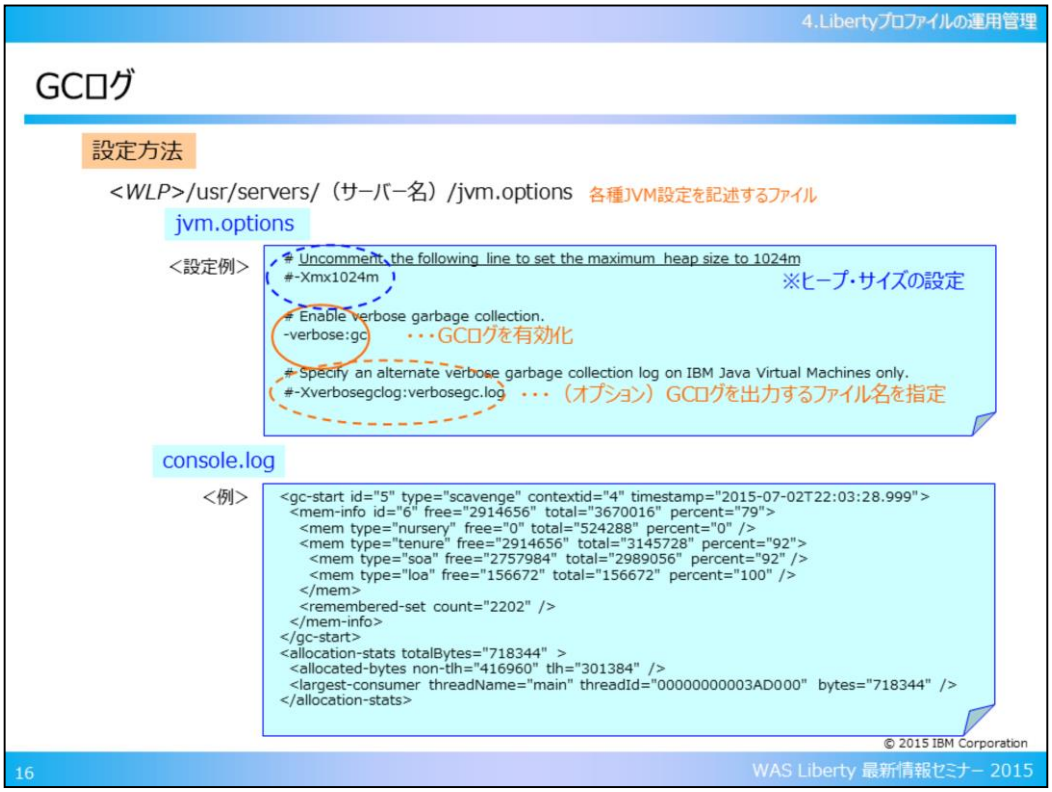
4.Libertyプロファイルの運用管理

【参考】ロギング・プロパティー

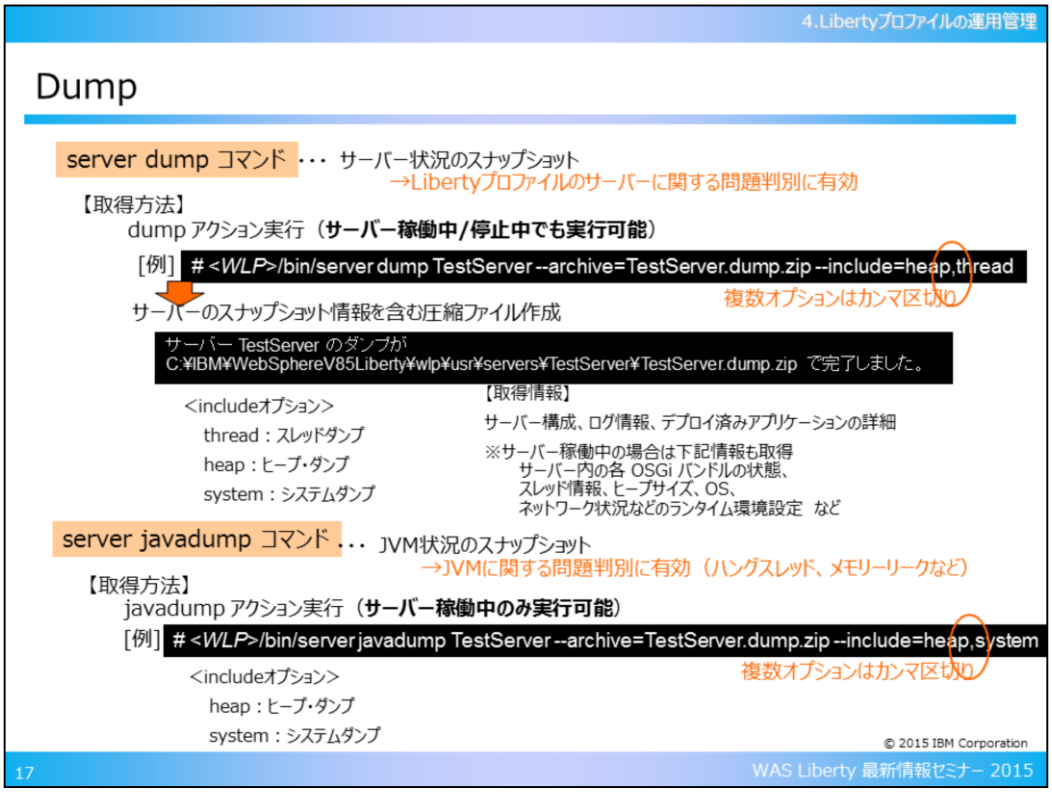
server.xmlで設定する属性	bootstrap.propertiesで設定するプロパティー	説明
logDirectory	com.ibm.ws.logging.log.directory	FFDC を含むすべてのログ・ファイルのディレクトリーを設定します。
maxFileSize	com.ibm.ws.logging.max.file.size	ログ・ファイルの許容最大サイズ(MB)。これを超えるとローテートされます。 これを無効にするには、値を 0 に設定します。
maxFiles	com.ibm.ws.logging.max.files	最大ファイル・サイズが有効な場合に、この設定を使用して、保持する各ログ・ファイルの数を決定します。
consoleLogLevel	com.ibm.ws.logging.console.log.level	console.log ファイルに入れるメッセージの細分度を制御します。有効な値は INFO、AUDIT、WARNING、ERROR、および OFF です。デフォルトのレベルは AUDIT です。
copySystemStreams	com.ibm.ws.logging.copy.system.streams	true の場合、System.out （標準出力）および System.err （標準エラー）は console.log にもコピーされます。false の場合、System.out および System.err は、構成されたログ（messages.log や trace.log など）に書き込まれますが、システム・ストリーム（console.log）には書き込まれません。デフォルト値は true です。
messageFileName	com.ibm.ws.logging.message.file.name	メッセージ・ログのデフォルト名は messages.log です。このファイルは常に存在し、 System.out と System.err に加えて、INFO とその他（AUDIT、WARNING、ERROR、 FAILURE）のメッセージが含まれます。このログには、タイム・スタンプと発行元スレッド ID も含まれます。
suppressSensitiveTrace		サーバー・トレースは、ネットワーク接続経由で受信したバイトなどの型のないデータをトレースする場合に、機密データを公開してしまう可能性があります。 この属性を true に設定すると、機密である可能性のある情報がログ・ファイルおよびトレース・ファイルで公開されないようにします。 デフォルト値は false です。
traceFileName	com.ibm.ws.logging.trace.file.name	追加トレースまたは詳細トレースが有効な場合にのみ作成されます。
traceSpecification	com.ibm.ws.logging.trace.specification	トレース・ストリングを使用して、選択的にトレースを有効にします。デフォルトは *=info=enabled です。
traceFormat	com.ibm.ws.logging.trace.format	トレース・ログのフォーマットを制御します。Liberty プロファイル のデフォルトのフォーマットは ENHANCED です。フルプロファイルのように BASIC と ADVANCED のフォーマットも使用できます。
hideMessage	com.ibm.ws.logging.hideMessage	console.log および message.log から隠したいメッセージ（文字列）をカンマ区切りで指定します。 隠されたメッセージは trace.log にリダイレクトされます。

© 2015 IBM Corporation

copySystemStreamsの属性はデフォルトでtrueのため、System.outおよびSystem.errはmessages.logやtrace.logに加えてconsole.logにも書き込まれます。これをfalseに設定するとconsole.logには書き込まれません。



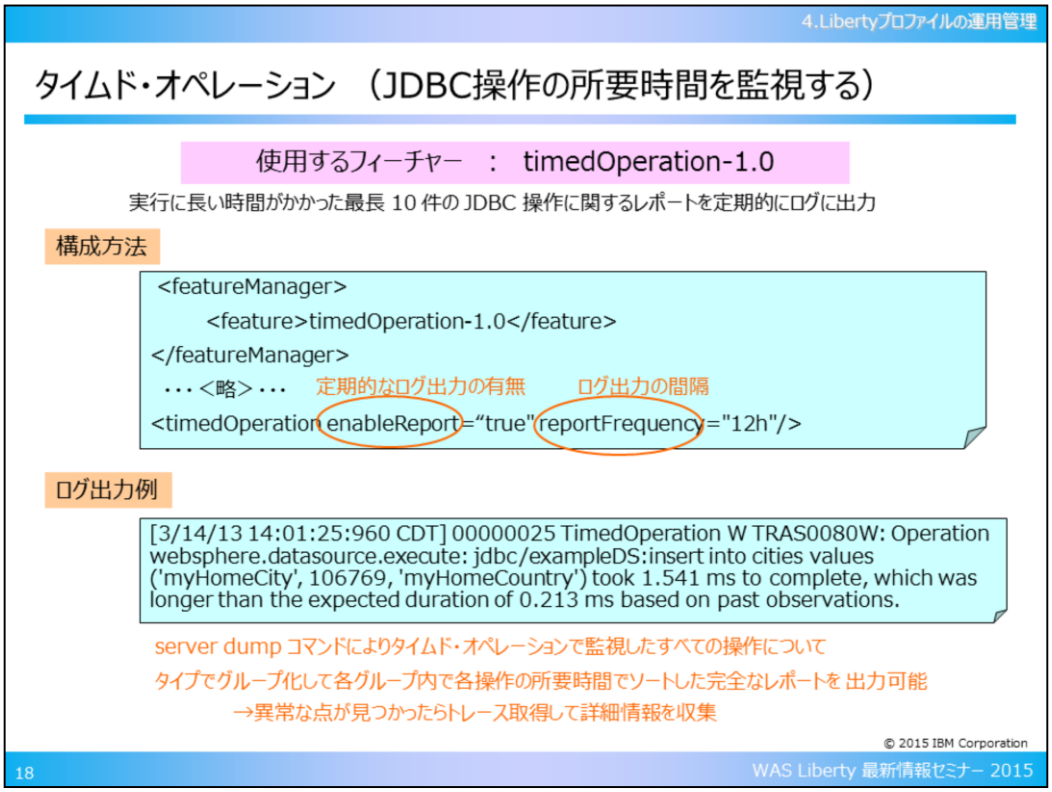
GCログの設定は、jvm.optionsファイルで行います。これは各種JVM設定を記述するファイルで、この中で "-verbose:gc"を指定することで、冗長GCログが出力されるようになります。デフォルトではconsole.logファイルに出力されますが、"-Xverbosegclog:（ファイル名）"を指定することで、他のファイルに出力させることもできます。



Libertyプロファイルでは、サーバー状況あるいはJVM状況のスナップショットを取得する機能が提供されています。取得方法は、serverコマンドのdumpアクションあるいはjavadumpアクションを使用します。

dumpアクションはサーバー稼働中でも停止中でも実行することができ、実行すると、そのサーバーのスナップショット情報を含む圧縮ファイルがLibertyプロファイルの各サーバーディレクトリ配下に作成されます。取得情報は、サーバー構成、ログ情報、デプロイ済みアプリケーションの詳細などです。サーバー稼働中にdumpアクションを実行した場合は、それに加えて、サーバー内の各 OSGi バンドルの状態、登録されたOSGiサービスの情報、スレッド情報、JVM、ヒープサイズ、オペレーティング・システム、ネットワーク状況などのランタイム環境設定などの詳細な情報が取得できます。このサーバーのスナップショット情報は、WDTのユーティリティー機能で取得することもできます。また、コマンド実行時にincludeオプションを使用することにより、圧縮ファイル内にスレッドダンプ（javacore）やヒープ・ダンプ、システムダンプも含めることができます。

Javadumpアクションはサーバー稼働中のみ実行することができ、実行すると、そのサーバーのスレッドダンプ（javacore）がLibertyプロファイルの各サーバーディレクトリ配下に作成されます。また、コマンド実行時にincludeオプションを使用することにより、同時にヒープ・ダンプやシステム・ダンプも作成されます。"--include"で複数オプションを指定する場合は、"heap,system"のようにカンマで区切ります。



V8.5.5では、実行に長時間かかったJDBC処理を監視するタイムド・オペレーションという新機能が提供されています。

タイムド・オペレーションを有効にすると、アプリケーションのJDBC処理のうち、実行に長時間かかったものを監視し、そのレポートを定期的にmessages.logに出力させることができます。

構成方法は、server.xmlに<timedOperatoin>エレメントを記述し、定期的なログ出力の有効無効や出力の時間間隔を指定します。

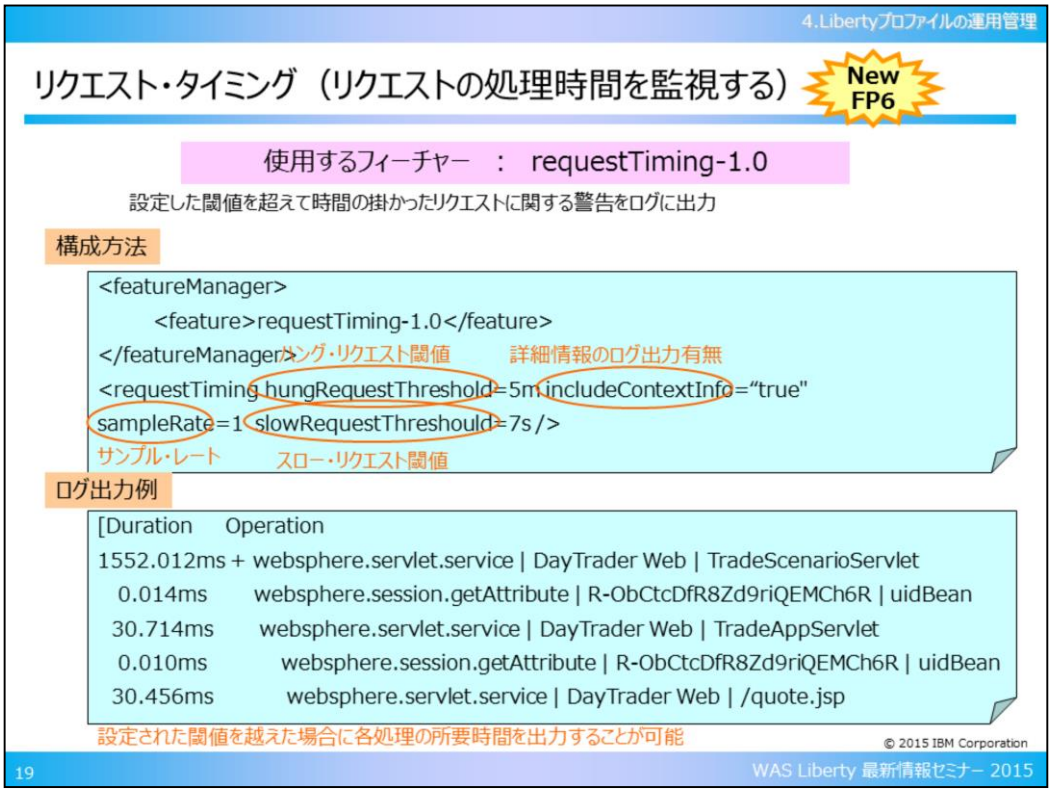
また、ログに出力させるだけでなく、serverコマンドのdumpアクションを実行することにより、タイムド・オペレーションで監視した全ての操作について、タイプでグループ化して各グループ内で各操作の所要時間でソートした完全なレポートを出力させることができます。

通常のmessages.logに出力させた情報を確認し、問題がありそうであればserverコマンドのdumpアクションでレポートを確認し、更に詳細な情報が必要であればトレースを出力させるといった対応により、問題判別を行います。設定可能な属性は以下の通りです。

maxNumberTimedOperations 時間指定操作の総数がこの値に達すると、警告が記録されます。デフォルト値は"10000"です。

enableReport 最長の時間指定操作 10 件について、タイプ別にグループ化し、予想される所要時間によって各グループ内でソートした詳しいレポートを、定期的にログに生成できるようにします。デフォルト値は"true"です。

reportFrequency 最長の時間指定操作 10 件について、タイプ別にグループ化し、予想される所要時間によって各グループ内でソートした詳しいレポートを、ログに生成する頻度。正整数の後に時間単位 (時間 (h)) を付けて指定してください。



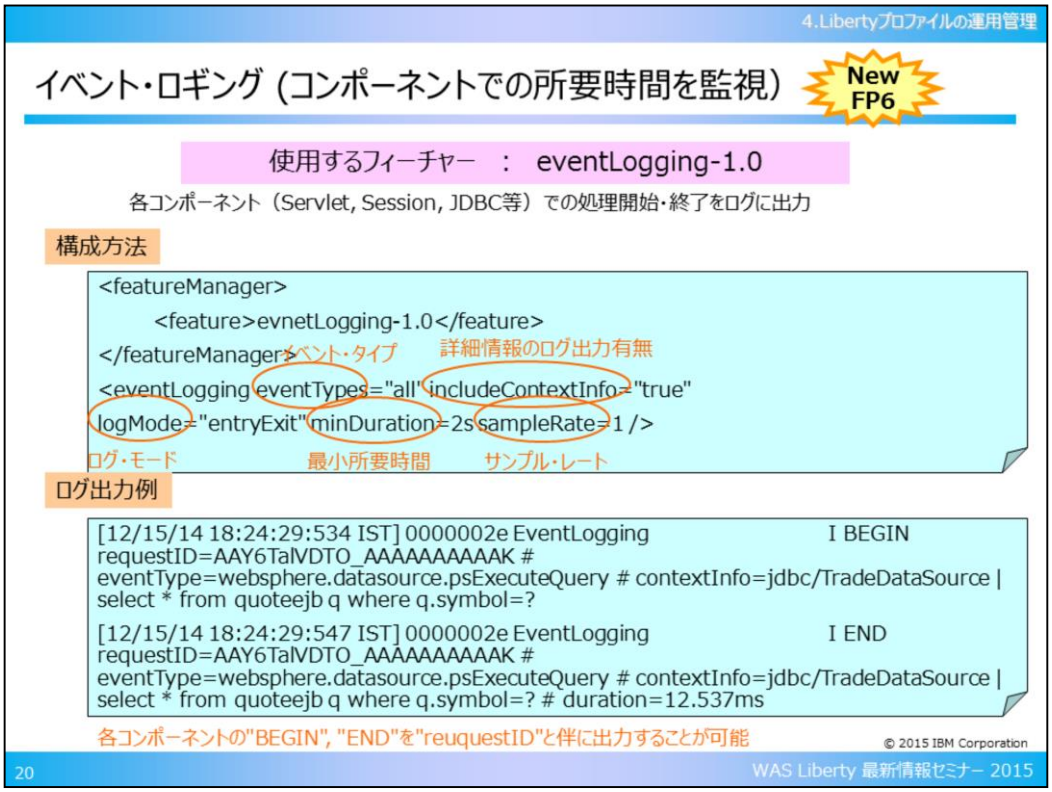
requestTiming-1.0 はFixPack 6からの新フィーチャーであり、リクエストの処理時間を監視するために利用します。設定された閾値を超えた場合にログが出力されます。設定可能な属性は以下の通りです。

hungRequestThreshold　　リクエスト処理がハングしていると判定するための閾値です。デフォルト値は"10m"（10分）です。

includeContextInfo　　ログにコンテキスト情報（詳細情報）を含めるかを指定します。デフォルト値は"true"です。

sampleRate　　サンプル・レートを指定します。Nを指定した場合、N個のリクエスト毎に1回サンプリングされます。デフォルト値は"1"で、毎回のリクエストをサンプリング対象としています。

slowRequestThreshold　　リクエスト処理が遅いと判定するための閾値です。デフォルト値は"10s"（10秒）です。



eventLogging-1.0はFixPack 6からの新フィーチャーでコンポーネントでの所要時間を監視します。設定された最低所要時間の閾値を超えた場合にログに出力されます。

ログには"requestID"と伴に出力することが可能なので、特定のリクエストをトラックすることが容易にできます。設定可能な属性は以下の通りです。

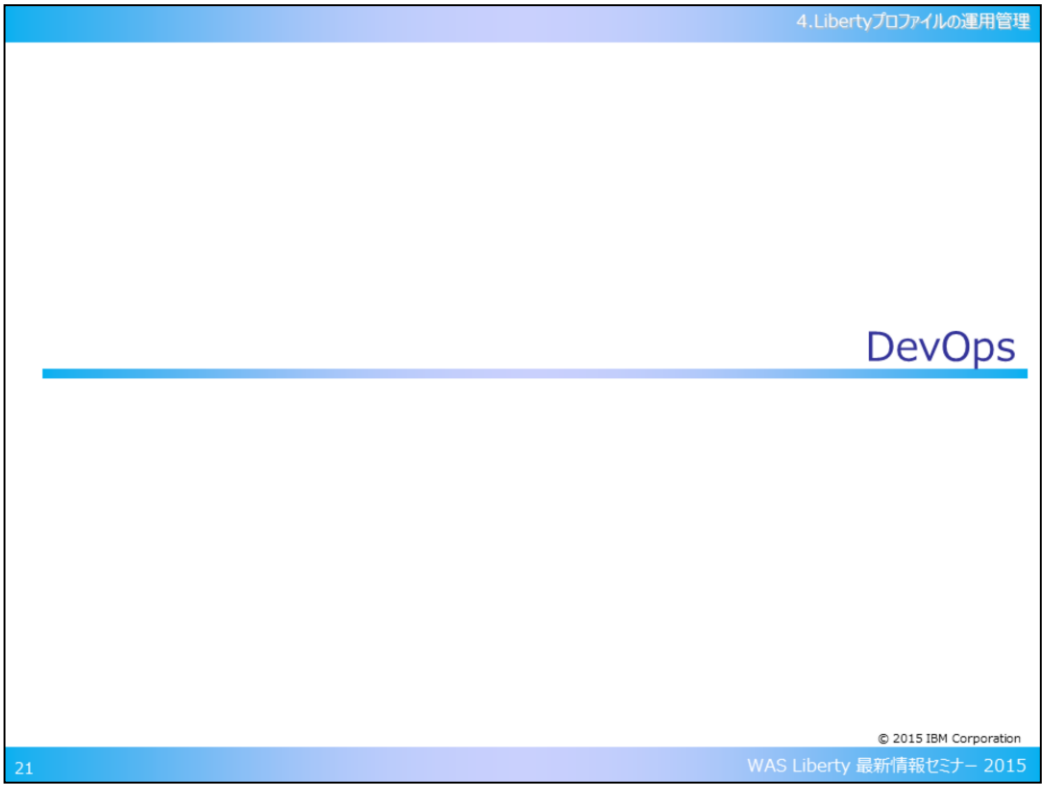
eventTypes　監視対象とするイベント・タイプをカンマ区切りで指定します。 デフォルト値は"all"で、すべてのイベント・タイプを監視対象としています。

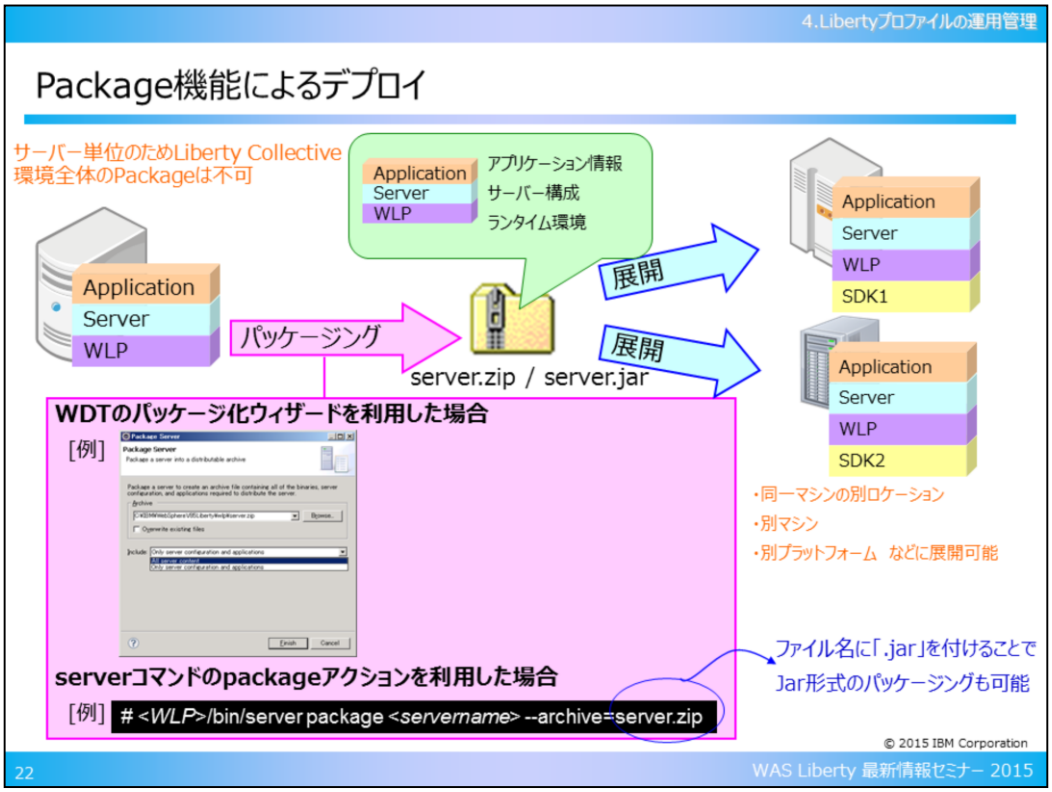
includeContextInfo　ログにコンテキスト情報（詳細情報）を含めるかを指定します。デフォルト値は"true"です。

logMode　監視のタイミングを"entry", "entryExit", "exit"の中から指定します。デフォルト値は”exit"です。

minDuration　コンポーネントでの処理時間の最小時間を指定します。デフォルト値は"1s"（1秒）です。

sampleRate　サンプル・レートを指定します。Nを指定した場合、N個のリクエスト毎に1回サンプリングされます。デフォルト値は"1"で、毎回のリクエストをサンプリング対象としています。


4.Libertyプロファイルの運用管理
DevOps
© 2015 IBM Corporation
21
WAS Liberty 最新情報セミナー 2015

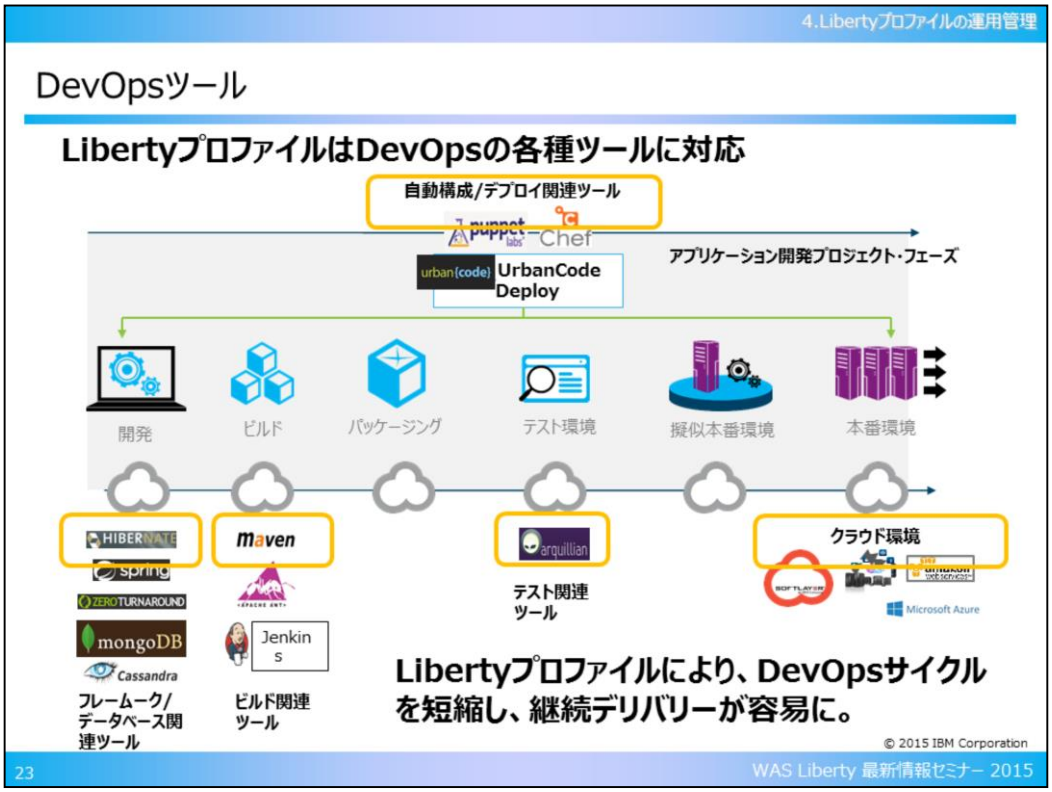


Package機能によるデプロイとは、一度構築したランタイム環境、サーバー構成、アプリケーション情報を全て1つのzip形式の圧縮ファイルにパッケージングし、それをzip展開することによってLibertyプロファイルを導入するという方法です。

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要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。



LibertyプロファイルはDevOpsの各種ツールに対応しています。

上記の図はアプリケーション開発の各プロジェクトフェーズと、各種DevOpsツールの関係をまとめたものです。
次のページで各ツールの情報を整理します。

4.Libertyプロファイルの運用管理

【参考】各種DevOpsツールでのLibertyプロファイル利用方法(1/2)

WASdevサイト(英語)ではDevOpsツールについての豊富な情報が提供されています

- 自動構成・自動デプロイ関連ツール
  - IBM UrbanCode Deploy ・・・自動デプロイ・ツール
    https://developer.ibm.com/wasdev/blog/tag/urbancode-deploy/
  - Puppet ・・・オープンソース、構成管理ツール
    https://developer.ibm.com/wasdev/blog/tag/puppet/
  - Chef ・・・Chef社、構成管理ツール
    https://developer.ibm.com/wasdev/blog/tag/chef/

- フレームワーク/データベース関連
  - Hibernate ・・・オープンソース、ORMフレームワーク
    https://developer.ibm.com/wasdev/blog/tag/hibernate/
  - Spring Framework ・・・オープンソース、JEEフレームワーク
    https://developer.ibm.com/wasdev/blog/tag/spring/
  - JRebel ・・・Zero Turnaround、開発支援、再デプロイツール
    https://developer.ibm.com/wasdev/blog/tag/jrebel/
  - MongoDB ・・・オープンソース、NoSQLデータベース
    https://developer.ibm.com/wasdev/blog/tag/mongodb/
  - Apache Cassandra ・・・オープンソース、分散データベース管理ツール
    - https://developer.ibm.com/wasdev/downloads/#asset/osi-Cassandra_Sample

© 2015 IBM Corporation

【参考】各種DevOpsツールでのLibertyプロファイル利用方法(2/2)

- ビルド関連ツール
  - Apache Maven ・・・オープンソース、Javaプロジェクト管理ツール
    https://developer.ibm.com/wasdev/blog/tag/maven/
  - Apache Ant ・・・Javaビルド・ツール
    https://developer.ibm.com/wasdev/blog/tag/ant/
  - Jenkins ・・・継続インテグレーション・ツール
    https://developer.ibm.com/wasdev/blog/tag/jenkins/
- テスト関連ツール
  - Arquillian ・・・統合テスト・ツール
    https://developer.ibm.com/wasdev/blog/tag/arquillian/
- 全般
  - IBM Redbook
    Configuring and Deploying Open Source with WebSphere Application Server Liberty Profile
    http://www.redbooks.ibm.com/abstracts/sg248194.html
    http://www.redbooks.ibm.com/redpieces/abstracts/sg248194.html（新バージョン・ドラフト版, 2015年7月現在）

© 2015 IBM Corporation

まとめ

- 基本操作
  - コマンド・ラインから起動・停止・状況確認
  - 本番環境での構成では動的アップデートを抑止
- モニター
  - JConsole/JMXクライアント等から可能
  - AdminCenterからも一部可能
- ログ
  - messages.log, console.logがあり、各種設定が可能(server.xml, bootstrap.properties)
  - GCログは、jvm.optionsファイルで設定可能
- 問題判別
  - 2種類のDumpコマンド（server dump, server javadump)
  - 監視用Featureにより問題判別が容易に
- DevOps
  - 各種DevOpsツールに対応
  - WASdevサイト(英語)から豊富に情報提供

© 2015 IBM Corporation

4.Libertyプロファイルの運用管理

© 2015 IBM Corporation