

Oracle Direct Seminar



ORACLE®

GlassFish v3.1 のご紹介

日本オラクル株式会社 シニア Java エバンジェリスト
寺田 佳央

Oracle Direct



Agenda

- はじめに
- Java EE 6 の概要
- Oracle GlassFish Server v3 の概要
- Oracle GlassFish Server v3 の詳細
- GlassFish v3.1の新機能
- まとめ

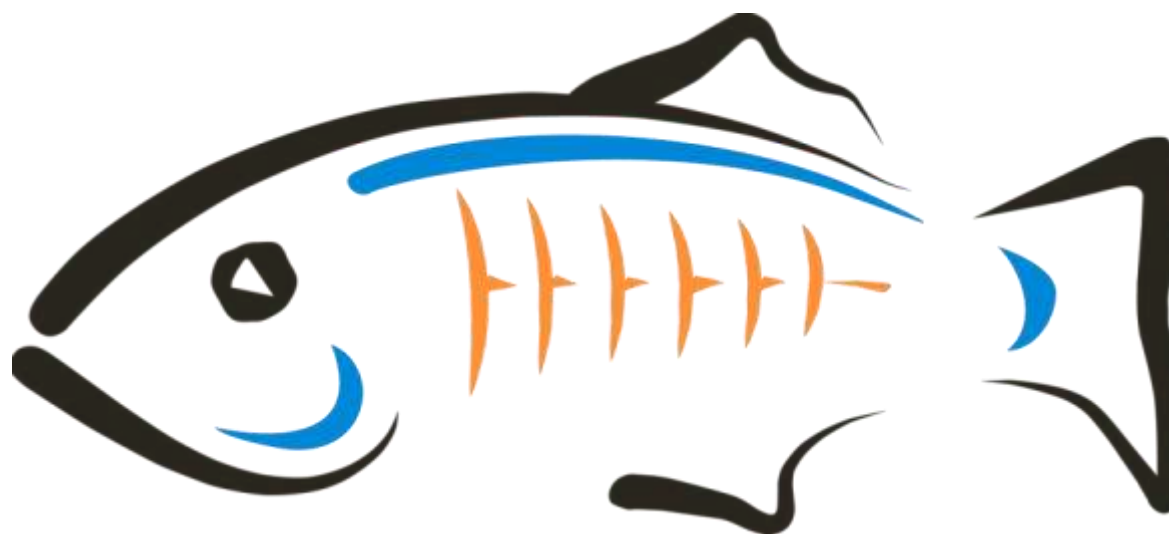




はじめに

GlassFish 3.1

祝



2011 年 2 月 28 日正式リリース

ORACLE

GlassFish と Java コミュニティ

- GlassFish とは

- OSS コミュニティ名
- アプリケーションサーバ名

- GlassFish コミュニティ

- 全てをコミュニティ上で(設計、開発、テスト)
- 多数の外部協力者の貢献
- ブログ/Wikiによる情報提供
- フォーラム/MLによるサポート
- CDDL/GPLv2デュアルライセンスで提供

- GlassFish アプリケーションサーバ

- GlassFish Server Open Source Edition (オープンソース製品)
- Oracle GlassFish Server (Oracle Fusion Middleware製品)



GlassFish » Community



[Start](#)



[Download](#)



[Communicate](#)



[Contribute](#)

<http://glassfish.org>



<http://www.oracle.com>

ORACLE

GlassFish の歴史

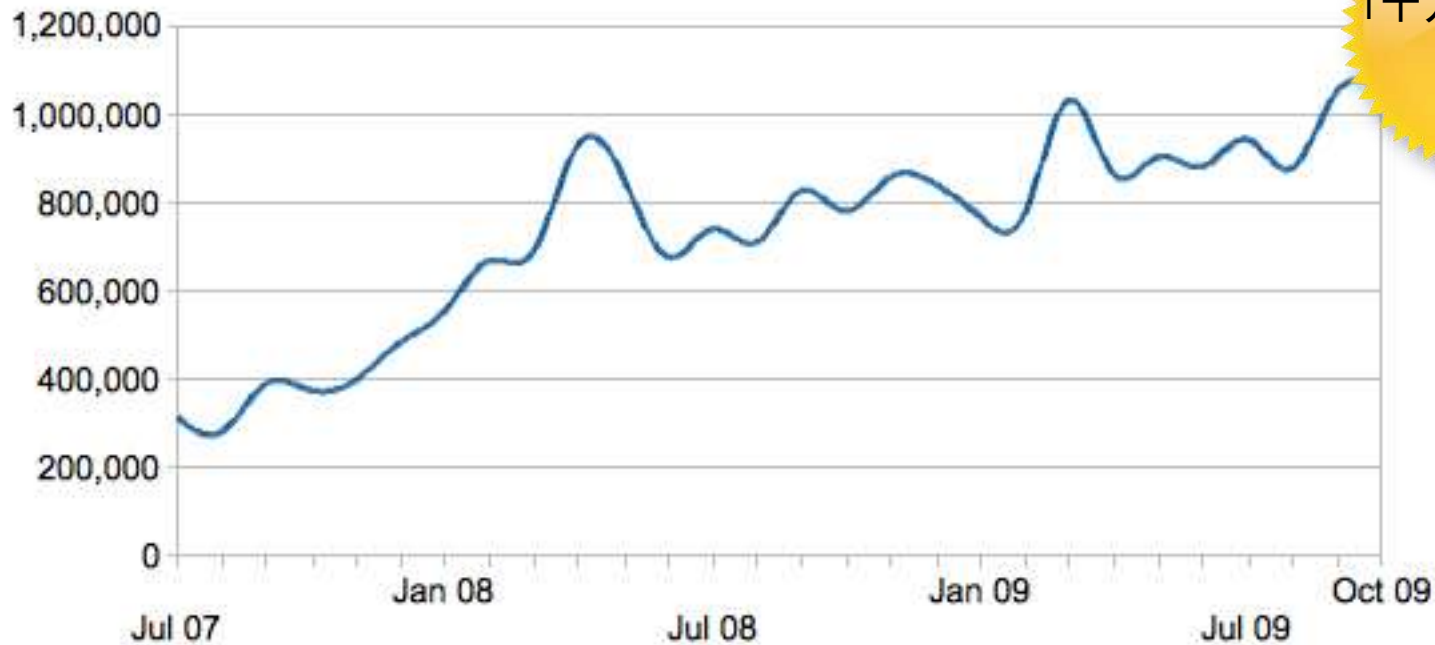


2005年 プロジェクト開始	2006年 Ver 1	2007年 Ver 2	2009年 Ver 3	2011年 Ver 3.1
-------------------	----------------	----------------	----------------	------------------

- **GlassFish v1**
 - Java EE 5 参照実装
 - オープンソースとして提供した最初のバージョン
- **GlassFish v2**
 - Java EE 5 参照実装
 - クラスタ機能(負荷分散／高可用性)
- **GlassFish v3**
 - Java EE 6 参照実装
 - デザインを刷新した新しいアーキテクチャ
 - OSGi モジュール化サブシステムに対応

月間ダウンロード数の推移

- 世界で最も成長率の高いアプリケーションサーバ



1千万Download
/1Year

オープンソースに対する Oracle の付加価値

オープンソース

- オープン開発
- 無料で利用可能
- 公開されたテスト
- フィードバック
- メーリングリスト
- フォーラムサポート



有償ユーザ

- 評価済みの安定製品
- ライセンス提供
- 障害サポート提供
- バグ修正/パッチ提供



Oracleの付加価値

- コンサルティング
- OEMライセンス契約



ORACLE®

GlassFish v3.1 リリースのアナウンス

2/28 正式リリース

- モジュール化され軽量な Java EE 6 実行環境
 - クラスタ機能の追加
 - ドメイン毎にマルチクラスタ環境を構築可能
 - クラスタ無いに複数インスタンスを構成可能
 - 1ドメイン辺り最高100インスタンスまで対応可能
 - ssh プロビジョニング
 - 中央集中管理機能
 - 高可用性機能の追加
 - GlasFish v2.x の拡張と最適化
 - GlassFish 組み込み API の改善
 - JAX-RS(RESTful API) の改善

今すぐ本番環境へ適用可能

- クラスタ機能と高可用性機能の改善
 - GlassFish v2.1.1 に比べ **34 %のパフォーマンス改善**
 - リモートノードの管理に SSH 経由の管理機能を提供
 - ロードバランサプラグインによるフェールオーバー
 - 専用インストーラを提供
 - glassfish-lbconfigurator-3_1.zip
 - Metro の高可用性
 - Reliable messaging sequence failover
 - Secure conversation session failover
 - 組み込みモードのMQブローカに対するクラスタリング
 - 幅広いプラットフォームのサポート

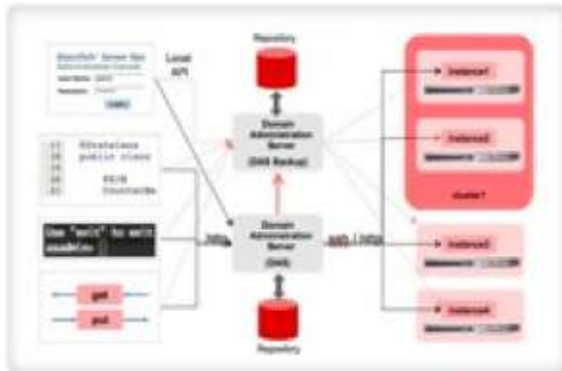
GlassFish の製品一覧

Distribution	License	Features
GlassFish Server Open Source Edition 3.1 <i>Web Profile</i>	CDDL & GPLv2	<ul style="list-style-type: none">• Java EE 6 compatibility• Web Profile support• In-memory replication / clustering• Centralized Administration
GlassFish Open Source Edition 3.1	CDDL & GPLv2	<ul style="list-style-type: none">• Java EE 6 compatibility• Full Java EE distribution• In-memory replication / clustering• Centralized Administration
Oracle GlassFish Server 3.1 <i>Web Profile</i>	Commercial	<ul style="list-style-type: none">• Adds<ul style="list-style-type: none">• Oracle GlassFish Server Control• Patches, support, knowledge base
Oracle GlassFish Server 3.1	Commercial	<ul style="list-style-type: none">• Adds<ul style="list-style-type: none">• Oracle GlassFish Server Control• Patches, support, knowledge base

製品版で提供される拡張管理機能

GlassFish Server Control

DAS Backup & Recovery



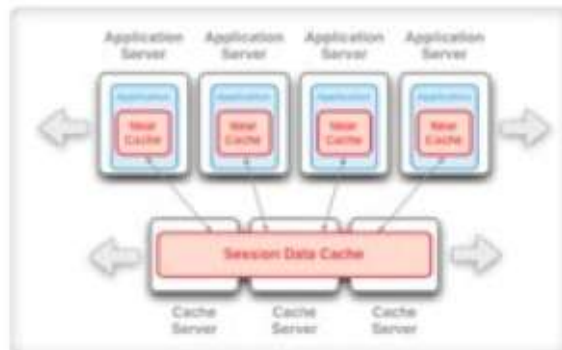
Performance Tuner



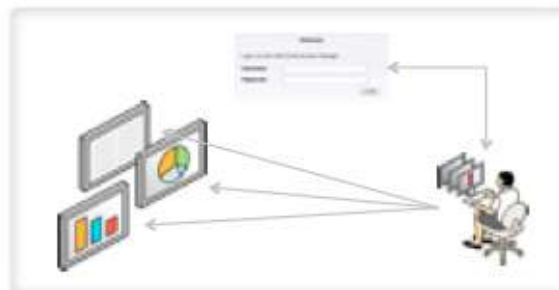
Monitoring
Scripting Client

Thread Id	Count	Duration
http-thread-pool-8080-(3)	2	52
http-thread-pool-8080-(4)	9	221
http-thread-pool-8080-(5)	9	1769
http-thread-pool-8080-(6)	5	122
http-thread-pool-8080-(7)	0	1
http-thread-pool-8080-(8)	2	457
http-thread-pool-8080-(9)	17	237
http-thread-pool-8080-(10)	20	552
http-thread-pool-8080-(11)	21	253
http-thread-pool-8080-(12)	18	146
http-thread-pool-8080-(13)	7	93
http-thread-pool-8080-(14)	2	43
http-thread-pool-8080-(15)	12	128
http-thread-pool-8080-(16)	9	127
http-thread-pool-8080-(17)	6	71

ActiveCache for GlassFish



Oracle Access
Manager Integration



Load Balancer
Plugin & Installer



ORACLE®

製品版で提供される拡張管理機能

GlassFish Server Control

- Oracle Access Manager (OAM) との統合
- WebSphere/WebLogic と JMS による総合運用性
- ドメイン管理サーバのバックアップ/リストア
- パフォーマンスチューナー
- モニタリングスクリプトクライアント
- インストーラによるロードバランサプラグインの導入
- Oracle DataBase Thin Driver
- MySQL Database Driver
- Coherence *Web (Ver 3.7) 2011年 の中旬リリース
- DTrace モニタリング (JDK 7 が必須)

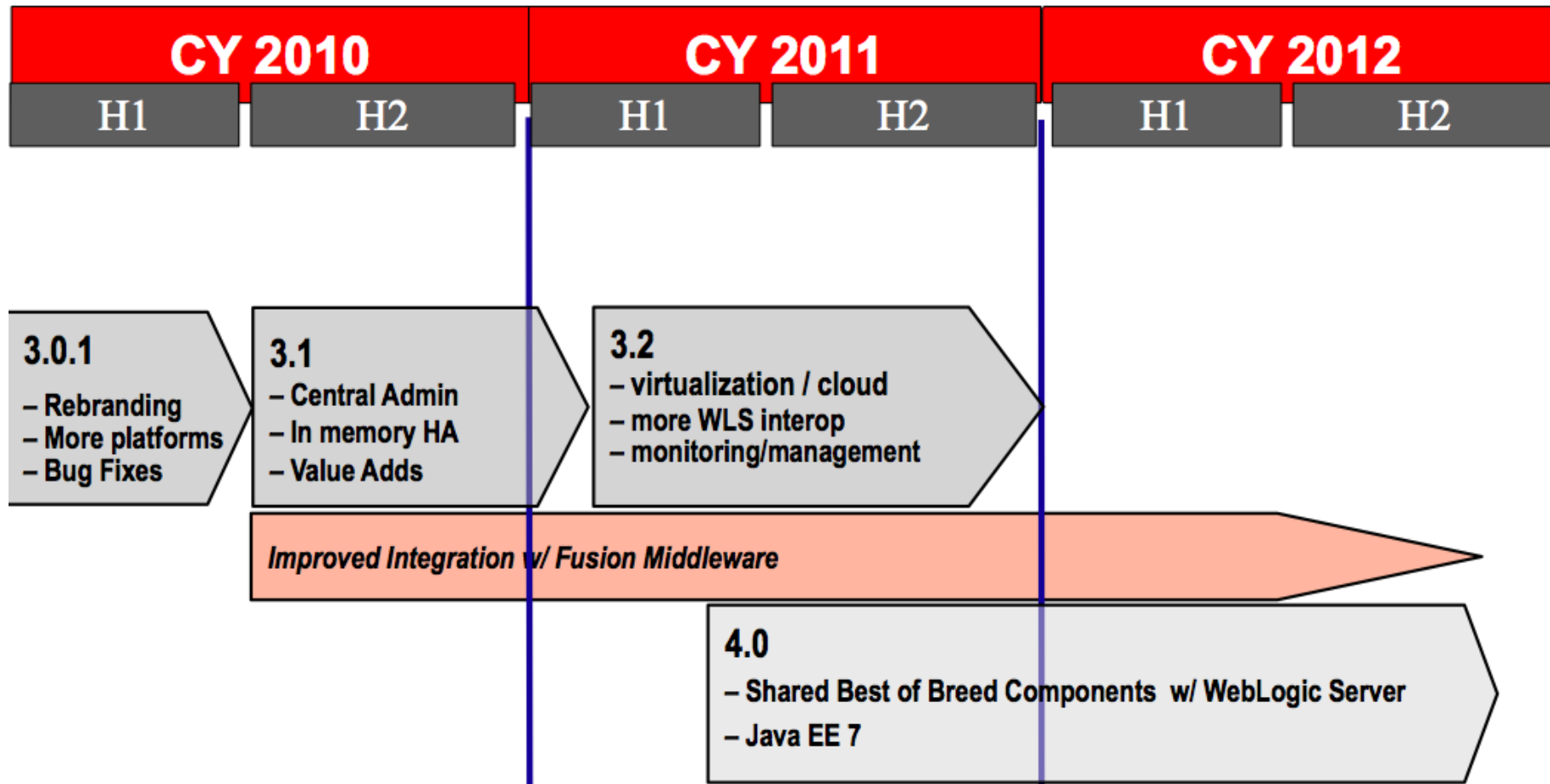
アップグレード/アップデート

- アップグレードのオプション
 - GlassFish v2.x → v3.1 サイドバイサイドインストールと移行
 - 3.x (例 3.0.x) → v3.1 アップデートセンターによるアップグレード
- ドメイン起動時のアップグレード
 - `asadmin start-domain -upgrade mydomain`

参考資料

- オープンソース版
 - 製品情報
 - <http://glassfish.org>
 - PDF を ZIP にまとめた形式
 - <http://glassfish.org/docs>
- 製品版
 - 製品情報
 - <http://www.oracle.com/goto/glassfish>
 - オンライン PDF ファイル
 - http://download.oracle.com/docs/cd/E18930_01/index.htm

GlassFish オープンソース版ロードマップ





Java EE 6 の概要

Java EE 6 の参照実装

開發生産性の大幅な向上

- 拡張性
- プロファイル
 - Webプロファイル
 - Enterprise Platform
- Pruning
 - 使われなくなった仕様の削減
- かんたん開発
 - 新技術の追加
 - DI, CDI, JAX-RS, Bean Validation 等
 - 更新技術
 - JPA 2.0, EJB 3.1, JSF 2.0 等



ORACLE

Java EE 6

拡張性

- Java EE 以外のフレームワークも容易に利用可能
 - Spring, Struts, Wicket 等 3rd パーティフレームワークを利用可能
 - 複雑な設定は不要
 - 追加するフレームワークの Servlet、Servlet フィルタ、コンテキストリスナーは自動検知、自動登録
 - フレームワーク毎の設定項目は web fragment 設定ファイルに集約

Java EE 6

プロファイル

- Java EEの技術を用途毎に分割して提供
 - Java EEのサブセットを提供
- 独自プロファイルの開発が可能
 - 例: 電話会社向けプロファイル
- Java EE 6で最初に提供されるプロファイル
 - Webプロファイル(Webの開発に特化)
 - Enterprise Platform(フルJava EE)



Web Profile

X Profile

Y Profile

Full Java EE 6 (Enterprise Platform)

ORACLE

Web プロファイル

Webアプリケーションの開発に特化した軽量プロファイル

▪ Webプロファイルに含まれる技術

- Servlet
- JSP / EL
- JSTL
- JSF
- Bean Validation
- EJB Lite
- JPA
- JTA
- DI/CDI
- Managed Beans
- Interceptors
- Common Annotations



Java EE 6

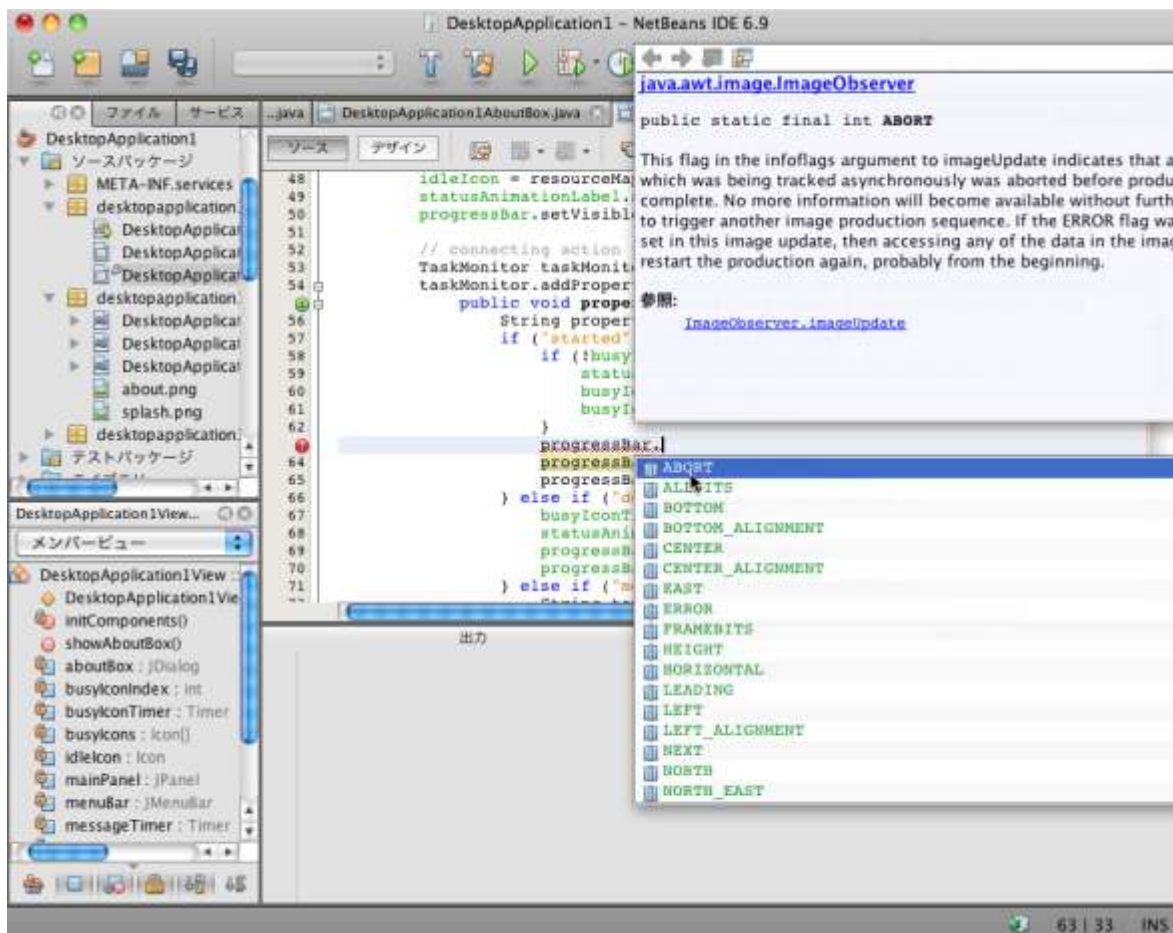
仕様の削減 (Pruning)

- 2段階プロセス
 - 古く使われなくなったAPIの整理
 - コンポーネントのオプション化
 - 次期バージョン(Java EE 7)で オプション化
 - JAX-RPC(->JAX-WS)
 - EJB Entity Beans(->JPA)
 - JAXR
 - JSR-88



かんたん開発

開発を容易にする様々な統合開発環境



ORACLE
JDEVELOPER



ORACLE

Java EE 6 の主要な技術

- Servlet 3.0
- JSP 2.2
- JSF 2.0
- EJB 3.1
- JTA 1.1
- JPA 2.0
- JAX-WS 2.2
- JAXB 2.2
- JAX-RS 1.1
- DI 1.0/CDI 1.0
- Bean Validation 1.0
- JMS 1.1
- JavaMail 1.4
- EL 2.2
- Common Annotations 1.1
- Managed Bean 1.0
- Interceptors 1.1
- JACC 1.4
- その他

さらに進んだかんたん開発

Servlet 3.0

JSR-315

- 特徴
 - 設定ファイル(web.xml)のオプション化
 - アノテーションベースの設定
 - 拡張性
 - web-fragments.xml の提供
 - マルチパート対応
 - ファイルアップロード
 - 非同期 Servlet のサポート
 - セキュリティ(login/logout処理に対応)

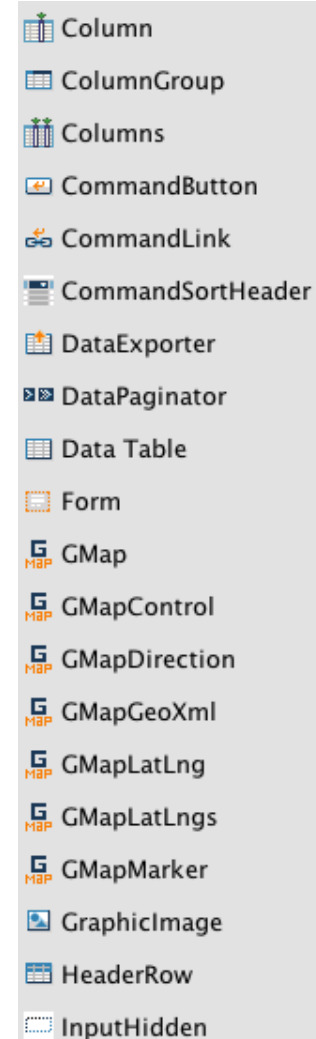
Java SE 5の言語仕様で新たに追加されたアノテーションを使用し宣言的プログラミングモデルを採用。またジェネリクスの利用も可能

JavaServer Faces 2.0

JSR-314



- コンポーネントベース開発 (MVC)
 - Facelets による実装
 - テンプレート機能の提供
 - カスタム複合コンポーネント
 - Ajax 対応
 - ブックマーク可能なページ
- 設定項目の簡略化
 - ページナビゲーションの改良
 - faces-config.xml のオプション化
- Bean Validation のサポート
- Servlet コンテナのバージョンに非依存
 - Servlet 3.0 コンテナの他 2.5 上でも動作可能



EJB 3.1

JSR-318

- 特徴
 - パッケージの簡略化
 - EJB 3.1 “Lite” の提供
 - ローカルビジネスインタフェースのオプション化
 - 標準化された Global JNDI名
 - Java SEに組み込み可能なEJBコンテナ
 - その他の新機能

EJB Lite

軽量版の提供 (Web Profileで利用可)

Full EJB 3.1機能のサブセットを提供

- Lite
 - ローカルセッションBeans
 - CMT/BMT
 - Declarative Security
 - Interceptors
- Full = Lite +
 - Message-Driven Beans
 - Web Service Endpoint
 - 2.x/3.x Remote view
 - RMI-IIOP Interoperability
 - Timer Service
 - Async method call
 - 2.x Local view
 - CMP/BMP Entity

Bean Validation 1.0

- アプリケーション中で宣言的なバリデーションが可能
- カスタムバリデーションを作成可能
- 1度の制限でどこでもバリデート可能
 - Bean、フィールド、プロパティに制限
 - Null チェック、数値適用範囲チェック、メールアドレスチェック等
- JSF 2.0 に統合
 - f:validateRequired, f:validateRegexp
 - ManagedBean
 - @NotNull, @Size(max=40) String address 等
- JPA 2.0 に統合
 - Entity クラス
 - @NotNull, @Size(max=40) String address 等

JAX-RS 1.1

- RESTful サービス用の高レベル HTTP API
- POJO とアノテーションベース
 - API の利用が可能
- HTTP メソッドとのマッピング
 - GET, POST, PUT, DELETE 等
- EJB と統合された JAX-RS

Dependency Injection

DI 1.0 / CDI 1.0

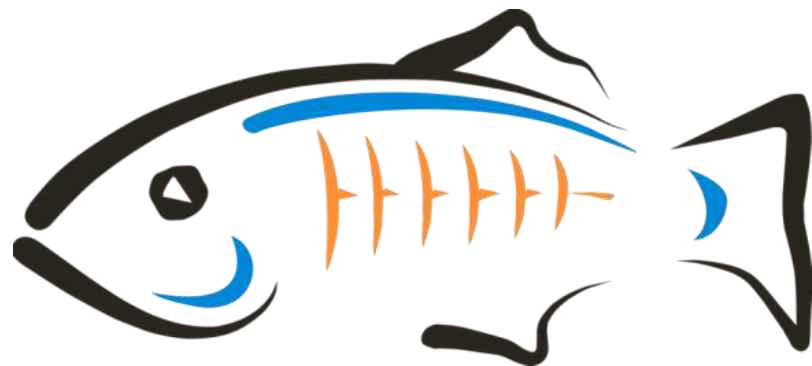
- @Inject アノテーション
 - @Inject @LoggedIn User user;
- Injection メタモデル
 - どんなBeanもInject対象
 - EJB session beans
 - Plain classes with @ManagedBean
 - CDIがモジュール内で見つけたクラス
 - デフォルトで無効、有効化する場合は、beans.xmlを配置
 - META-INF/、WEB-INF/に配置



Oracle GlassFish Server v3 の概要

GlassFish v3 の概要

- **Java EE 6 の参照実装**
 - Java EE 6 の仕様に完全準拠
 - 軽量、高速起動
 - かんたん開発／かんたん管理
 - プロファイルに対応
 - Webプロファイル版、Full Platform プロファイル版を提供
- **先進的アプリケーションサーバ**
 - RESTfulの管理インタフェースを提供
 - OSGiモジュールサブシステム対応
 - 進化した非同期 I/O サーバ
- **かんたんな操作**
 - unzipによるインストール(Tomcatと同様)
- **Oracle JRockit VM 正式対応**



Oracle GlassFish Server の入手

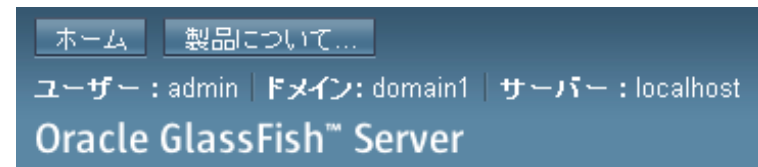
- Full Platform プロファイル
 - 完全な Java EE API を利用する場合に使用
- Web プロファイル
 - Full Platform に含まれるライブラリの内 Web 技術に特化した開発・運用を行う場合に使用
 - Web プロファイルから更新ツールを利用し Full Platform へアップグレード可能
- 各プロファイルの入手先

- <http://www.oracle.com/technetwork/middleware/glassfish/index.html>



オープンソース版と製品版の違い

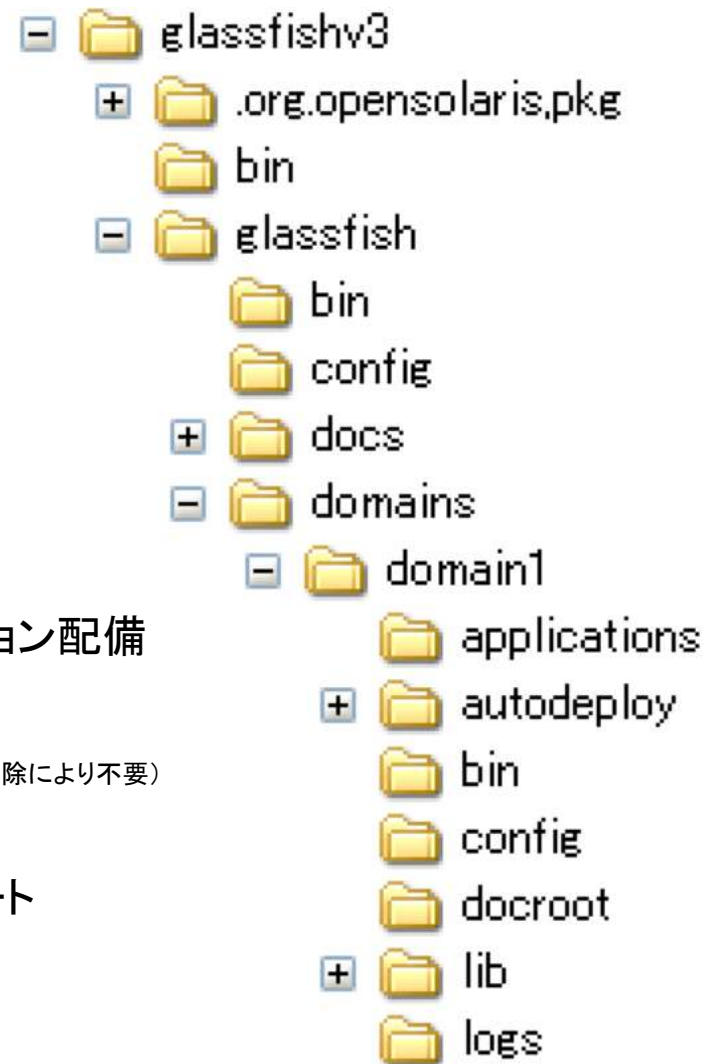
- オープンソース版
 - 無償利用可能・コミュニティドリブン
- 製品版
 - Oracleによる正式サポートを提供
 - 製品のアドオン・追加管理機能等を提供



```
C:\¥glassfishv3¥bin> asadmin version --verbose
バージョン = Oracle GlassFish Server 3.0.1 (build 22)、JRE バージョン
1.6.0_21
コマンド version は正常に実行されました。
C:\¥glassfishv3¥bin> asadmin version --verbose
バージョン = GlassFish Server Open Source Edition 3.0.1 (build
22)、JRE バージョン 1.6.0_21
コマンド version は正常に実行されました。
```

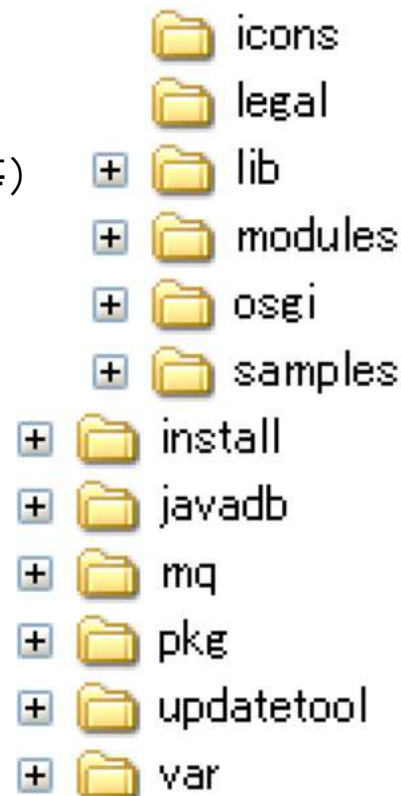
GlassFish v3 のディレクトリ構造

- インストールディレクトリ
 - IPS パッケージディレクトリ
 - 共通ユーティリティコマンドへのリンク
 - GlassFish の本体
 - 実行可能コマンド(asadmin 等)
 - インストール全般の設定
 - ドキュメント
 - ドメイン管理用ディレクトリ
 - デフォルトで作成されるドメイン
 - デプロイされたアプリケーション配備
 - オートデプロイ用ディレクトリ
 - 空ディレクトリ(削除予定,startservの削除により不要)
 - ドメイン固有の設定
 - デフォルトのドキュメントルート
 - ドメイン固有のライブラリ
 - ドメインのログファイル



GlassFish v3 のディレクトリ構造

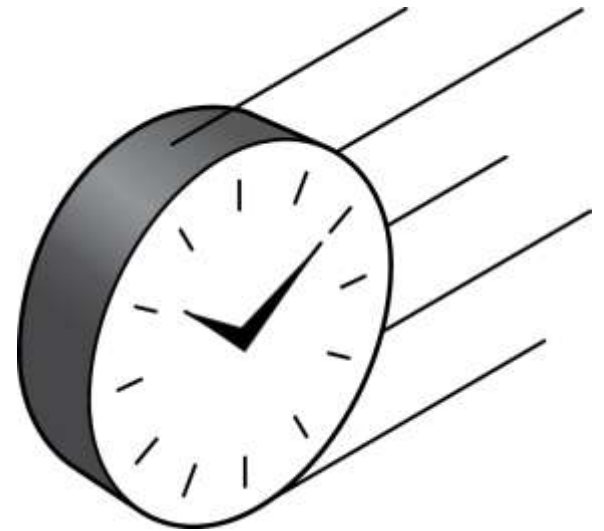
- ・ アイコンファイル
 - ・ 法務上必要なファイル(ライセンス、コピーライト)
 - ・ GlassFish 全般で使用するライブラリ(dtd,xsd 等)
 - ・ モジュール(各機能の jar 等)
 - ・ OSGi フレームワークのファイル
 - ・ サンプルアプリケーション
-
- ・ インストール時使用ファイル
 - ・ 内蔵 JavaDB の本体
 - ・ 内蔵 OpenMQ の本体
 - ・ IPS コマンドツール
 - ・ 更新ツール GUI ツールデ
 - ・ インストール時使用



OSGi 対応により GlassFish v2.x から大幅に変更
GlassFish v3.1 より nodeagent ディレクトリが追加

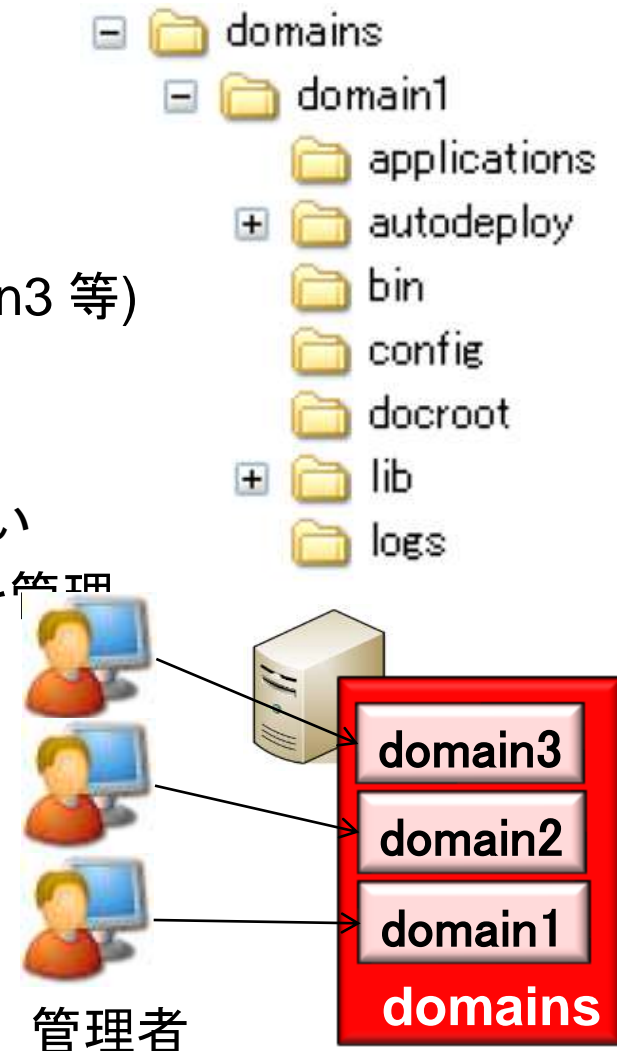
起動時間の劇的な短縮

- 超高速起動 (Web Profile 版利用時)
 - 起動時間約 4 秒 (CPU: 2.4GHz Core 2Duo/4GB)
 - 軽量コンテナ
 - 使われない機能を未初期化
 - 開発効率の大幅な向上
 - 再起動時間の待ち時間の減少



GlassFish のドメイン

- GlassFish の管理構成単位
 - デフォルトで作成されるドメイン: domain1
 - 複数のドメインを作成可能 (domain2, domain3 等)
 - ドメイン毎に管理者を設定
 - 各ドメイン間は独立して管理
 - 設定変更は他のドメインに影響を及ぼさない
 - 管理者は管理コンソール等を通じてドメインを管理
 - Web 管理コンソール
 - コマンドライン管理コンソール
 - RESTful 管理チャネル
 - JMX/AMX 管理チャネル



ORACLE

Web 管理コンソール

- Web ベースの管理コンソール
 - http://IP_ADDRESS:PORT/
 - デフォルトのポート番号: 4848
 - 管理者のパスワード
 - パスワード無し
 - インストール時に指定
 - 管理コンソールアクセス時にロード
 - 管理コンソールとサービス提供ポート番号は独立
 - セキュアな管理が可能
 - ウィザードベースの簡単操作



http://IP_ADDRESS:4848

ORACLE

コマンドライン管理コンソール

- GlassFish 管理用コマンドを提供
 - \$GLASSFISH_INSTALL/bin/asadmin
 - > **asadmin [オプション] コマンド名 *[--パラメータ 値]**
 - シェルスクリプトを記載し管理の自動化も可能
- コマンドラインからのみ実施可能な操作
 - ドメインの作成・削除
 - ドメインのバックアップ・リストア

```
>./asadmin create-domain -- help
asadmin Utility Subcommands                create-domain(1)
NAME
    create-domain - creates a domain
SYNOPSIS
    create-domain [--help]
    [--adminport adminport]
    [--instanceport instanceport]
    ...
```

asadmin コマンドオプション

--host	ドメイン管理サーバが稼働するホスト名
--port	ドメイン管理サーバが稼働するポート番号
--user	管理者のユーザ名
--password	管理者のパスワード

ORACLE

RESTful 管理チャネル

GlassFish v3.0から追加された新しい管理機能

- RESTful による管理・監視
 - GlassFish v3 から追加された追加管理機能
 - HTML, JSON, XML フォーマットによる出力が可能
 - HTTP ヘッダ:
 - Accept: application/html
 - Accept: application/json
 - Accept: application/xml
 - JAX-RS アプリケーションから管理・監視
 - GET メソッドで設定参照
 - POST メソッドで設定更新

The screenshot displays the 'GlassFish REST Interface' for a 'Domain'. Under the 'Attributes' section, there are four input fields: 'log-root' with the value '\${com.sun.aas.instanceRoot}/logs', 'application-root' with the value '\${com.sun.aas.instanceRoot}/applications', 'locale' which is empty, and 'version' with the value '19'. An 'Update' button is located below these fields. The 'Child Resources' section lists several links: [configs](#), [resources](#), [servers](#), [property](#), [applications](#), [system-applications](#), [stop](#), [restart](#), [uptime](#), [version](#), [rotate-log](#), and [host-port](#).

http://IP_ADDRESS:4848/management/domain

ORACLE

RESTful管理チャネルの例

・ 設定情報の取得例

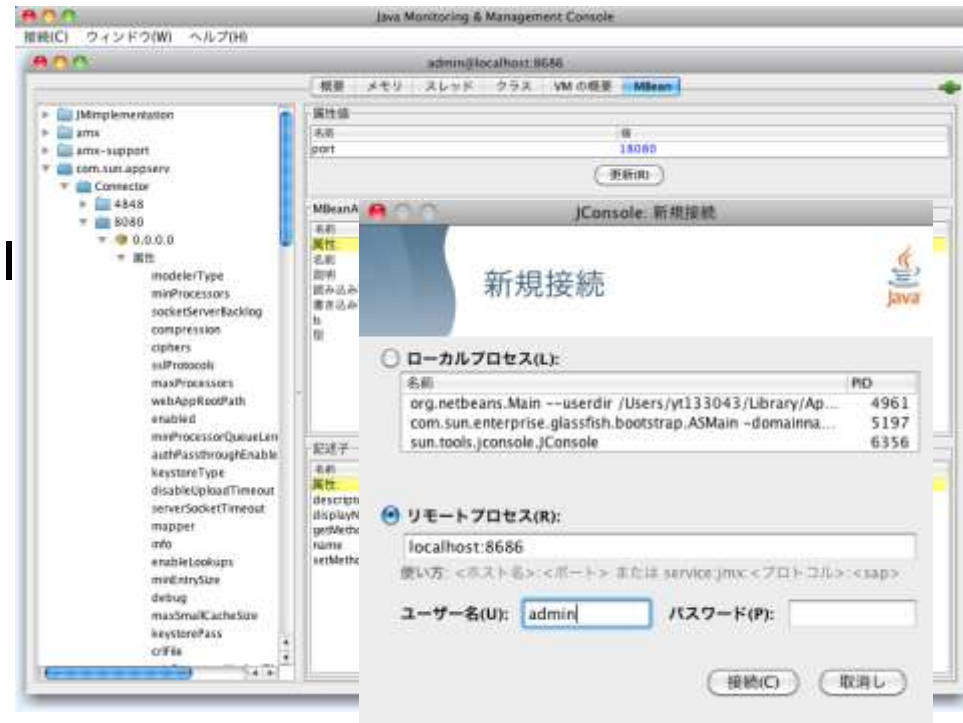
```
> curl -u admin:adminadmin -X GET -H "Accept: application/json" http://localhost:4848/management/domain/configs/config/server-config/network-config/network-listeners/network-listener/http-listener-1
{ "HttpListener1":{"port":"8080", "enabled":"true", "protocol":"http-listener-1" , "transport":"tcp", "address":"0.0.0.0", "name":"http-listener-1", "jdk-enabled" : "false", "thread-pool":"http-thread-pool"},
...中略
```

・ 設定情報の更新例

```
> curl -u admin:adminadmin -X POST -d "port=18080" -H "Accept: application/json" http://localhost:4848/management/domain/configs/config/server-config/network-config/network-listeners/network-listener/http-listener-1
"http://localhost:4848/management/domain/configs/config/server-config/network-config/network-listeners/network-listener/http-listener-1" updated successfully.
```

JMX/AMXベース管理チャネル

- JMX API を実装した Java アプリケーションで管理
 - JMX リスナーポート: 8686
 - 管理者・パスワードは CLI/GUI と同じ
 - 独自管理機能の作成が可能
- JMX コンソールを利用
 - JConsole(JDK に付属)
 - MC4J



再配備時におけるセッションの持続

- 再配備時にセッションを保存
- 開発環境で有効
- コマンドオプション: `-keepSessions=true`
- 統合開発環境と連携
 - Eclipse
 - NetBeans

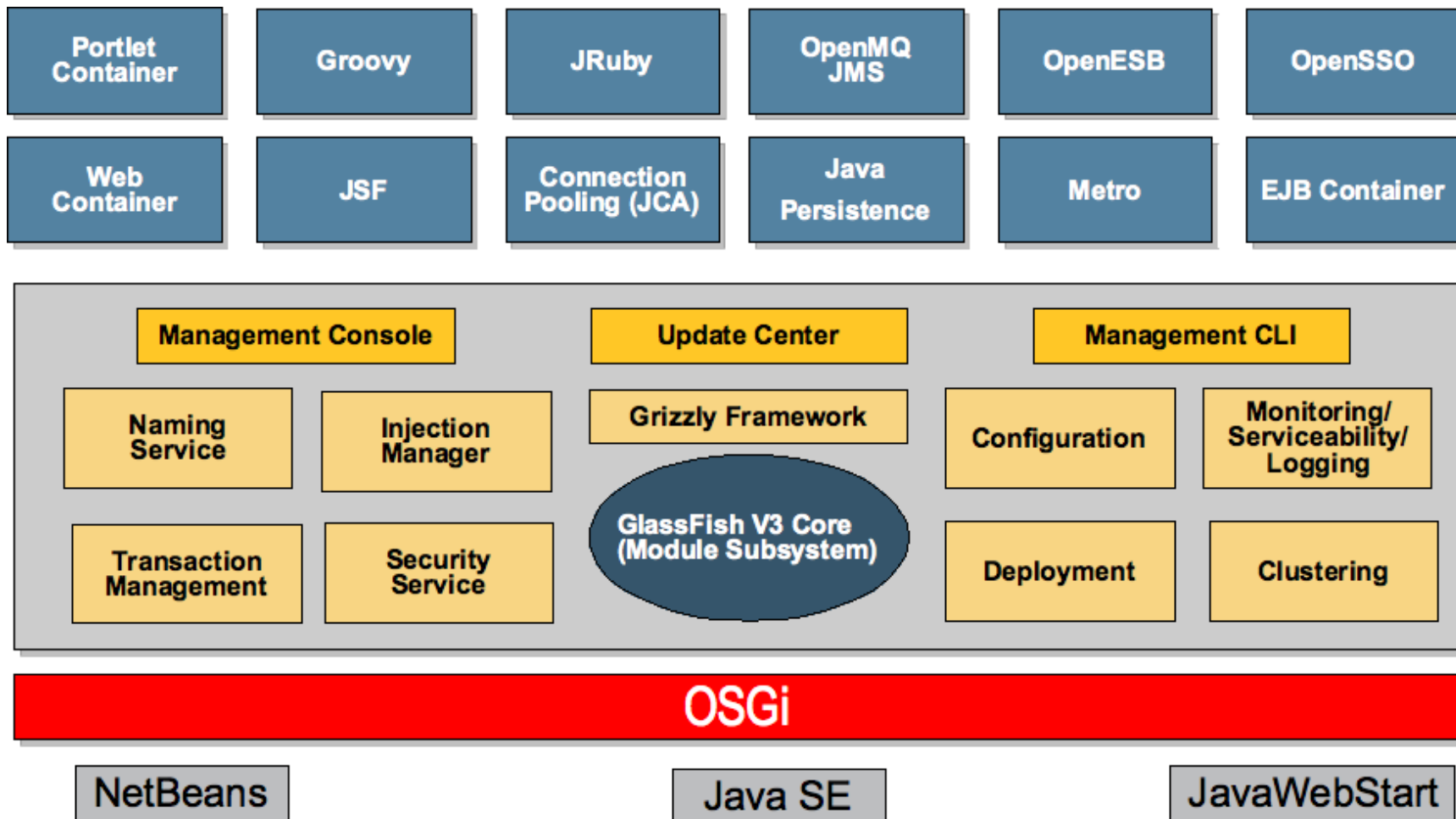




Oracle GlassFish Server v3.x の詳細

GlassFish v3 のアーキテクチャ

OSGi ランタイム上で稼働



ORACLE

OSGi モジュールサブシステムの概要

- モジュール層
 - モジュール=OSGi バンドル
 - 拡張メタデータを持つ jar ファイル
 - モジュールのロード
 - 自身のクラスのロード
 - 外部パッケージのインポート等
- ライフサイクル層
 - インストール・アンインストール起動・停止等の処理を担当
- サービス層
 - サービスは一般的な Java オブジェクト
 - サービスレジストリへ登録・解除
 - サービスレジストリから取得・破棄



GlassFish v3 とモジュールサブシステム

- 2 つのモジュールサブシステムを採用
 - **GlassFish OSGi ランタイム**
 - OSGi Release 4 に準拠したランタイムを利用可能
 - 組み込まれているランタイムは Apache Felix
 - Knopflerfish, Equinox 等も利用可能
 - OSGi ランタイム無しでも動作可能 (static mode)
 - 別途 OSGi の管理ツール(CLI/GUI)も利用可能
 - **GlassFish Kernel (HK2: Hundred-Kilobyte Kernel)**
 - JSR-277 ベースのモジュールサブシステム
 - Java SE 7 へ組み込まれる予定だったため Java SE 7 対応として HK2 を実装→(Java SE 7 でJSR-277は不採用)
 - OSGi 対応前は全ての機能は HK2 のモジュールとして作成
 - OSGi ランタイムだけでは補えないモジュール管理機能を担当

HK2

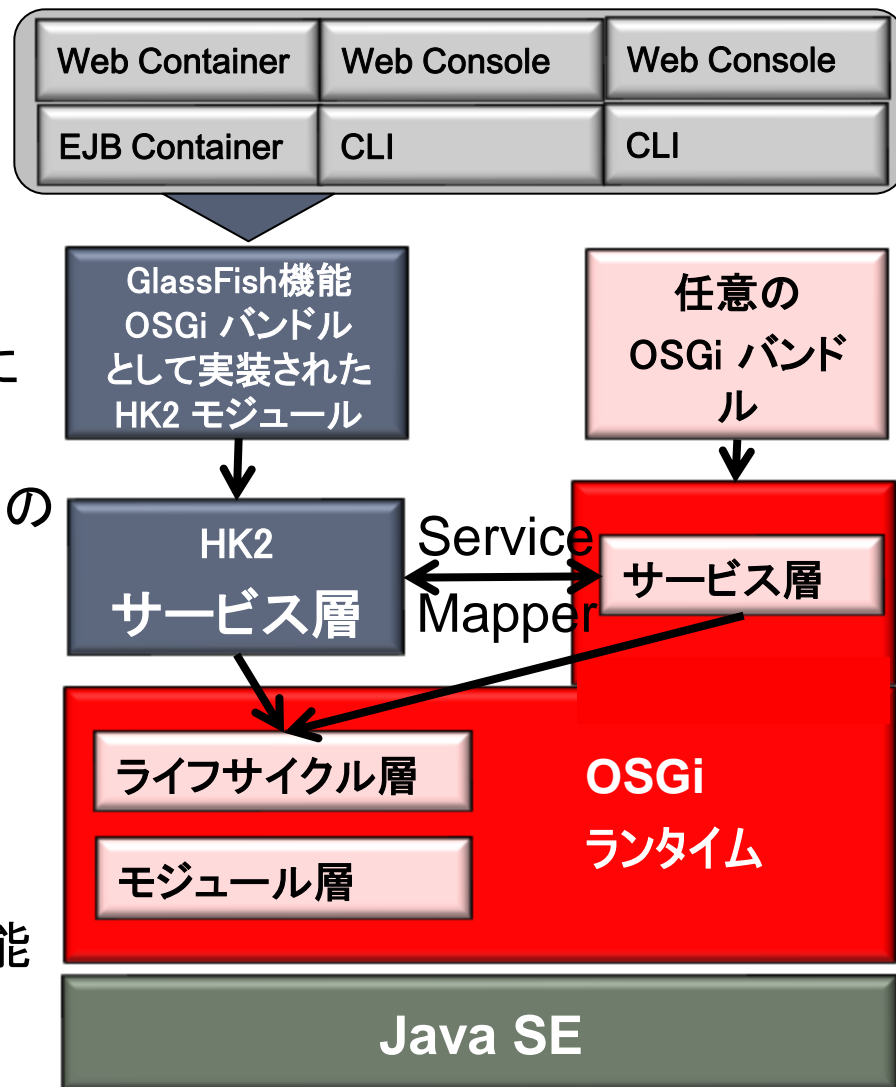
OSGi

Java SE

ORACLE

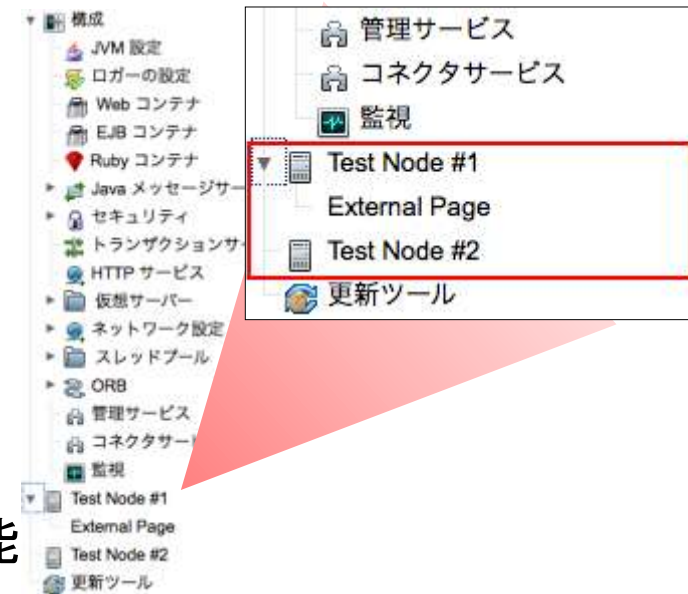
GlassFish における OSGi と HK2 の役割

- OSGi ランタイム
 - デフォルト: Apache Felix を使用
 - 他の OSGi ランタイムも動作可能
- OSGi と HK2
 - モジュール層・ライフサイクル層に OSGi ランタイムを使用
 - サービス層に HK2 を使用し任意の OSGi バンドルと相互運用可能
- GlassFish のモジュール
 - 全ての GlassFish モジュールは HK2 API を使って実装された OSGi バンドル
 - 任意の OSGi バンドルと連携可能
 - @Resource で DI が可能



OSGi の採用によるメリット

- ・ 動的なモジュールの追加・削除が可能
 - ・ 再起動は不要
- ・ 拡張可能なアプリケーションサーバ
 - ・ 独自モジュールを作成し管理機能に組み込み可能
- ・ 任意の OSGi バンドルを利用可能
 - ・ Spring DM(OSGi) と Java EE の連携も可能



GlassFish の拡張可能ポイント

コマンドライン	独自 CLI 管理コマンドの作成(独自追加コンテナの起動・停止等)
Web 管理コンソール	Web 管理コンソールの追加(独自の監視画面等)
監視	独自監視項目に対する整形フォーマットでの出力等
コンテナ	独自コンテナの追加(スクリプト言語用のコンテナ追加等)

GlassFish の OSGi 対応

Apache Felix コンソール

```
% telnet localhost 6666 サーバ起動後に無停止で動的な
Trying 127.0.0.1... モジュールの追加・削除・起動・停止が可能
Connected to localhost.
Escape character is '^]'.
Felix Remote Shell Console:
=====
-> ps (GlassFish v3で導入されているバンドル一覧)
START LEVEL 1
ID State Level Name
[ 0] [Active ] [ 0] System Bundle (2.0.2)
..... (中略)
[ 245] [Installed ] [ 1] MyBundle (1.0.0)
-> start 245
-> stop 245
```

GlassFish v3 のパッケージ管理

- IPS (Image Packaging System: pkg (5)) の採用

- OpenSolaris の成果を利用

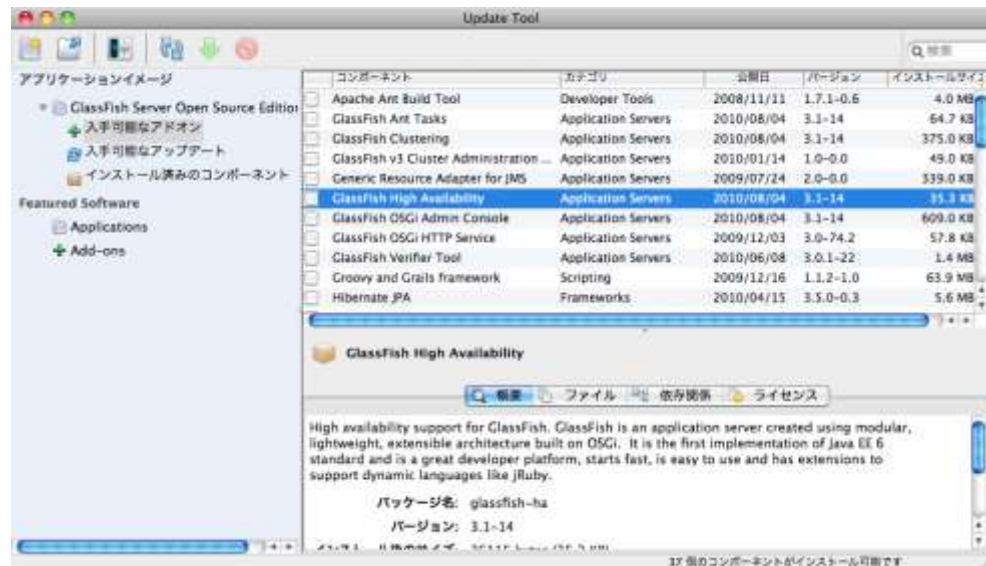
- Pythonを使用
- Zipファイルとして管理

- GlassFish の更新が容易

- インストール済み
- 入手可能なアドオン
- 入手可能なアップデート

- OSGi と IPS

- IPS はネットワーク越しの分散パッケージ管理が可能
- IPSはバージョン管理が容易



```
$ GLASSFISH_INSTALL/pkg/bin/pkg list
```

NAME (PUBLISHER)	VERSION	STATE	UFIX
ant (contrib.glassfish.org)	1.7.1-0.6	installed	----
felix	2.0.2-0	installed	u---
glassfish-appclient	3.0-74.2	installed	u---
glassfish-branding	3.0-74.2	installed	u---
glassfish-branding-gui	3.0-74.2	installed	u---

Swing の GUI ツールとコマンドベース ツール

ORACLE

GlassFish と Grizzly

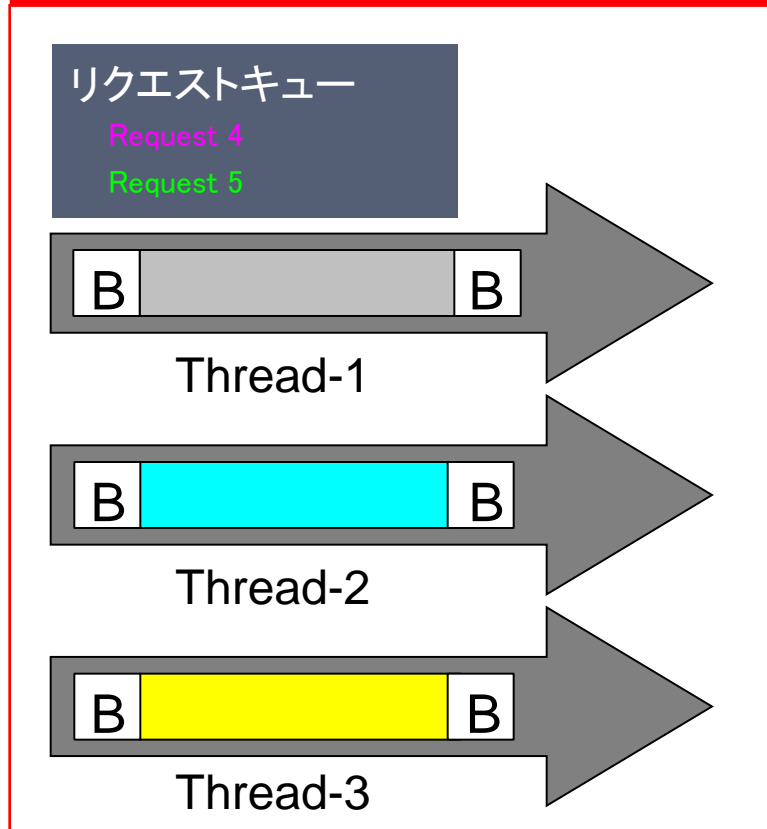


- Grizzly とは

- ネットワークサーバアプリケーションのオープンソースプロジェクト
- Java NIO を採用し Java NIO の複雑なコーディングを隠蔽
- GlassFish の HTTP サーバ実装プロジェクトとして 2004年開始
- TCP/UDP/SSL 等マルチプロトコルに対応しハイパフォーマンス
- ハイパフォーマンスなバッファの使用とバッファ管理を実施
 - Direct Byte Buffer/Heap Byte Buffer/View Byte Buffer
- ハイパフォーマンスなスレッドプールを選択可能
- GlassFish にバンドルされる Grizzly のバージョン
 - GlassFish v2.1.x = Grizzly 1.0.x
 - GlassFish v3.x = Grizzly 1.9.18 or later
- GlassFish v3 より OSGi モジュールとして実装
 - GlassFish v2.1(Grizzly 1.0.x) の実装からアーキテクチャを大幅に変更

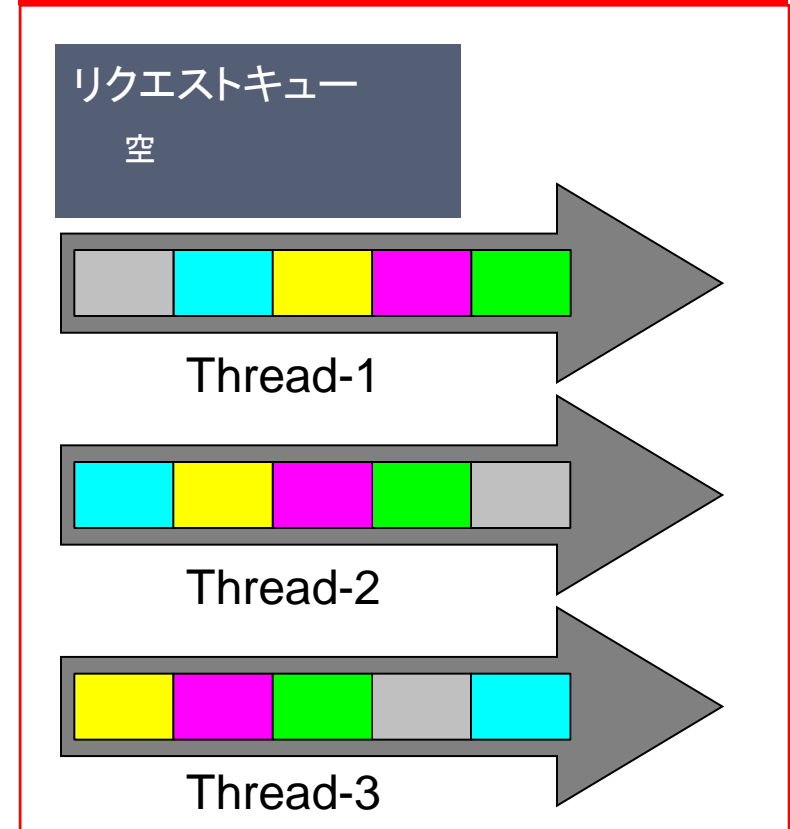
Non Blocking IO (Java New I/O) 対応

同期 I/O (Blocking)



1 処理に1スレッドを占有

非同期 I/O (Non Blocking)

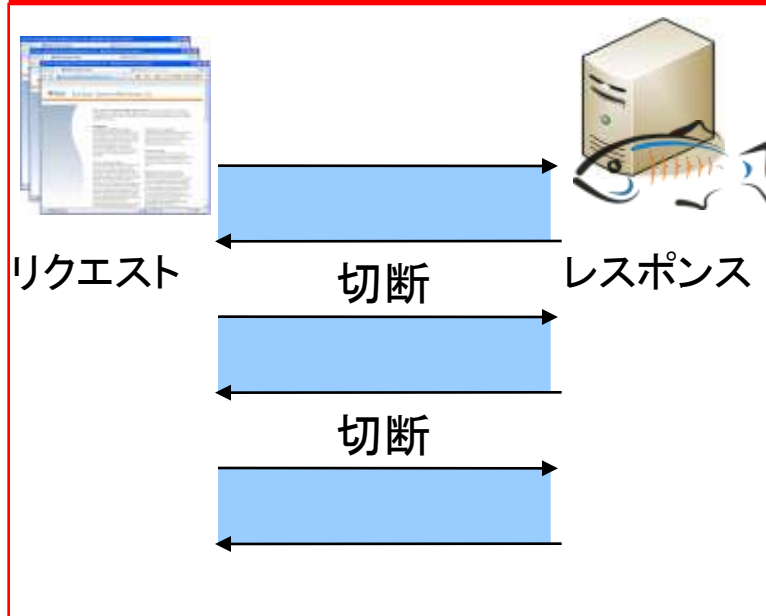


スレッド内で複数の処理を分割

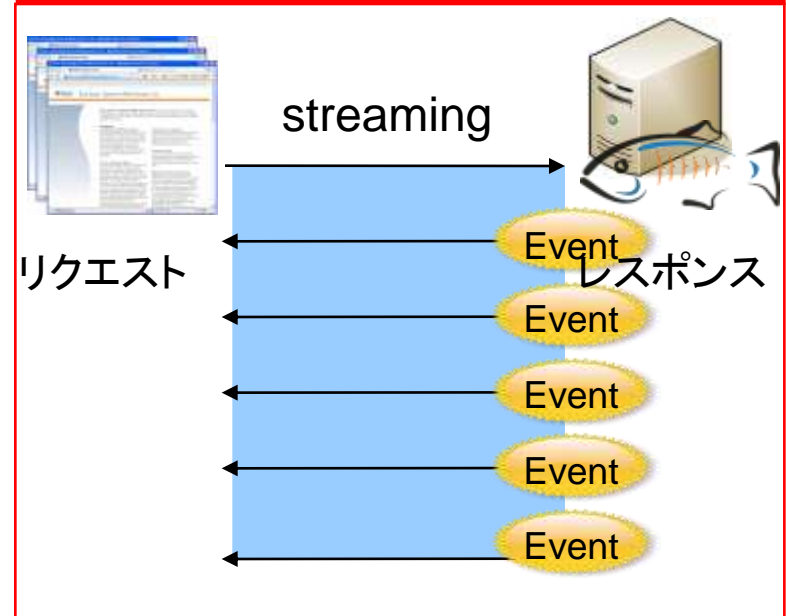
Non Blocking IO (Java New I/O) 対応

- Comet アプリケーション
 - 適用例: 株価情報配信、オークション、チャット等

従来のWebアプリケーション



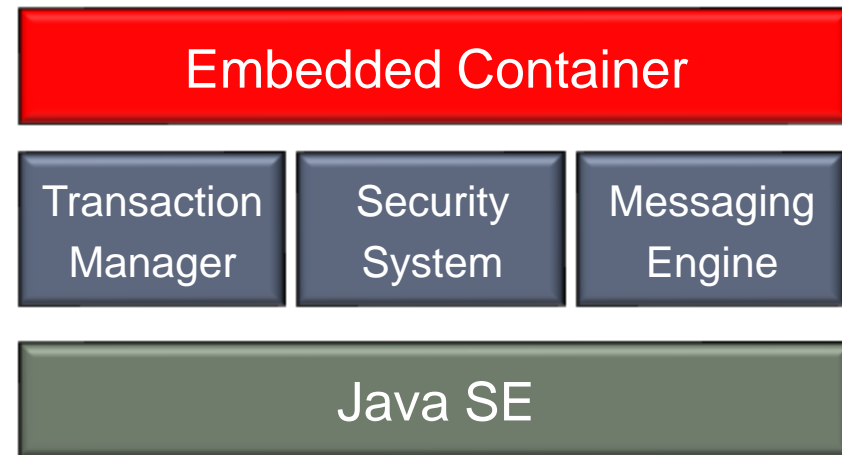
Comet アプリケーション



非同期 I/O は Comet や DB 等の長時間処理が必要な場合に有効

GlassFish 組み込み可能コンテナ

- GlassFish Embedded Server APIを提供
 - EJB 3.1 Embeddable API
 - Java SEプログラム内からGlassFish を起動
- Maven Plug-in の提供
 - Apache Maven との連携
 - JUnit による容易なテスト
- デスクトップアプリケーションでEJB 機能を利用可能



組み込み可能コンテナの概念



GlassFish v3.1の新機能

GlassFish v3.1 の新機能

- アプリケーションバージョンニング
- クラスタ対応
 - SSH プロビジョニング (Node Agent の廃止)
 - ローリングアップグレード機能の提供
- WebSocket 対応
- WebLogic との互換性を提供
 - WebLogic デプロイメント記述子をサポート
- 管理・監視機能の強化 (Oracle GlassFish Server)
 - DTrace を使用したモニタリング (Java SE 7 使用時)
- Coherence *Web を統合 (製品版)

アプリケーションバージョンニング

- 同一アプリケーションの複数配備が可能
 - MS1, Beta1, RC1, GA 版等複数を配備可能
 - 単一バージョンのみ有効
 - アプリケーションのアップグレード・ロールバックが可能
 - 配備数に上限無し

Applications

Applications can be enterprise or web applications, or various kinds of modules.

Deployed Applications (3)

Deploy...

Undeploy

Enable

Disable

Filter:

	Name	Enabled	Engines	Action
<div></div>	HelloWorld:Beta	✖	web	<div> Redeploy Reload</div>
<div></div>	HelloWorld:GA	✔	web	<div>Launch Redeploy Reload</div>
<div></div>	HelloWorld:MS1	✖	web	<div> Redeploy Reload</div>

アプリケーションバージョンニング

同一アプリケーションの複数配備

```
> asadmin deploy --name=HelloWorld:MS1 ./
HelloWorldWeb-MS1.war

> asadmin deploy --enabled=false --
name=HelloWorld:Beta ./HelloWorldWeb-Beta.war
Application deployed with name HelloWorld:Beta.
Command deploy executed successfully.

> asadmin deploy --enabled=false --
name=HelloWorld:GA ./HelloWorldWeb-GA.war
Application deployed with name HelloWorld:GA.
Command deploy executed successfully.
```

アプリケーションバージョンニング

アプリケーションの有効化と確認

```
> asadmin enable HelloWorld:GA
```

```
Command enable executed successfully.
```

```
> asadmin list-applications -l
```

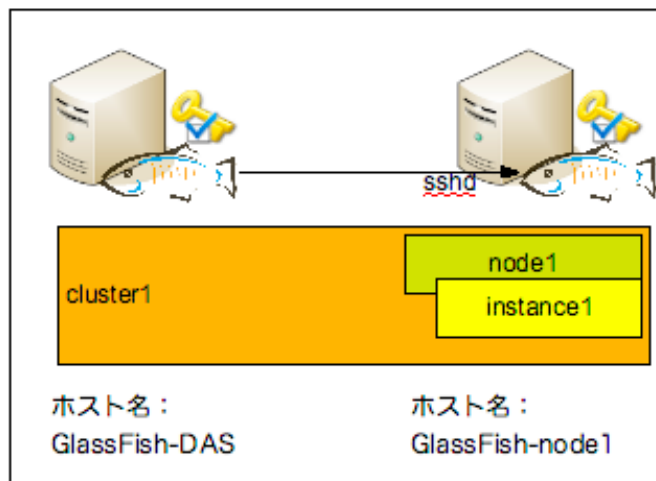
NAME	TYPE	STATUS
HelloWorld:MS1	<web>	disabled
HelloWorld:Beta	<web>	disabled
HelloWorld:GA	<web>	enabled

```
Command list-applications executed successfully.
```

刷新されたクラスタ機能

自己増殖クラスタ

- ドメイン管理サーバが稼働するOSと同一環境
 - 例:DAS: が Windows で Node が Unix は未サポート
- 対象ノードでは sshd の起動が必須
 - Windows 環境は Cygwin を利用
- 対象ノードへのインストールはSSH経由で実施

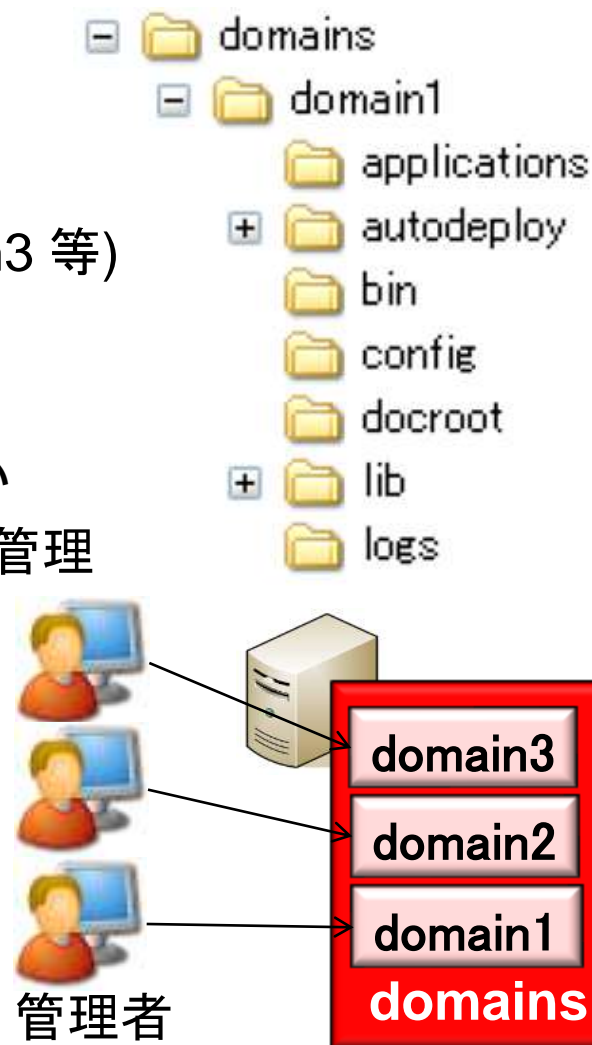


環境構築デモ(約 8 分):

<http://tinyurl.com/3yto2gc>

GlassFish のドメイン

- GlassFish の管理構成単位
 - デフォルトで作成されるドメイン: domain1
 - 複数のドメインを作成可能 (domain2, domain3 等)
 - ドメイン毎に管理者を設定
 - 各ドメイン間は独立して管理
 - 設定変更は他のドメインに影響を及ぼさない
 - 管理者は管理コンソール等を通じてドメインを管理
 - Web 管理コンソール
 - コマンドライン管理コンソール
 - RESTful 管理チャネル
 - JMX/AMX 管理チャネル



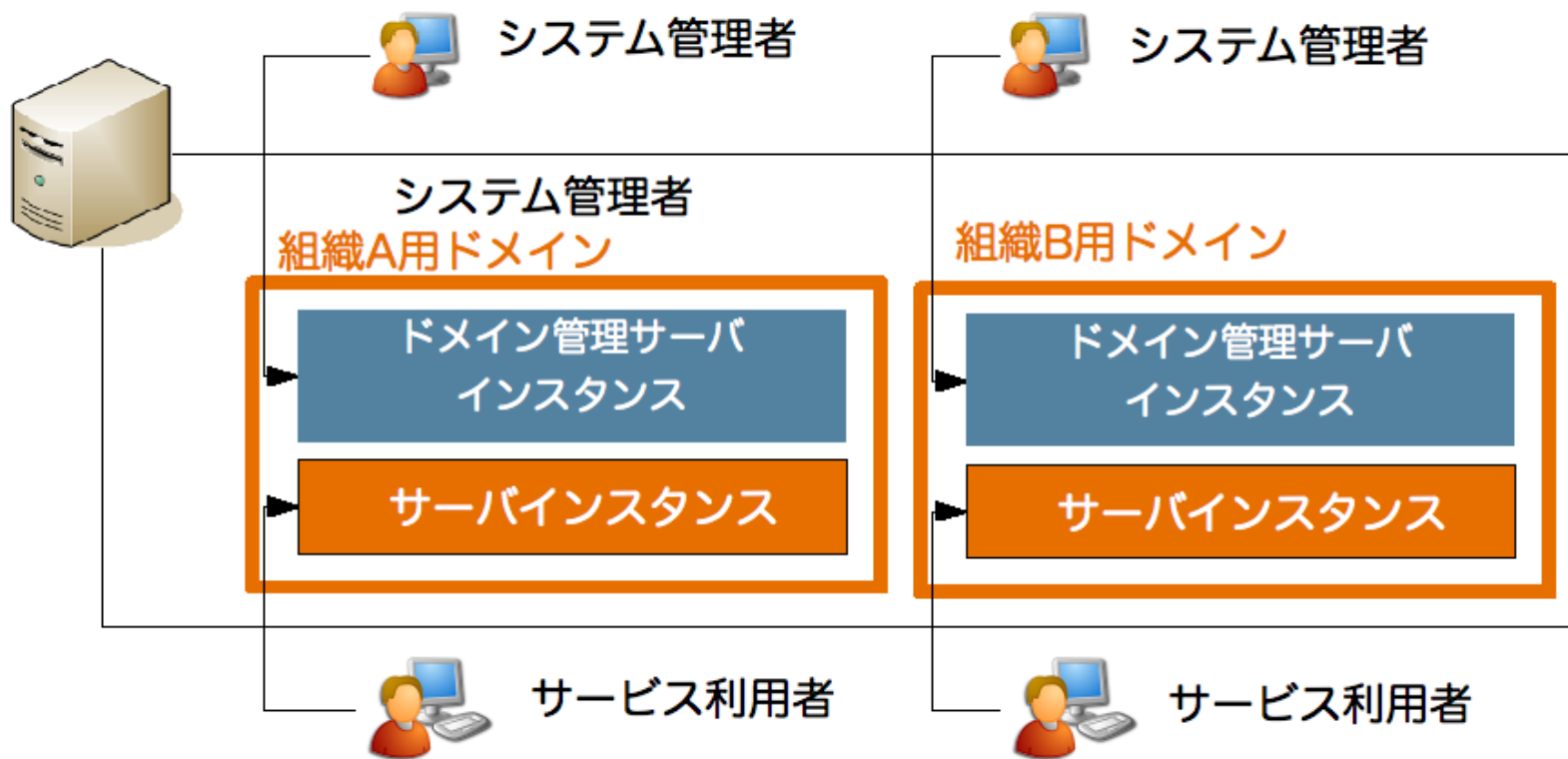
ドメイン管理の基本

- ドメイン管理の基本最小構成
 - 開発環境・小規模環境に適用可能

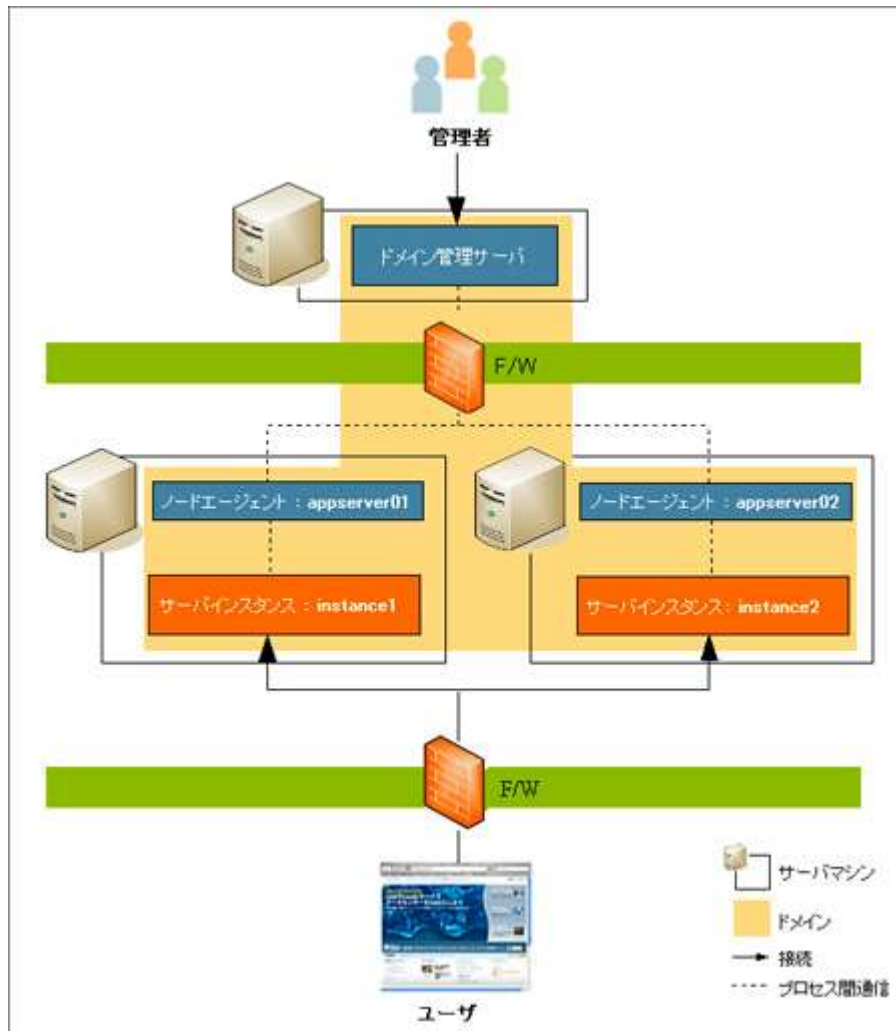


複数ドメイン構成

- 1 インストーラで複数ドメインを構築



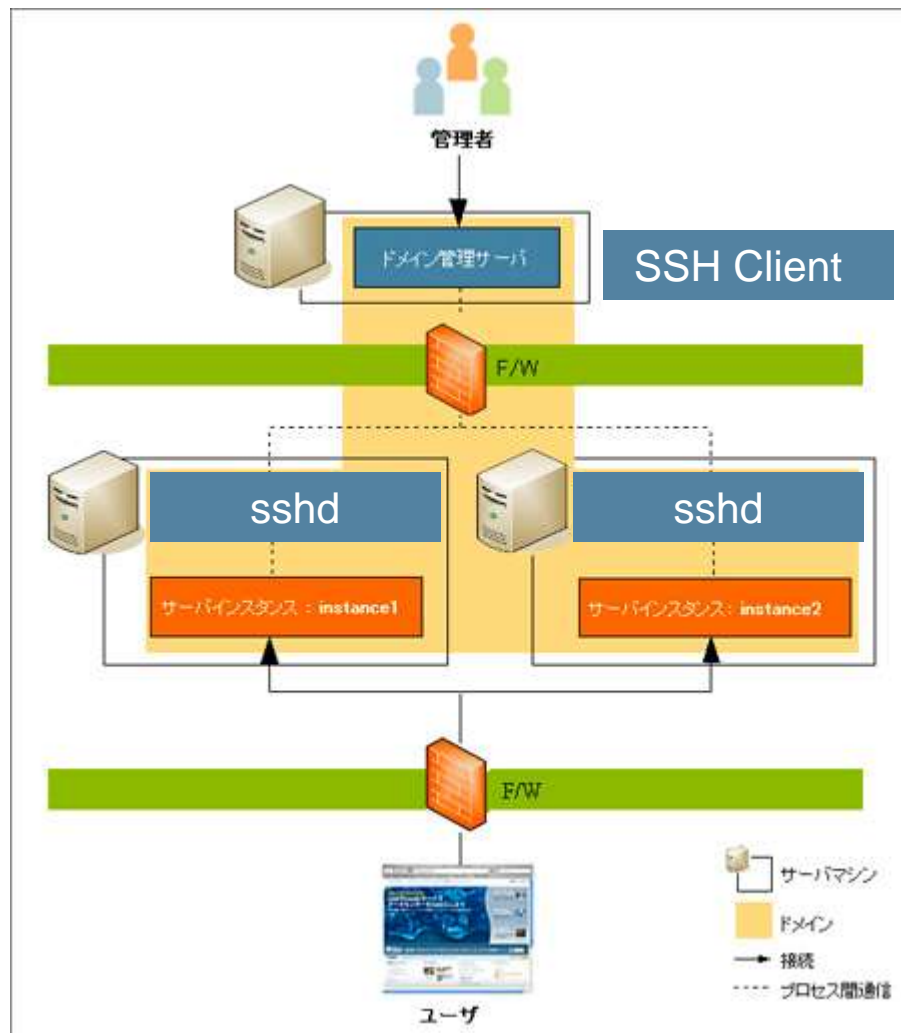
GlassFish v2.1.1のクラスタ機能



• ノードエージェント方式

- 各物理ノードにGlassFishをインストール
- 各物理ノードにノードエージェントを作成
- ノードエージェントの起動・停止は各マシン上で実施
- ドメイン管理サーバから各ノードエージェントのインスタンスを管理

GlassFish v3.1のクラスタ新機能

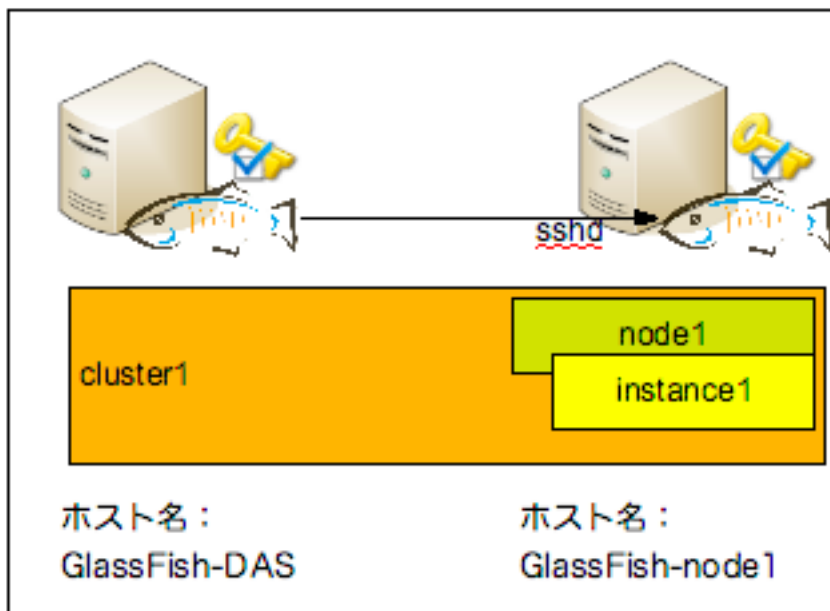


- **SSH プロビジョニング方式**
 - ドメイン管理サーバ内に SSH Client ライブラリを統合
 - Hudson(Trilead-ssh2)の成果物を利用
 - 各物理ノードでsshdを起動
 - システムの初期化とドメイン管理サーバへの登録
 - GlassFish v3.1 M3 より EA 提供

Demo

SSH プロビジョニングによるインストール

全ての管理は管理サーバから



リモートノードはsshdが起動しているだけ
インストールから各種操作は全て DAS から

1. GlassFish DASへインストール
2. ドメイン起動
3. クラスタ作成
4. SSH鍵生成
5. リモートのノードへGlassFishをインストール
6. リモートノードの作成
7. リモートでインスタンス生成
8. リモートのインスタンス起動
9. サンプルアプリの動作確認

参考 URL : <http://tinyurl.com/3yto2gc>

クラスタ環境のログ収集

リモートのログも収集可能

```
> asadmin collect-log-files --target cluster1  
Log files are downloaded for instance1.  
Log files are downloaded for instance2.  
Created Zip file under  
/GlassFishv3.1/glassfish3/glassfish/domains/  
domain1/collected-logs/log_2010-12-19_18-42-  
07.zip.  
Command collect-log-files executed successfully.
```

WebSocket 対応

WebSocket とは

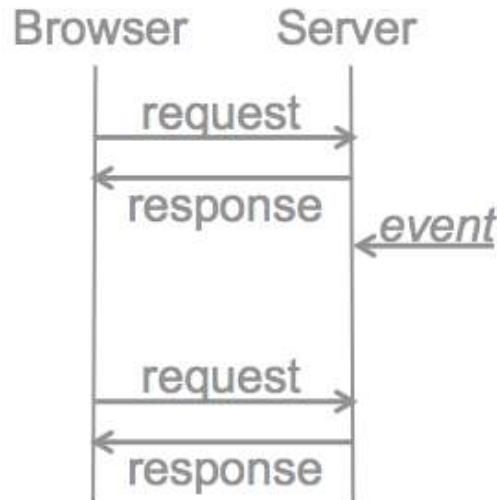
- WebSocket は単一の TCPソケットで、双方向、全2重 (Full Duplex)の通信チャネルの技術を提供し、Web ブラウザ、Web サーバの双方で実装されるプロトコル

Comet のアプローチ

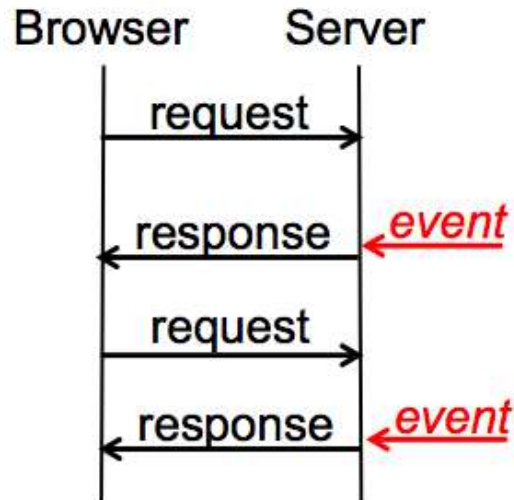
- Long Polling 方式
 - イベント発生時にレスポンスを返信し再接続
- Streaming 方式

当社は、この技術を採用しています。

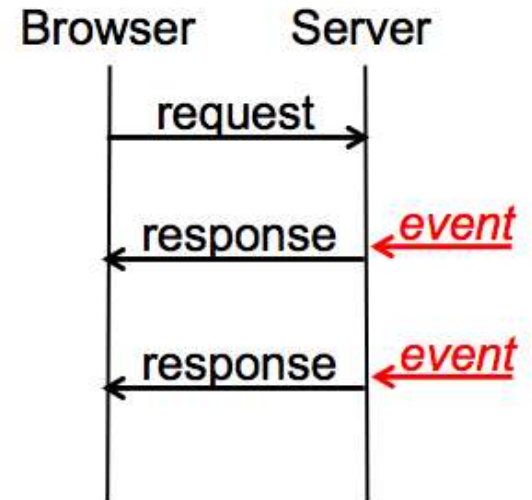
HTTP Polling



Comet Long Polling



Comet Streaming



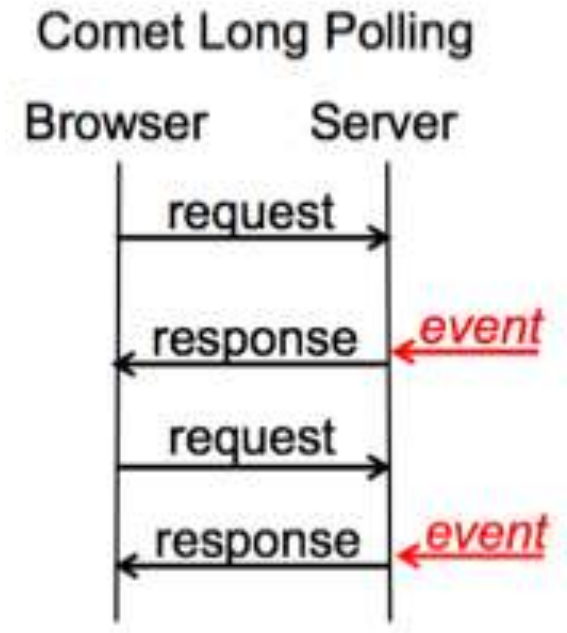
Comet Long Polling 方式の利点と欠点

- 利点

- HTTP 1.1 準拠
- クライアントサイド APIの標準化
- Hidden iframe, XMLHttpRequest
- 広範囲への適用

- 欠点

- HTTPヘッダによる負荷
- ネットワーク帯域、CPU



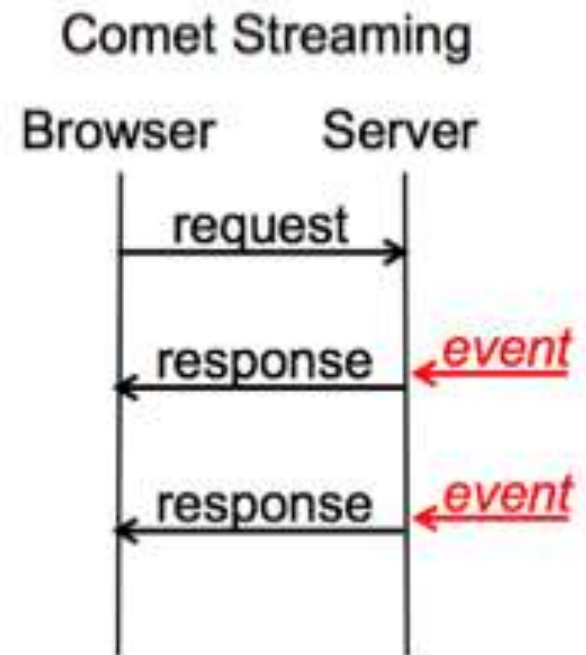
Comet Streaming 方式の利点と欠点

- 利点

- HTTP 1.1 準拠
- クライアントサイドAPIの標準化
 - Hidden iframe, XMLHttpRequest
- Long polling に比べ高効率

- 欠点

- ブラウザ毎の対応が困難
- Proxy Server 導入時の問題発生の可能性



WebSocket の特徴

- 双方向通信
- 全2重
- 単一 TCP ソケットを利用した操作
- プレイン/SSL ソケット共に利用可能
- HTTP に類似
 - ws://HOST:PORT/CONTEXT

WebSocket の利点と欠点

- 利点

- 効果的な双方向、全2重通信
- 単一TCPソケット
- Proxy/Firewall環境で利用可能

- 欠点

- クライアントブラウザの対応
- プロトコル自体継続議論中

WebSocket API

4 The WebSocket interface

```
[Constructor(in DOMString url, in optional DOMString protocols)]
[Constructor(in DOMString url, in optional DOMString[] protocols)]
interface WebSocket {
    readonly attribute DOMString url;

    // ready state
    const unsigned short CONNECTING = 0;
    const unsigned short OPEN = 1;
    const unsigned short CLOSING = 2;
    const unsigned short CLOSED = 3;
    readonly attribute unsigned short readyState;
    readonly attribute unsigned long bufferedAmount;

    // networking
        attribute Function onopen;
        attribute Function onmessage;
        attribute Function onerror;
        attribute Function onclose;
    readonly attribute DOMString protocol;
    void send(in DOMString data);
    void close();
};
WebSocket implements EventTarget;
```

<http://dev.w3.org/html5/websockets/>

1000 イベントのデータ転送量

	1 client	2 clients	100 clients
WebSockets	11 Kb	44 Kb	102 Mb
Comet "Streaming"	66 Kb	153 Kb	112 Mb
Comet "Long Polling"	112 Kb	338 Kb	746 Mb

WebSocket が最も効率的

WebSocket サンプルの入手先



- grizzly-websockets-chat-1.9.26-javadoc.jar
- grizzly-websockets-chat-1.9.26-javadoc.jar.md5
- grizzly-websockets-chat-1.9.26-javadoc.jar.sha1
- grizzly-websockets-chat-1.9.26-sources.jar
- grizzly-websockets-chat-1.9.26-sources.jar.md5
- grizzly-websockets-chat-1.9.26-sources.jar.sha1
- grizzly-websockets-chat-1.9.26.pom
- grizzly-websockets-chat-1.9.26.pom.md5
- grizzly-websockets-chat-1.9.26.pom.sha1
- grizzly-websockets-chat-1.9.26.war
- grizzly-websockets-chat-1.9.26.war.md5
- grizzly-websockets-chat-1.9.26.war.sha1

入手先: <http://tinyurl.com/2aejyfa>



Java EE 7 について

テーマ:クラウド

Java EE 7 に含まれる技術



- JPA 2.1 (JSR-338)
- JAX-RS 2.0 (JSR-339)
- JMS 2.0
- JavaServer Faces 2.2
- WebTier
 - HTML 5対応
 - WebSocket 対応
 - JSON API のサポート

まとめ

- **Java EE 6 の参照実装**
 - Java EE 6 の仕様に完全準拠
 - 軽量、高速起動
 - かんたん開発／かんたん管理
 - プロファイルに対応
 - Webプロファイル版、Full Platform プロファイル版を提供
- **先進的アプリケーションサーバ**
 - RESTful の管理インタフェースを提供
 - OSGiモジュールサブシステム対応
 - 進化した非同期 I/O サーバ
- **かんたんな操作**
 - unzipによるインストール(Tomcatと同様)
- **Oracle JRockit VM 正式対応**

まずはお試ください

OTN × ダイセミ でスキルアップ!!



- ・一般的な技術問題解決方法などを知りたい！
- ・ 세미나資料など技術コンテンツがほしい！

Oracle Technology Network(OTN)を御活用下さい。

<http://forums.oracle.com/forums/category.jspa?categoryId=492>

一般的技術問題解決にはOTN揭示版の
「Java」をご活用ください

※OTN揭示版は、基本的にOracleユーザー有志からの回答となるため100%回答があるとは限りません。
ただ、過去の履歴を見ると、質問の大多数に関してなんらかの回答が書き込まれております。

<http://www.oracle.com/technetwork/jp/java/index.html>

過去のセミナー資料、動画コンテンツはOTNの
「OTNセミナー オンデマンドコンテンツ」へ

※ダイセミ事務局にダイセミ資料を請求頂いても、お受けできない可能性がございますので予めご了承ください。
ダイセミ資料はOTNコンテンツ オン デマンドか、セミナー実施時間内にダウンロード頂くようお願い致します。

ORACLE

OTNセミナー オンデマンド コンテンツ

ダイセミで実施された技術コンテンツを動画で配信中!!

ダイセミのライブ感はそのままに、お好きな時間で受講頂けます。

最新のコンテンツ



エンジニアのための
ITIL実践術
再生時間: 60分



ここからはじめよう
Oracle PL/SQL入門
再生時間: 60分



実践!!高可用システム構築
-RAC基本
再生時間: 60分



お悩み解決! Oracle
のサイジング
再生時間: 60分

Database



今さら聞けない!? バックアップ・リカバリ
再生時間: 60分



意外と簡単!? Oracle
Database 11g -セ
再生時間: 60分



実践!!バックアップ・リカバリ
再生時間: 60分



意外と簡単!? Oracle
Database 11g -デ
再生時間: 60分

>> もっと見る

twitter

最新情報つぶやき中
oracletechnetjp

- ・人気コンテンツは?
- ・お勧め情報
- ・公開予告 など

OTN オンデマンド

検索

※掲載のコンテンツ内容は予告なく変更になる可能性があります。

期間限定での配信コンテンツも含まれております。お早めにダウンロード頂くことをお勧めいたします。

ORACLE

Oracle エンジニアのための技術情報サイト オラクルエンジニア通信

<http://blogs.oracle.com/oracle4engineer/>

twitter

最新情報つぶやき中
oracletechnetjp

技術資料

- ダイセミの過去資料や製品ホワイトペーパー、スキルアップ資料などを多様な方法で検索できます
- キーワード検索、レベル別、カテゴリ別、製品・機能別

コラム

- オラクル製品に関する技術コラムを毎週お届けします
- 決してニッチではなく、誰もが明日から使える技術の「あ、そうだったんだ！」をお届けします



こんな資料が人気です

- ✓ 6か月ぶりに資料ダウンロードランキングの首位が交代！
新王者はOracle Database構築資料でした。
- ✓ データベースの性能管理手法について、Statspack派もEnterprise Manager派も目からウロコの技術特集公開中

オラクルエンジニア通信



ORACLE

ITプロジェクト全般に渡る無償支援サービス

Oracle Direct Conciergeサービス

■ パフォーマンス診断サービス

- Webシステム ボトルネック診断サービス **NEW**
- データベースパフォーマンス 診断サービス

■ 移行支援サービス

- SQL Serverからの移行支援サービス
- DB2からの移行支援サービス
- Sybaseからの移行支援サービス
- MySQLからの移行支援サービス
- Postgre SQLからの移行支援サービス
- Accessからの移行支援サービス
- Oracle Application ServerからWeblogicへ移行支援サービス **NEW**

■ システム構成診断サービス

- Oracle Database構成相談サービス
- サーバー統合支援サービス
- 仮想化アセスメントサービス
- メインフレーム資産活用相談サービス
- BI EEアセスメントサービス
- 簡易業務診断サービス

■ バージョンアップ支援サービス

- Oracle Databaseバージョンアップ支援サービス
- Weblogic Serverバージョンアップ支援サービス **NEW**
- Oracle Developer/2000(Forms/Reports) Webアップグレード相談サービス

オラクル社のエンジニアが 直接ご支援します
お気軽にご活用ください!

オラクル 無償支援

検索

ORACLE



1日5組限定！

製品無償評価サービス

提供シナリオ一例

- ・データベースチューニング
- ・無停止アップグレード
- ・アプリケーション性能・負荷検証
- ・Webシステム障害解析

インストールすることなく、すぐに体験いただけます

- サービスご提供までの流れ

1. お問い合わせフォームより「製品評価サービス希望」と必要事項を明記し送信下さい
2. 弊社より接続方法手順書およびハンズオン手順書を送付致します
3. 当日は、弊社サーバー環境でインターネット越しに製品を体感頂けます

※サービスご提供には事前予約が必要です

Web問い合わせフォーム

「ダイデモ」をキーワードに検索することで申し込みホームページにアクセスできます

<http://www.oracle.com/jp/direct/services/didemo-195748-ja.html>

ORACLE

あなたにいちばん近いオラクル



Oracle Direct

まずはお問合せください

Oracle Direct

検索

システムの検討・構築から運用まで、ITプロジェクト全般の相談窓口としてご支援いたします。

システム構成やライセンス/購入方法などお気軽にお問い合わせ下さい。

Web問い合わせフォーム

専用お問い合わせフォームにてご相談内容を承ります。

<http://www.oracle.com/jp/direct/inquiry-form-182185-ja.html>

※こちらから詳細確認のお電話を差し上げる場合がありますので、ご登録されている連絡先が最新のものになっているか、ご確認下さい。

フリーダイヤル

0120-155-096

※月曜～金曜 9:00～12:00、13:00～18:00

(祝日および年末年始除く)

ORACLE